ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA

DOTTORATO DI RICERCA IN
COMPUTER SCIENCE AND ENGINEERING

Ciclo 34

Settore Concorsuale: 09/H1 - SISTEMI DI ELABORAZIONE DELLE INFORMAZIONI

Settore Scientifico Disciplinare: ING-INF/05 - SISTEMI DI ELABORAZIONE DELLE INFORMAZIONI

# Towards Unstructured Knowledge Integration in Natural Language Processing

Presentata da:

**Federico Ruggeri**

*Coordinatore Dottorato*
Prof. Davide Sangiorgi

*Supervisore*
Prof. Paolo Torroni

*Co-supervisori*
Prof. Marco Lippi

Esame Finale Anno 2022

# Abstract

In the last decades, Artificial Intelligence has witnessed multiple breakthroughs in deep learning. In particular, purely data-driven approaches have opened to a wide variety of successful applications due to the large availability of data. Nonetheless, the integration of prior knowledge is still required to compensate for specific issues like lack of generalization from limited data, fairness, robustness, and biases.

In this thesis, we analyze the methodology of integrating knowledge into deep learning models in the field of Natural Language Processing (NLP). We start by remarking on the importance of knowledge integration. We highlight the possible shortcomings of these approaches and investigate the implications of integrating unstructured textual knowledge.

We introduce Unstructured Knowledge Integration (UKI) as the process of integrating unstructured knowledge into machine learning models. We discuss UKI in the field of NLP, where knowledge is represented in a natural language format. We identify UKI as a complex process comprised of multiple sub-processes, different knowledge types, and knowledge integration properties to guarantee. We remark on the challenges of integrating unstructured textual knowledge and bridge connections with well-known research areas in NLP.

We provide a unified vision of structured knowledge extraction (KE) and UKI by identifying KE as a sub-process of UKI. We investigate some challenging scenarios where structured knowledge is not a feasible prior assumption and formulate each task from the point of view of UKI. We adopt simple yet effective neural architectures and discuss the challenges of such an approach.

Finally, we identify KE as a form of symbolic representation. From this perspective, we remark on the need of defining sophisticated UKI processes to verify the validity of knowledge integration. To this end, we foresee frameworks capable of combining symbolic and sub-symbolic representations for learning as a solution.

# Table of Contents

**Abstract** iii

**Table of Contents** v

**List of Figures** ix

**List of Tables** xi

**1 Learning with Structured Knowledge** 3

1.1 Knowledge Integration in Natural Language Processing . . . . . . . . . . . . . 4

1.2 Structured Knowledge . . . . . . . . . . . . . . . . . . . . . . . . . . . 6

1.2.1 Knowledge Graphs and Knowledge Bases . . . . . . . . . . . . . . . 6

1.2.2 Logic Rules . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

1.2.3 Linguistic Dependencies . . . . . . . . . . . . . . . . . . . . . . 9

1.3 Challenges . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

1.4 Contributions and Outline . . . . . . . . . . . . . . . . . . . . . . . . 10

**2 Background** 13

2.1 A Simple Interpretation of Knowledge Integration . . . . . . . . . . . . . . . 13

2.2 Unstructured Knowledge Integration . . . . . . . . . . . . . . . . . . . . 15

2.3 Unstructured Knowledge Integration Processes . . . . . . . . . . . . . . . . 16

2.3.1 Knowledge Integration . . . . . . . . . . . . . . . . . . . . . . . 17

2.3.2 Knowledge Mapping . . . . . . . . . . . . . . . . . . . . . . . . 19

2.3.3 Knowledge Validation . . . . . . . . . . . . . . . . . . . . . . . 19

2.4 Unstructured Knowledge Integration Properties . . . . . . . . . . . . . . . . 21

2.4.1 Knowledge Update . . . . . . . . . . . . . . . . . . . . . . . . . 21

2.4.2 Integration Efficiency . . . . . . . . . . . . . . . . . . . . . . . . 22

2.4.3 Integration Transparency . . . . . . . . . . . . . . . . . . . . . . 23

2.4.4 Integration Robustness . . . . . . . . . . . . . . . . . . . . . . . 23

2.5 The Role of Knowledge . . . 24
2.5.1 Source of Complementary Information . . . 24
2.5.2 Support for Reasoning . . . 24
2.5.3 Behavioural . . . 25
2.6 Unstructured Knowledge Types . . . 26
2.6.1 Textual Patterns . . . 26
2.6.2 Text Attributes . . . 28
2.6.3 Class Descriptors . . . 29
2.6.4 Class Rationales . . . 31
2.6.5 Additional Data . . . 33
2.6.6 Contextual Data . . . 34
2.6.7 Text Constraints . . . 35
2.6.8 Considerations on Knowledge Types . . . 37
2.7 Considerations on Unstructured Knowledge Integration . . . 38
2.7.1 Advantages . . . 38
2.7.2 Dealing with Natural Language Knowledge . . . 39
2.8 Related Work . . . 40
2.8.1 Reading Comprehension . . . 40
2.8.2 Few- And Zero-shot Learning . . . 41
2.8.3 Knowledge-Enhanced Text Generation . . . 42

**3 Knowledge Extraction from Text 45**
3.1 Knowledge Representation . . . 46
3.2 Challenges . . . 47
3.3 Argument Detection with Sub-Tree Structures . . . 47
3.3.1 Background . . . 48
3.3.2 Tree-Constrained GNNs . . . 50
3.3.3 Issues . . . 53
3.3.4 Adaptive Learning . . . 55
3.3.5 Experiments . . . 56
3.3.6 Discussion . . . 61
3.4 Arguments for Scientific Dialogues . . . 66
3.4.1 Dataset Collection . . . 68
3.4.2 Analysis . . . 71
3.4.3 Experiments . . . 78
3.4.4 Discussion . . . 83
3.5 Considerations on Knowledge Extraction . . . 85

**4 Unstructured Knowledge Integration** **89**
4.1 Knowledge Representation . . . 90
4.2 Challenges . . . 91
4.2.1 Poor Knowledge Quality . . . 91
4.2.2 Supervised Knowledge Mapping . . . 92
4.2.3 Partial Knowledge Mapping . . . 92
4.2.4 Dealing with Incorrect Knowledge Mapping . . . 93
4.3 Unfair Clause Detection with Legal Rationales . . . 93
4.3.1 Data . . . 94
4.3.2 Method . . . 97
4.3.3 Experiments . . . 99
4.3.4 A New CLAUDETTE Tool . . . 102
4.3.5 Discussion . . . 103
4.4 Unfairness Detection with Strong Supervision . . . 105
4.4.1 Data . . . 106
4.4.2 Method . . . 108
4.4.3 Weak and Strong Partial Supervision . . . 109
4.4.4 Experiments . . . 111
4.4.5 Discussion . . . 115
4.5 Sampling from Knowledge with Transformers . . . 117
4.5.1 Method . . . 118
4.5.2 Experimental Setup . . . 124
4.5.3 Memory Analysis . . . 127
4.5.4 Results . . . 128
4.5.5 Error Analysis and Discussion . . . 134
4.6 Considerations on UKI . . . 136

**5 Open Challenges and Future Research Directions** **139**
5.1 KE as a Sub-process of UKI . . . 140
5.2 KE as a Symbolic Representation . . . 142
5.3 Neural-Symbolic Reasoning for UKI . . . 143
5.3.1 Neural-Symbolic Learning: A Brief Overview . . . 143
5.3.2 NeSy for UKI . . . 145
5.4 Application Examples . . . 146
5.4.1 Text Classification with Class Descriptors . . . 147
5.4.2 Substitution-based Pattern Matching . . . 149

**A GNN architecture calibration** 157

**B Data Collection Framework Implementation** 159

**C Legal Rationales for Unfair Clause Detection** 163

**D Partial Strong Supervision in Unfair Clause Detection** 173

**Bibliography** 177

# List of Figures

2.1 UKI sub-processes . . . . . 17
2.2 The roles of knowledge. . . . . 25

3.1 Tree-constrained GNN architecture. . . . . 50
3.2 Most frequent class-specific tree fragments. . . . . 62
3.3 An example of execution of MARGOT. . . . . 67
3.4 An example dialogue from ArgSciChat dataset. . . . . 68
3.5 The data collection framework for ArgSciChat. . . . . 70
3.6 Dialogue diversity indicators based on SentenceBert. . . . . 74
3.7 Argumentative models class distribution in ArgSciChat and DrInventor datasets. 77
3.8 Argumentative graph measures in ArgSciChat. . . . . 78
3.9 Problem setup for Orig and Split dataset versions of ArgSciChat. . . . . 79
3.10 Argumentative dialogue generation. . . . . 86

4.1 Architecture of the MANN model for unfairness detection. . . . . 100
4.2 Cumulative memory distribution across all test folds . . . . . 102
4.3 Example of classification performed by the CLAUDETTE tool. . . . . 103
4.4 MANN architecture for unfair clause detection with supervision. . . . . 110
4.5 Memory-augmented transformer architecture. $M$ denotes the whole knowledge, $\bar{M}$, the part of the knowledge loaded in the memory after a possible sampling step. $|\cdot|$ denotes cardinality. . . . . 119
4.6 Memory analysis on IBM2015, 1-Topic . . . . . 132

5.1 An excerpt of a DeepProbLog program based on TC-GNNs. . . . . 147
5.2 An excerpt of a GS-MLN program for defining class descriptors dependencies. 148
5.3 Knowledge mapping mechanism via key-based term grounding. . . . . 150
5.4 An example of key-based term substitution. . . . . 151

B.1 Data collection interfaces in ArgSciChat. . . . . 162

D.1 Total legal rationales usage for Arbitration (A). . . . . . . . . . . . . . . . . . 174
D.2 Total legal rationales usage for Unilateral Change (CH). . . . . . . . . . . . . . 174
D.3 Total legal rationales usage for Content Removal (CR). . . . . . . . . . . . . . 175
D.4 Total legal rationales usage for Limitation of Liability (LTD). . . . . . . . . . . 175
D.5 Total legal rationales usage for Unilateral Termination (TER). . . . . . . . . . 176

# List of Tables

2.1 Knowledge types (KTs) in UKI. . . . . . . . . . . . . . . . . . . . . . . . . . 26

3.1 Corpora for argumentative sentence detection. . . . . . . . . . . . . . . . . . . . 57
3.2 Classification performance on IBM2015 corpus. . . . . . . . . . . . . . . . . . . 60
3.3 Classification performance on UKP-Sent corpus. . . . . . . . . . . . . . . . . . 60
3.4 Classification performance on PE corpus . . . . . . . . . . . . . . . . . . . . . . 61
3.5 Classification performance on AbstRCT. . . . . . . . . . . . . . . . . . . . . . . 61
3.6 Training and inference times on the IBM2015 corpus. . . . . . . . . . . . . . . 63
3.7 Constraint satisfaction for claim detection on the IBM2015 corpus. . . . . . . 64
3.8 The set of actions in ArgSciChat. . . . . . . . . . . . . . . . . . . . . . . . . . 69
3.9 Data collection framework statistics concerning completed dialogue sessions. 72
3.10 Dialogue-level statistics of ArgSciChat. . . . . . . . . . . . . . . . . . . . . . 74
3.11 Fifteen most frequent tri-grams question starters in ArgSciChat. . . . . . . . 75
3.12 ArgSciChat statistics regarding fact selection. . . . . . . . . . . . . . . . . . . 75
3.13 Cohen’s kappa computed for each annotator pair on ArgSciChat. . . . . . . . 76
3.14 Action set distribution on ArgSciChat. . . . . . . . . . . . . . . . . . . . . . . 76
3.15 Model performance on Argument Mining tasks. . . . . . . . . . . . . . . . . . 77
3.16 Samples amount for Orig and Split dataset versions of ArgSciChat. . . . . . . 79
3.17 Experimental results and human lower-bound on supporting fact selection. . . 81
3.18 Fact selection performance for LED models trained on the Split dataset according to Best and Union merging strategies. . . . . . . . . . . . . . . . . . . . 82
3.19 Baseline performance for graph argument-based fact selection. . . . . . . . . 82
3.20 Graph statistics for graph argument-based fact selection baselines. . . . . . . 82
3.21 Performance on DE message generation with different input combinations. . . 83
3.22 Sentence-level and message-level performance on DE message generation with different input combinations on the Split dataset. . . . . . . . . . . . . . . . . 83

4.1 Legal rationales for the legal qualification of unfairness (synthetic approach). 95

4.2 Legal rationales for the legal qualification of unfairness (analytical approach). 96
4.3 Results on 10-fold cross-validation on ToS-100. . . . . . . . . . . . . . . . . 102
4.4 ToS-100 corpus statistics. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 107
4.5 Legal rationales for the Arbitration category. . . . . . . . . . . . . . . . . . . 108
4.6 Classification performance according to macro-F1 computed on 10-fold cross-validation for unfair examples on ToS-100. . . . . . . . . . . . . . . . . . . . . 112
4.7 Memory statistics concerning model predictions on unfair examples only. . . 113
4.8 Qualitative evaluation of strong supervision concerning two examples of different unfairness categories. . . . . . . . . . . . . . . . . . . . . . . . . . . . 114
4.9 Corpus statistics for ToS-30. . . . . . . . . . . . . . . . . . . . . . . . . . . 126
4.10 Topics extracted from the IBM2015 corpus, with associated number of evidence and claims. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 127
4.11 Classification macro-F1 computed on 10-fold cross-validation for unfair examples on the ToS-30 dataset. . . . . . . . . . . . . . . . . . . . . . . . . . . 130
4.12 Memory-related metrics concerning predictions on ToS-30 unfair examples only. 131
4.13 Classification performance for dataset IBM2015. . . . . . . . . . . . . . . . 133

C.1 Legal rationales for the legal qualification of Limitation of Liability unfairness. 163
C.2 Legal rationales for the Content Removal category. . . . . . . . . . . . . . . 166
C.3 Legal rationales for the legal qualification of Unilateral Termination unfairness. 168
C.4 Legal rationales for the Unilateral Change category. . . . . . . . . . . . . . . 171

# Chapter 1

# Learning with Structured Knowledge

Knowledge integration into machine learning models has a long history, dating back to the dichotomy of symbolism and connectionism. In recent years, data-driven approaches have become more popular due to the increasing availability of data and computing resources and the practical limitations of symbolic AI, such as the knowledge-acquisition bottleneck [181]. Nonetheless, purely data-driven methods struggle with generalisation when there is no abundance of data. Thus, the need of leveraging prior contextual information was rapidly reformulated as the direct integration of knowledge data into machine learning models [139, 262], hopefully conveying benefits from both approaches and opening a new research area known as *Informed Machine Learning* (IML) [267].

Purely data-driven methods avoid potential conflicts with some basic principles of logic [91], such as inferring general rules from a limited set of examples by providing confidence of inference validity. However, there are still many circumstances where these approaches fail to reach satisfying generalisation capabilities [267]. Intuitively, when limited train data is available, trained models may lack the ability to transcend given information [249, 281]. Other critical factors dictated by the given context, such as fair behaviour, security guidelines and coherence concerning natural laws, may have to be considered. In particular, these factors are mandatory requirements, albeit data availability may not be a significant issue.

To this end, integrating context information as prior knowledge may improve the learning process, both in terms of more efficient use of the available data and a better ability to generalise, address biases and compensate for lack of sufficient training data. Lastly, with more complex deep learning architectures, leveraging well structured and easily interpretable knowledge provides ways to explain and analyse trained models, often depicted as black boxes, by evaluating how this prior information is conveyed along the architecture to produce a prediction [222].

The definition of knowledge is strictly dependent on the given task, even though there may be multiple equivalent formulations. More precisely, we distinguish between sources of knowledge and their subsequent formulation concerning model compliance.

Knowledge can stem from several heterogeneous sources, such as natural or social sciences, individual experts, daily life, etc. From a computer-scientific perspective, knowledge is often considered validated information about relations between entities in specific contexts [267].

The critical step in integrating prior information into machine learning models is defining its representation. Depending on the given model architecture, learning algorithm and task, knowledge integration can take different forms [267]: (i) algebraic equations, (ii) logic rules, (iii) simulation results, (iv) differential equations, (v) knowledge graphs, (vi) probabilistic relations, (vii) symmetries (viii) human feedback. Intuitively, the more knowledge we consider, the more easily we can integrate it into machine learning models.

An additional design choice of informed machine learning is knowledge integration. That is, how to integrate collected prior information into machine learning models. A standard way to inject knowledge is to enrich the training data [16, 107, 120, 182]. Contrary to feature engineering, in which expertise is exploited to create ad-hoc features, knowledge integration is viewed as a separate source of information that is jointly used with original data as if they were a single augmented data set. More advanced techniques view knowledge as part of the model architecture itself, explicitly affecting its representational capability [10]. Famous examples are given by convolutional neural networks (CNNs) and recurrent neural networks (RNNs), where spatial and temporal dimensions are naturally handled as design biases. Another notable example is incorporating logic rules and symbols directly as particular neurons in a neural network model [17, 293].

In addition to prior assumptions concerning data availability and model hypothesis set, knowledge can also be integrated as part of the learning algorithm, typically affecting the original training objective using a proper regularizer [81], and as a benchmark, a tool to evaluate model predictions.

## 1.1 Knowledge Integration in Natural Language Processing

Natural language is the backbone of a large set of information processing applications, whether in textual or speech form. Indeed, language represents one of the significant conveyors of information between individuals [122]. In this context, the development of automated systems able to capture semantically rich notions and infer new knowledge is the core challenge of the AI research area known as *natural language processing and understanding* (NLP/NLU) [44, 110]

Due to the rich, flexible and complex structure of natural language, external knowledge in NLP presents several linguistic facets, mainly concerning syntax, lexicon, salience, sentiment, discourse and semantics. Notable examples of such aspects are word-level properties, such as synonyms, antonyms, homonyms. Usually, we wrap the latter properties large lexical and semantic ontologies, such as WordNet [180], the Penn Treebank [259] and ConceptNet [245]. Other examples are anaphora and pronouns [67, 253], named entities [285] and semantic and constituency parse trees [53, 123].

Natural language inherently follows a sequential flow of data in which words stand as the atomic linguistic units of such structure. Nonetheless, like the above co-reference resolution, we might need to establish linguistic properties involving non-sequential atomic units in many situations.

For instance, semantic parse tree [53] and the well-known knowledge graphs [197] are structured word representations that define semantic relations between linguistic atomic units, e.g. words. The semantic meaning attributed to the relation can span among domain-specific affinities (e.g. words belonging to the same taxonomy), temporal juxtapositions (e.g. words denote events that co-occur), logical entailment of concepts and so on.

Depending on the given abstraction level, nodes in the graph can span from generic words [180] to task-specific entities or concepts [245], such as proper nouns for entity relations, documents [289], debate topics, etc. Alternatively to knowledge graphs, we can also formulate knowledge via logic rules [111, 170]. Additionally, logic statements can also describe domain-specific dynamics that should regulate learning.

Intuitively, in the vast expressive richness of language-based communication, we can convey a concept in an arbitrary number of ways. This property represents one of the critical challenges in NLP. The rich expressiveness of natural language is evident when considering domain-specific scenarios that are subject to tight in-domain regulations concerning language. For instance, texts are characterized by a specific vocabulary of terms in the legal domain and follow a well-known set of stylistic patterns [80]. As an extreme example, efficient deep learning models trained on well-organized texts like journals or newspapers [177] struggle to handle noisy texts full of jargonistic expressions, such as the ones coming from social media [75, 160].

Additionally, it may be not achievable to fully understand abstract concepts that inherently require contextual knowledge by using purely data-driven approaches since the contextual knowledge may not be explicitly covered by the data generating distribution [249].

Therefore, the integration of prior external knowledge can substantially ease the learning process of machine learning models. Moreover, it may also provide improved performance and guarantee desired properties, like model transparency and interpretability [267].

From an application point of view, research domains such as law, biology and medicine, humanities, argumentation, education and so on usually require an extensive amount of contextual knowledge. Additionally, the added information stored in the knowledge may be affected by changes over time. Thus, we generally view knowledge as non-stationary.

In these domains, it may be hard to acquire complete knowledge exclusively from data since the acquisition process is often challenging, time-consuming, expensive, and, in some cases, not feasible. For instance, in topic-oriented online debates, it may be necessary to acquire helpful information about the topic that may not appear explicitly in user interventions [1]. The legal and medical domains represent another example, where an extensive knowledge base is mandatory to address specific case studies [64, 126, 185, 300, 304].

Therefore, providing relevant knowledge and achieving easily interpretable automated models denote major critical requirements of such domains. To this end, the NLP research community spent a great deal of effort to formulate methodologies to acquire different forms of knowledge and adequately integrate such knowledge into autonomous learning models [267].

## 1.2 Structured Knowledge

Depending on the given task, knowledge integration can follow different formulations. However, knowledge representations are often interconnected. For instance, we can express a semantically equivalent formulation of a knowledge graph via logic rules. Vice versa, we can transform a rule set into an equivalent knowledge graph. In this section, we consider significant contributions in informed machine learning in the context of NLP.

### 1.2.1 Knowledge Graphs and Knowledge Bases

The most common approach to knowledge representation in NLP is using knowledge bases, usually structured as graphs, i.e. knowledge graphs. In particular, information like words, generic concepts, expressions, entities and so on is organized into nodes and connected via specific semantic relations. Notable examples of knowledge graphs application include question answering (QA), named entity recognition (NER), text generation, reading comprehension, commonsense reasoning and text classification. For instance, in QA, nodes in the graph are named entities, tied together by domain-specific relations, that help fill missing information that is implicitly required [59, 60, 105, 235, 254]. Similarly, knowledge graphs aid the detection of non-trivial connections between named entities, highlighting implicit word groupings that we can use to achieve improved performance in named entity recognition [30, 49, 55, 260], sentiment analysis [23, 31, 132, 163, 234] and text classification [43, 269, 297, 299]. Notable examples enrich entity information by exploring general-purpose ontologies, such as WordNet [180], FreeBase [24], DBPedia [6] and ConceptNet [245], and unstructured text sources like Wikipedia as external knowledge [55, 260], or other domain-specific ontologies, such as biomedical databases [2].

Furthermore, researchers have adopted knowledge graphs to overcome the limitations of available data when considering challenging reasoning tasks that involve extensive contextual information like commonsense reasoning [249, 250]. In this scenario, the information stored within knowledge graphs is a mandatory requirement to address the task since the underlying decision-making process involves commonsense knowledge that is not available in the data itself.

Alternatively to classification tasks, knowledge graphs have been used extensively for language representation, i.e. word embeddings. In particular, word-level relations can be utilized as augmented contexts in word to vector training [21, 177]. The basic idea is to exploit external information to distinguish and capture the variety of word meanings depending on the given adjacent context while excluding potential ambiguities and covering possible semantic gaps [33, 165, 236, 258, 274, 287]. Notable research proposes learning word representations by accurately masking text portions based on knowledge graph entity-level information to capture word to phrase-level properties [58, 303]. Standard research suggests integrating such knowledge at training data or learning algorithm levels, but recent work also proposes leveraging external information as a benchmark tool [89, 188].

Another important task in natural language processing is text generation, mainly text-to-text generation. Learning to generate appropriate sentences from limited training data is one of the most ambitious goals due to the great variety of possible solutions in natural language. As a result, common knowledge about entity relations is exploited to assess commonsense, i.e. implicit knowledge mainly [37, 99, 306]. Moreover, structured knowledge bases can condition generated dialogues for non-goal oriented tasks in the recommendation and open-ended discussions [179, 306].

From the integration point of view, the primary challenge is to adequately use the informative content stored in knowledge graphs depending on the given domain of interest. For instance, large-scale knowledge bases like ConceptNet [245] are orders of magnitudes larger than individual task-specific input examples, e.g. sentences for text classification. Here, knowledge integration is mainly centred on extracting the small portion of information within the knowledge graph related to the given input example of interest. To this end, several techniques of information retrieval have been proposed [215]. As a further step towards enriching knowledge graphs in the direction of unstructured knowledge representation, Sun et al. [254] offer to augment graph entities with automatically retrieved text descriptions from Wikipedia via traditional information retrieval techniques.

Another challenge in adopting knowledge graphs is integrating the retrieved information within autonomous learning models like deep neural networks. Since knowledge follows a structured form, such as a graph, ad hoc solutions like graph convolutional neural networks [128, 232] have recently being adopted [254] as to account for specific relational inductive biases [10]. We can also use entities in a graph to acquire unlabeled data by considering similarity operations, such as simple word matching, which can be later used in distant supervision to enrich training data [182, 290].

Overall, knowledge graphs are often used for task-oriented information extraction to provide additional valuable information to address the task. Nonetheless, we can also use this knowledge to extend current graphs by leveraging the data-driven approach of autonomous learning models [184].

### 1.2.2 Logic Rules

Alternatively to knowledge graphs, we can formulate prior information as a set of constraints in the form of first-order logic (FOL) rules. Formal restrictions come as a natural representation of task-specific requirements, such as structural conditions for sequential prediction tasks. A notable example in natural language processing is explored by Hu et al. [111] in which named entity recognition and sentiment analysis tasks are conditioned by domain specific constraints in the form of logic rules. More precisely, a neural network model is trained via knowledge distillation [151] by trying to emulate a hard rule-based solver. In named entity recognition logic rule map structural constraints, such as *"if token $i-1$ was tagged as $Y$ then token $i$ cannot be tagged as $X$"*. Likewise, in sentiment analysis, one challenging point of the task is to capture the contrastive sense, e.g., given by the conjunction “but” within a sentence. Despite the intuitiveness of FOL rules and the impressive performance in various tasks, the approach has been limited to simple a priori fixed constraints with manually selected weights. To further integrate rich human knowledge, mutual distillation [113] and domain agnostic [114] solutions have been proposed to learn the confidence values for logical rules.

Similarly to knowledge graphs, researchers have employed logic rules for learning rich language representations by incorporating commonsense knowledge [28, 219, 221, 272]. As an example, Demeester et al. [56] propose to formulate entity information in the form of relational constraints that are totally independent of the number of compatible pairs, allowing scalability to large databases. An interesting yet different task, termed *knowledge graph embedding* (KGE), tackles the problem of encoding entire knowledge graphs into continuous vector spaces in order convey rich external knowledge into machine learning models in an intuitive compliant way without losing structure properties. In this context, logic rules are employed to guide the distillation process so as to consider entity specific relations [100].

### 1.2.3 Linguistic Dependencies

One last broad area of structured text representations is *Linguistic Dependencies*. In particular, we include the set of text representations that capture syntactic and semantic properties of words within a text. Extensive research work has been presented in the last two decades regarding the automatic derivation of these structured text representations. We provide a brief overview, while the reader can find more details in [53, 123]. Notable examples are part-of-speech (POS) tags [29, 87, 205, 212, 233, 261], named entities [41, 138, 227], constituency and dependency parse trees [73, 101, 127, 148, 164] and Abstract Meaning Representation (AMR) graphs [9]. These abstract text representations are generally independent of specific text properties. For instance, different sentences may have the same AMR graph. Additionally, these abstract text representations fall within the terminology of Universal Dependencies (UD) [53] in their broader formulation as they convey information common to many natural languages.

The informative content of described linguistic dependencies structures is of particular relevance for other tasks that remain within the scope of NLP. For instance, constituency and dependency trees have been adopted as an intermediary representation step of unstructured text to address the task of text classification [39, 130, 246]. In this context, tree-oriented kernels have been proposed for text similarity [155, 186]. POS tags have been adopted for sentiment analysis [124, 201], text classification [187] and named entity recognition [145]. Moreover, AMR graphs have been proposed for argument mining [198] and, most notably, text generation for [18, 50, 244, 271].

## 1.3 Challenges

Integrating structured knowledge into machine learning models can bring several benefits compared with purely data-driven methods. Notable examples are improved performance, increased model transparency and data specific biases compensation. Nonetheless, in real-life applications, the available knowledge is often too generic, thus, lacking domain-specific information. To this end, more precise knowledge bases have to be defined. For instance, general-purpose knowledge graphs like ConceptNet [245] may not report adequate terminology or relations for applications in the healthcare domain. To this end, domain-specific knowledge bases that can relate, for instance, symptoms with diseases in a practical way for the expert user have to be arranged [162].

However, properly defining domain-specific knowledge may not be feasible due to the lack of data or being too costly to obtain. As a notable example, in the legal domain, the motivations explaining why specific juridical actions have been taken may be articulated over multiple documents and contain several cross-references to legal laws. Here, correctly highlighting reasons of action may be prohibitively time-consuming, if possible at all. The difficulty is mainly attributed to how these motivations are deeply entangled within domain-specific linguistic constructs and stylistic patterns. Therefore, manually extracting structured knowledge, e.g. in the form of logic rules, may be quite challenging in the above scenario.

Another example concerns online debates in which high-order contextual information, such as cross-topic relations, is crucial even for trivial actions like inferring rebuttal or support relations between user interventions. The initial assumption of having a good and complete knowledge base at own disposal may not always be corroborated. In particular, we may not entirely represent such information via a knowledge graph and, therefore, must be extracted from given text sources.

Lastly, the automatic extraction of structured knowledge inherently requires a corresponding model. The last step, in turn, requires ad-hoc data for training the model itself. Generally speaking, tools for the automatic extraction of structured knowledge are inherent restrictions for its integration.

## 1.4 Contributions and Outline

We have discussed how the integration of structured knowledge can hide critical bottlenecks in specific domains. Under the introduced scope of knowledge integration in NLP, we discuss the implications of directly using unstructured text information as a form of knowledge. The natural language is one of the significant communication devices of humans for information exchange. A good amount of information is often shared in natural language in various domains.

Behind this motivation, we discuss the concepts of unstructured knowledge retrieval, integration and reasoning for machine learning models to pave the way to automated models able to leverage vast amounts of unrefined knowledge. In particular, we focus on natural language processing tasks, where reasoning and semantic understanding of relevant textual cues is of critical importance.

We identify the following research questions:

**Q1** *How can unstructured textual knowledge be integrated into machine learning models?*

**Q2** *How are different types or formulations of unstructured textual knowledge related? Is it possible to define a taxonomy?*

**Q3** *Can we define criteria to evaluate the process of unstructured knowledge integration?*

**Q4** *Which challenges does unstructured knowledge integration introduce compared to structured knowledge integration?*

**Q5** *When should we consider adopting unstructured knowledge integration rather than structured knowledge integration?*

**Q6** *To what extent does unstructured knowledge integration relate to existing research domains requiring knowledge-intensive reasoning capabilities?*

**Q7** *Can we adopt structured knowledge integration to address some of the challenges of unstructured knowledge integration?*

In Chapter 2, we introduce Unstructured Knowledge Integration (UKI) as the process of integrating unstructured knowledge into machine learning models. More precisely, we discuss UKI in the field of NLP, where we represent unstructured knowledge by information expressed in natural language, e.g. text documents.

We discuss possible shortcomings of structured knowledge and investigate the implications of directly adopting text as a form of knowledge (**Q5**). In particular, we describe the sub-processes that define UKI (**Q1**): (i) the process of knowledge integration, (ii) the process of knowledge mapping and (iii) the process of knowledge validation. The first covers the issue of defining a knowledge integration mechanism that is compliant with the expressed semantic of unstructured text knowledge. For instance, when textual patterns represent knowledge are integrated via comparison in pattern matching. Second, knowledge mapping defines how to implement the knowledge integration mechanism. Third, the process of knowledge validation describes the demanding requirements of the knowledge integration and mapping processes that have to be satisfied (**Q3**).

Furthermore, we identify several text knowledge types (KTs) depending on the semantic meaning attributed to the knowledge (**Q2**). Each KT defines its own set of UKI sub-processes and demanding requirements (**Q4**). For instance, the above example of textual patterns is a KT. Another example is given by textual constraints or class rationales.

Within the perspective of knowledge integration, we identify three primary roles of UKI: (i) source of complementary information, (ii) support for reasoning and (iii) behavioural. This distinction reflects the taxonomy for knowledge integration introduced in [267] and discussed at the beginning of this Chapter.

We provide a detailed list of broad research areas that show affinities with UKI like reading comprehension, few- and zero-shot learning and knowledge-enhanced text generation. We view some of these tasks as instances of UKI and remark the unique characteristics of UKI (**Q6**).

We discuss UKI for structured knowledge extraction (KE) (**Q7**). In particular, in Chapter 3, we identify KE as a sub-process of UKI as it can provide abstract text representations that satisfy specific knowledge integration requirements based on the given KT. We present a few experimental settings where a KE process can benefit the decision-making process of a model. We conclude by remarking on the challenges and strong assumptions of structured knowledge extraction and integration.

As a follow-up, in Chapter 4, we present a few experimental text classification scenarios where the definition of a KE process is particularly challenging. More precisely, we can exploit no available knowledge relevant to the task. To this end, we formulate each problem under the framework of UKI by leveraging unstructured textual knowledge (**Q5**). We identify the major challenges of adopting UKI in these domains in addition to dealing with natural language, such as poor knowledge quality, supervised and partial knowledge mapping, and scalability (**Q4**). We explore simple methodologies for integrating unstructured textual knowledge into deep learning models. Nonetheless, albeit not guaranteeing specific UKI properties, the proposed methods provide relevant benefits in addressing the tasks and highlight new challenges.

Finally, Chapter 5 provide a unified view of KE and UKI under the general scope of combining symbolic and sub-symbolic representations (**Q7**). To this end, we discuss the adoption of Neural-Symbolic Learning and Reasoning (NeSy) as a valuable tool to integrate KE as a sub-process of UKI to guarantee certain challenging requirements specific to given KTs.

# Chapter 2

# Background

Structured knowledge is a valuable additional source of information to regulate the learning process of autonomous learning models. Nonetheless, in many scenarios, such information may not be available or hard to obtain. Therefore, we discuss the process of directly using unstructured text as a form of knowledge due to the sheer amount of this type of information in different domains.

In this chapter, we introduce Unstructured Knowledge Integration (UKI) as the process of integrating unstructured text knowledge into machine learning models. In Section 2.1, we elaborate UKI from the available structured knowledge extraction and integration processes. Then, in Section 2.3, we describe the sub-processes regulating UKI. We provide some remarks about integrating textual knowledge (Section 2.4) and the role of knowledge (Section 2.5). In Section 2.6, we categorize different textual knowledge representations into knowledge types. Lastly, in Section 2.8), we compare UKI with popular research areas in NLP.

## 2.1 A Simple Interpretation of Knowledge Integration

To understand why structured knowledge is helpful, we have to widen up the current point of view. In particular, we momentarily abandon the problem how to integrate structured knowledge and concentrate on why it is the correct process to adopt. As mentioned in Chapter 1, we can integrate knowledge into machine learning models under multiple paradigms, such as data augmentation, architectural biases and additive learning guidance. All these integration paradigms are dependent on the structure of the available data. In other terms, we would say that we can derive structured knowledge from a particular source of data via a specific mapping function.

$$z = g(x_z) \tag{2.1}$$

where $z$ is the extracted structured knowledge, $g$ is the mapping function, and $x_z$ is the original data from which knowledge has been extracted. For instance, $z$ can be represented by a knowledge graph about researchers of a scientific community, while $x_z$ can be their set of publications and corresponding metadata. With a slight abuse of notation, $z$ can also represent a particular architectural bias, e.g. sequential dependencies with time-series data, task-specific learning objectives or an identity function in case of data augmentation. Thus, we can generally denote $z$ as the result of the problem-solving approach that starts from a given problem and ends up producing a suitable solution.

Once extracted, the auxiliary source of information represented by $z$ is integrated into the learning model. We can then view the learning phase as minimizing an objective function that relates input data to corresponding model predictions. However, we now condition the learning phase on $z$. In the traditional setting of supervised learning, formulated as a minimization problem, we have:

$$\min_{\theta} \mathscr{L}(y, f_\theta(x)|z) \tag{2.2}$$

where $f_\theta$ is the learning model with $\theta$ as the parameter set. We distinguish between $x$ and $x_z$ since we can derive $z$ from other data sets. For instance, a large-scale knowledge base like WordNet [180] has been built upon several sources of information. In this case, $z$ is the knowledge base, and the data from which $z$ has been defined is $x_z$. We can then employ $z$ to address a particular task, e.g. text classification, on a domain-specific set of data $x$. Alternatively, $x$ can be equivalent to $x_z$, i.e. $x \equiv x_z$ is the knowledge $z$ is extracted from $x$. An example is given by constituency and dependency parse trees introduced in Section 1.2.

Structured knowledge integration into a machine learning model, i.e. the integration of $z$ to condition $f_\theta(x)$, is mainly centred on defining an adequate $f_\theta$ to deal with $x$ and $z$ jointly. In particular, an appropriate $f_\theta$ for integrating $z$ has to be determined to condition the learning process. For instance, if $z$ is a knowledge graph, the topology of the graph structure could be directly leveraged by $f_\theta$ via Graph Neural Networks (GNNs) [128, 229]. This reasoning is mainly motivated by the fact that a few $z$ formulations (knowledge graphs, logic rules, etc...) should be considered to guarantee broad applicability. For instance, AMR graphs are widely applied in NLP [18, 50, 244, 271], since such an abstract text representation captures semantic dependencies that act as the building block of many task-specific reasoning processes on text.

Keeping in mind well-known challenges regarding structured knowledge integration (Section 1.3), we investigate whether we should bridge the gap between $f$ and $g$ in the opposite direction, that is, from $g$ to $f$. More precisely, the natural language representations of $x$ and $x_z$ already store the information required to solve a particular problem. Indeed, albeit lossy of information, natural language stands as one of the backbones of human communication, along with speech and non-verbal communication [122]. Indeed, we could formulate instructions, rules or constraints and data properties using natural language.

We focus on the problem of directly integrating such a natural language knowledge $x_z$ into a learning model, denoted as *unstructured knowledge*. For further clearance, we expand the set of knowledge representations to plain textual knowledge. For instance, as with zero-shot learning [38], a group of textual class descriptors can denote the knowledge to be integrated. In this scenario, we can cast down the knowledge integration process to a textual comparison operation. The aim is to aid the classification model by providing information about the similarity between input data and the class descriptors.

Similarly, a rule or constraint set can act as integrated knowledge. Unlike the previous example, the desired usage of knowledge enforces ad-hoc integration processes that convey hard requirements like model interpretability, transparency and consistency. In other terms, we have to integrate textual constraints into the learning model such that their satisfaction is guaranteed and that the latter can be verified.

## 2.2 Unstructured Knowledge Integration

*Unstructured Knowledge Integration* (UKI) is the process of integrating unrefined textual knowledge into machine learning models. We mainly refer to deep learning models due to their current vast adoption [256]. The term unrefined indicates the absence of a particular semantic frame and structure associated with the knowledge itself. For instance, a knowledge graph is a structure that inherently defines how data is organized, i.e. the graph topology, how data is interpreted, i.e. the relation semantic, and the reasoning process, i.e. by navigating the graph. The graph topology denotes the knowledge structure, while the interpretation and reasoning define its semantic frame. In contrast, in UKI, we do not primarily enforce a particular semantic frame and a structure. Instead, we focus on determining the desired knowledge interpretation and its possible integration formulation.

The property of defining knowledge in plain natural language opens up to a wide variety of unstructured text knowledge representations. Similarly to structured knowledge, semantically equivalent unstructured text knowledge representations exist.

Indeed, we can express the same concept in natural language in several ways. Nonetheless, this semantic equivalence might not be reflected from the learning point of view. In particular, some unstructured text knowledge representations may be more expressive and more easily integrated into deep learning models. For instance, a specific unstructured text knowledge representation can contain far too abstract concepts that are loosely related to input data. In this case, a knowledge integration implementation based on embedding semantic similarity, i.e.

both knowledge and input data are embedded by the model and then compared, may not be as effective as desired.

We can consider other types of unstructured knowledge. For instance, images can convey enough informative content to express a particular concept into a deep learning model. Similarly, sound and voice signals can take the same role as textual knowledge. We focus on UKI in the context of NLP as it provides a wide variety of applications and several knowledge representations. Indeed, a good amount of unstructured text information is often shared in a wide variety of domains.

Generally speaking, the same abstract concepts and goals, which reflect the same human reasoning process, can be expressed under different representations. For example, we can think of enforcing a particular rule regarding road stop signals into an autonomous driving system. Such a rule can be expressed via an image or textual description. Still, the conveyed message is the same. Depending on the given model architecture, a particular knowledge representation is more suitable than the other.

The significant advantage of UKI concerns its ambition of exploiting unstructured text information as integrated knowledge. As described in Section 1.2, structured knowledge in NLP is tied to hard requirements, such as adequate data availability. Still, within the domain of NLP, the abundance of unstructured textual knowledge is a property of many application domains. For instance, sizeable textual knowledge bases like Wikipedia store a vast amount of information that we can exploit. We motivate the need of investigating UKI as it leverages a vast amount of unstructured data that is still challenging to integrate into deep learning models. We discuss more in detail the advantages of UKI in Section 2.7.

The ambition of integrating unstructured text knowledge notwithstanding, adequate integration processes have to be defined to use such knowledge properly. In Section 2.3, we define UKI as the composition of three distinct sub-processes.

## 2.3 Unstructured Knowledge Integration Processes

We define UKI as a system comprised of distinct sub-processes. The first is the process of knowledge integration, which defines the integration mechanism by which we should employ unstructured text knowledge. The second is the process of knowledge mapping, which describes how knowledge, or a portion of it, should be linked to input data. The third is the process of knowledge validation, which verifies if the adopted knowledge integration and mapping processes do not violate knowledge specific requirements. In this section, we discuss UKI sub-processes in detail. Depending on the given knowledge type (KT), we have to consider

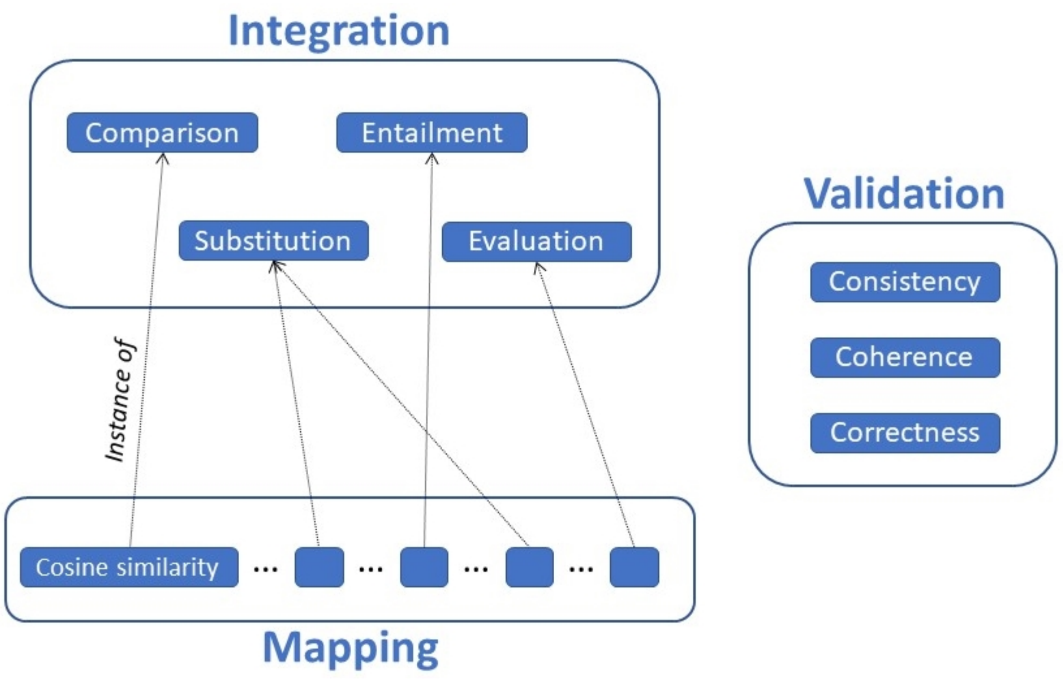


Figure 2.1: UKI sub-processes. We define four unstructured knowledge integration mechanisms. A knowledge mapping mechanism is an instance of each knowledge integration mechanism. Lastly, knowledge validation defines three major requirements concerning the integration of unstructured knowledge.

different UKI sub-processes as the KT can enforce a set of ad-hoc requirements. Figure 2.1 provides a summary schema of UKI sub-processes.

### 2.3.1 Knowledge Integration

Textual knowledge can represent different concepts. Some examples are additional data, data properties, learning constraints and task-specific descriptors. Then, we have to integrate unstructured text knowledge by its conveyed concept. For instance, knowledge in the form of additional and contextual data acts as a reference point of comparison: the learning model should use such unstructured knowledge by comparing it with given input data to identify some relevant task-specific properties. We describe the process of knowledge integration as the definition of an integration mechanism, e.g. comparison in the above example, that is aligned with the desired interpretation and usage of unstructured text knowledge.

Note that the unstructured knowledge integration process is only limited to defining how to employ knowledge. It does not provide any particular detail about how we should do such integration. We address the latter by the knowledge mapping process depending on the given knowledge type (Section 2.6). Similarly, we identify different unstructured knowledge integration mechanisms depending on the given knowledge type.

### Comparison

Intuitively, we base the knowledge integration mechanism of comparison on the knowledge representing a reference point for input data. Such a reference point may refer to properties within each text input. Alternatively, it can mean additional information somehow correlated to the text input.

The comparison mechanism matches common properties in both the unstructured knowledge and the input data. Thus, the knowledge contains information that partly or entirely refers to the properties of each text input. For instance, suppose the knowledge includes a list of textual expressions or patterns like word sequences or linguistic templates. In this scenario, the knowledge explicitly describes the syntactic, lexical and semantic properties of input data.

### Entailment

Differently from comparison, the entailment knowledge integration mechanism covers a broader meaning. Generally speaking, we use the term entailment to express the concept of logical implication between a source and a target, like in NLI [250]. The integrated unstructured knowledge can also provide details concerning the decision-making process of a model. A notable example is when the knowledge encodes explanations of each possible model action. In particular, consider the context of text classification, where the model also explains each of its predictions. In this case, the explanation stored in the unstructured knowledge acts as an indicator of the model's decision process. The output of the decision-making process of a model is connected to the knowledge via entailment. Entailment can also be distantly related to the final prediction of a model. For instance, suppose the unstructured knowledge strongly correlates with task-specific input data. The knowledge is associated with its corresponding set of input texts can be defined via entailment if it reflects the correlation between the knowledge and the input data.

### Substitution

Substitution is the most straightforward integration mechanism as it simply describes a modification of input data by providing new additional information. For instance, contextual information or text input-specific attributes, such as POS tags and NER entities, are integrated by enriching the given text input with the provided information. We view this enrichment process as substituting the original input with its modified counterpart.

#### Evaluation

We could also employ unstructured knowledge as a metric to evaluate the satisfaction of certain desired behaviours. As a concrete example, we define knowledge as a set of constraints that have to be enforced in the decision-making process of a model. Note that these constraints are not specific to a single input nor stand as a reference point for comparison. Textual constraints describe behaviours that we require to be enforced in the model.

### 2.3.2 Knowledge Mapping

The second UKI sub-process is knowledge mapping. Generally speaking, knowledge mapping can be viewed as the concrete realization of the more abstract knowledge integration process. Once a knowledge integration mechanism has been identified, knowledge has to be mapped to the provided input data via the model for knowledge integration to take place. In particular, we require a knowledge mapping mechanism responsible for retrieving the portion of unstructured text knowledge specific to one input example. To this end, the knowledge mapping mechanism should be aligned with the desired interpretation of unstructured text knowledge.

In specific scenarios, knowledge mapping is trivial since no information retrieval phase for extracting relevant knowledge for the given input data is required. For instance, we define the unstructured knowledge as input specific text attributes, like word aspects as with Sentiment Analysis [163]. In this case, we integrate unstructured knowledge as an additional property of the input text, i.e. via the integration mechanism of substitution. Thus, no explicit knowledge mapping mechanism is required.

### 2.3.3 Knowledge Validation

Knowledge integration and mapping processes have to be adequately validated concerning unstructured knowledge. Thus, we require the definition of a set of mechanisms that assess knowledge requirements. For instance, we could integrate text constraints like additional data via comparison. However, such an approach misuses unstructured knowledge as there is no alignment between its interpretation and integration.

We identify three primary requirements concerning unstructured knowledge integration: (a) it should preserve consistency throughout the integration process (**consistency**); (b) it should be integrated coherently with respect to its desired interpretation (**coherence**); (c) it should be mapped correctly to input data (**correctness**).

### Consistency

Before considering integrating unstructured knowledge into a model, we need to verify whether the information stored in the knowledge somehow correlates with the given task domain. For instance, suppose that knowledge is defined as a set of textual patterns. These patterns should describe expressions that we can observe in input data. Otherwise, the integration of unstructured knowledge is ill-posed and does not benefit. More generally, consistency denotes whether the integrated unstructured knowledge is adequate and aligned with the task of interest and the corresponding input data.

Despite this relatively simple example, knowledge consistency is particularly crucial for certain KTs as their integration can be viewed as a support for the decision-making process of a model or as a desired behaviour. Additionally, the integration mechanism might also have side effects like modifying the knowledge itself (Section 2.4). In this scenario, a UKI system is mandatory to guarantee consistency.

### Coherence

Even if the knowledge provides content consistent with the given task domain, it is still necessary to validate its correct usage throughout the knowledge integration process. Such property is denoted as *coherence*. We can view coherence as the main property to be validated concerning unstructured knowledge integration. In particular, the integration process should not violate the desired interpretation of unstructured knowledge.

We must enforce these requirements in both knowledge integration and knowledge mapping processes. In the first, coherence is expressed as choosing an appropriate integration mechanism. In the second, the knowledge mapping mechanism must be consistent with the employed knowledge integration process.

Coherence satisfaction can be trivial or particularly challenging depending on the given KT since sophisticated knowledge mapping, and integration mechanisms are required. More details are provided when describing KTs in Section 2.6.

### Correctness

Similarly to coherence, we also require certain KTs to have a knowledge mapping mechanism capable of correctly associating unstructured knowledge to input data We denote this requirement as the *correctness* property. As notable examples of KT where correctness represents an explicit requirement, we consider textual class descriptors and class rationales. Textual class descriptors replace class labels in a classification setting, where textual class rationales provide explanations regarding each model prediction. In both cases, the unstructured knowledge contains information that directly regards the decision-making process of a model. Thus, the knowledge mapping component must define the right association between the unstructured knowledge and the input data.

## 2.4 Unstructured Knowledge Integration Properties

In addition to knowledge validation, we also denote a set of ’secondary’ properties and dynamics that characterize a UKI system. In particular, these properties are secondary, as we can view them as sub-parts of the requirements listed for knowledge validation (Section 2.3.3). Nonetheless, these properties describe behaviours usually formulated as mandatory requirements to satisfy a given problem setup. Thus, their importance is not secondary, but rather complementary to knowledge validation.

### 2.4.1 Knowledge Update

The integrated unstructured knowledge can be subject to modifications or updates and, thus, generally considered dynamic rather than static. We distinguish between two macro scenarios where we can modify knowledge.

In the first scenario, knowledge modification takes place at the knowledge integration level. The unstructured knowledge integration and mapping processes have the side effect of updating the knowledge. Such an update is generally data-driven [114, 282] since the primary intent of knowledge modification is to ease its integration based on observed input data.

In this case, the entity of the modification is generally considered of minor impact as the underlying meaning of the knowledge remains almost unaltered. For instance, suppose the unstructured knowledge is defined as a set of textual patterns. In this case, the knowledge modification process alters those textual patterns with slight linguistic inconsistencies with input data. By doing so, the degree of matching textual patterns increases as the unstructured knowledge adapts to observed data. In this context, the only constraint regarding knowledge is preserving its semantic meaning.

In the second scenario, the domain of interest or the task itself is subject to specific changes. That is, the underlying problem setup changes based on certain time-dependent factors. Consequently, the knowledge becomes invalid, and we can no longer integrate it.

For instance, consider the automatic judgment prediction settings in the legal domain. In this scenario, the task is to emulate the judge’s decision-making process based on a set of given documents. In particular, the judge has to decide whether the attorney’s request should be accepted or rejected.

The model can support its prediction on a set of legal laws to do so. The latter group of laws constitutes the knowledge for the given problem. Here, some laws may be changed or are not valid anymore. Such phenomena are related to the problem of guaranteeing the *consistency* property of knowledge validation. Thus, the nature of the unstructured knowledge update is implicitly tied to the given task.

### 2.4.2 Integration Efficiency

We should also evaluate the knowledge integration process based on the risk of becoming a bottleneck in the model decision-making process. We denote as integration efficiency the set of capabilities that allow: (i) ease of knowledge integration into a model (**ease of integration**); (ii) relevant task-specific performance improvements (**integration performance**); (iii) scalability from small to large-sized knowledge (**scalability**).

#### Ease of Integration

The ease of integrating unstructured knowledge refers to the applicability of the knowledge integration processes. For instance, we might articulate the chosen unstructured knowledge mapping mechanism into multiple complex steps. Additionally, it might be based on some assumptions regarding input data. Consider textual patterns as unstructured knowledge. The devised knowledge mapping mechanism evaluates each pair of (textual pattern, input text) for comparison. Such comparison is made at the word level by evaluating individual word groups based on shared named entities. For instance, all nouns in the textual patterns are compared with those in the input text. In this example, the unstructured knowledge mapping mechanism is articulated into multiple operations based on the assumption that certain named entity groups are observed.

Generally speaking, we require a measure of computational efficiency that assesses the applicability of the unstructured knowledge integration process. This requirement is particularly relevant when considering KTs requiring sophisticated knowledge integration processes to satisfy their underlying requirements.

#### Integration Performance

The unstructured knowledge integration process also measures the degree of improvement regarding task-specific model performance. Model performance is one of the main evaluation methods of a UKI system since task-specific evaluation criteria are usually provided. In other terms, we rely on the given problem setup to indirectly measure the quality of a UKI system. In the context of deep learning models, we must calculate the property of integration performance explicitly to determine the degree of impact of unstructured knowledge properly. For instance, if unstructured knowledge is integrated via attention-based mechanisms [77], there might be the risk that the model is not exploiting knowledge as intended or, worst, is ignoring it.

**Scalability**

Integration scalability refers to the capability of scaling unstructured knowledge integration processes to large scale unstructured knowledge sets. Integration scalability also covers potential issues regarding each knowledge integration process when varying the size of the knowledge. For instance, the unstructured knowledge mapping process might bring different results based on available knowledge. In deep learning, a notable example is given by attention-based knowledge mapping mechanisms [92, 252, 266, 276]. In this case, the attention mechanism is employed to extract information from the knowledge proportionally to the obtained attention values. Depending on the knowledge size, the distribution derived from attention values can be far too flat. In particular, most knowledge elements are attributed to a low distribution mass. Thus, the described knowledge mapping mechanism fails to map knowledge to input data correctly.

### 2.4.3 Integration Transparency

The knowledge integration process might also be subject to high-level requirements such as model interpretability or model reasoning transparency. In other terms, the decision-making process of a model should be easily understandable when an interpretation semantic is provided.

Regarding unstructured knowledge, such requirements are mainly translated to explicitly motivate the choices of the knowledge mapping mechanism when linking an input to a specific knowledge subset. Depending on the given knowledge type (KT), the integration transparency property can be trivial or assume crucial importance. For instance, linguistic, textual patterns are trivially motivated by how the knowledge mapping mechanism is defined. The interpretation semantic is defined as the degree of matching between textual patterns and given input data.

On the other hand, transparently integrating textual class rationales may require a more sophisticated knowledge mapping mechanism that may rely on structured text representations (Section 1.2) as they provide adequate interpretation semantics.

### 2.4.4 Integration Robustness

We can formulate unstructured textual knowledge in different ways via natural language. We then require a knowledge integration process that is robust to equivalent textual representations.

For instance, two textual patterns could differentiate by using synonyms or by following a different syntactic structure. In this case, the unstructured knowledge integration process should not be too much affected by linguistic differences.

Generally speaking, a UKI system robust to knowledge representation formulations is a valuable resource in a wide variety of application domains. It is not particularly necessary to spend too much effort on knowledge representation in such scenarios. Lastly, other UKI properties like unstructured knowledge update (Section 2.4.1) are strongly tied to integration robustness. For instance, we can guarantee robustness by defining a UKI system that adapts to the data-driven learning approach of a model.

## 2.5 The Role of Knowledge

The knowledge integration process is tightly related to the role attributed to the knowledge. Regarding UKI, we define three significant roles for unstructured knowledge: (i) knowledge as a source of additional information that is complementary to input data; (ii) knowledge as a support device for reasoning; (iii) knowledge as behaviour that conditions the decision making process of a model. We discuss each knowledge role in more detail in the following sections. Figure 2.2 shows the described knowledge roles. In particular, we report the knowledge types (KTs) that follow a specific knowledge role. We will discuss them in detail in Section 2.6. Lastly, we depict knowledge roles in the general data-driven learning approach of deep learning models.

### 2.5.1 Source of Complementary Information

In this role, unstructured knowledge enriches the informative content of input data. In particular, the integrated unstructured knowledge provides the following benefits: (i) it represents additional data that a model can leverage; (ii) it compensates for potential missing information in input data that is valuable for a specific task; (iii) it counterbalances potential biases in input data. Thus, this role of knowledge is mainly centred on providing an enriched input representation that is specific to the task of interest. Other research areas like data-augmentation share some of these properties [72].

### 2.5.2 Support for Reasoning

Unstructured knowledge is also a valuable support device for reasoning. Textual patterns give a notable example. In this case, textual patterns as the unstructured knowledge encode partial or complete prior information about the task. By relying on such knowledge, the model has

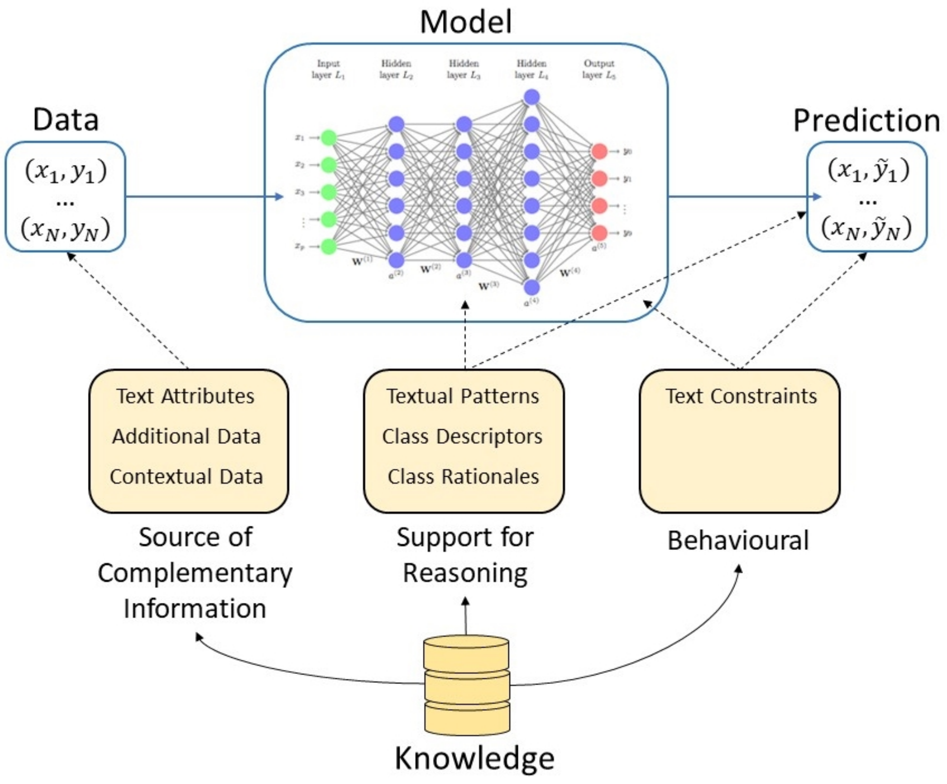


Figure 2.2: The roles of knowledge depicted within the general data-driven learning approach of deep learning models. Knowledge roles are reported based on the knowledge types (KTs) that support them.

a term of comparison to regulate its decision-making process. In the case of textual patterns, we analyse input data through the lens of these patterns: the model takes a determined action based on the matching between a textual pattern and the given input text.

Slightly different use of unstructured knowledge is defined when considering textual class rationales. Here, the informative content provided by the knowledge provides a means of interpretation of the decision making process of a model, rather than solely supporting it. By referring to such knowledge and its integration process, we can also correctly analyse the behaviour of a model.

### 2.5.3 Behavioural

Lastly, unstructured knowledge can also explicitly describe a specific behaviour a model should follow. In this case, knowledge imposes an explicit modification of the decision-making process of a model. Text constraints give a notable example. These constraints encode desired properties that affect the output of an intermediate section of a model. In other terms, unstructured knowledge is a set of instructions that add up to the ones that implicitly describe the task of interest.

## 2.6 Unstructured Knowledge Types

The flexibility of natural language for expressing knowledge in NLP allows the definition of various concepts, such as textual patterns, textual constraints and task descriptors. We denote each of these concepts as a *Knowledge Type* (KT). KTs do not differentiate in their representation as they all share the exact natural language representation. Instead, the point of distinction is their intended interpretation and, consequently, integration into a deep learning model. For instance, a set of additional data to be integrated with data-augmentation [72] is one of the KT we distinguish. Task-specific textual constraints give another KT. Different from the first example, this KT's integration process introduces additional requirements, such as the validation and measurement of the degree of satisfiability of those constraints. Table 2.1 reports a summary of identified KTs.

Table 2.1: Knowledge types (KTs) in UKI. We report the knowledge integration mechanism and some examples of knowledge mapping mechanisms for each KT. Regarding knowledge validation, we identify the priority of each UKI property to guarantee. Lastly, we provide some examples.

| Type | Integration | Mapping | Consistency | Coherence | Correctness | Example(s) |
|---|---|---|---|---|---|---|
| Textual Patterns | comparison | [213, 276] | Secondary | Primary | Secondary | "In my opinion"<br>"A but B" |
| Text Attributes | substitution | [112] | Primary | Secondary | Secondary | sentiment aspects,<br>POS tags,<br>stance |
| Class Descriptors | entailment | [4, 250] | Primary | Primary | Primary | human guidelines |
| Class Rationales | entailment | [32] | Primary | Primary | Primary | legal explanations |
| Additional Data | substitution | [90] | Primary | Secondary | Secondary | unsupervised examples |
| Contextual Data | substitution | [250] | Primary | Primary | Secondary | sentence neighbourhood,<br>topic description |
| Text Constraints | evaluation | [7, 104, 305] | Primary | Primary | Primary | user profiles,<br>game instructions |

### 2.6.1 Textual Patterns

Textual patterns are the first KT that we introduce. In particular, we consider the set of syntactic, lexical and semantic characteristics that all together can distinguish specific texts from others. As a toy example, we can assume to mainly detect opinions by textual patterns like *"I think that"* and *"In my opinion"*. These patterns constitute a form of domain-specific knowledge that often remains implicit in provided data. Additionally, textual patterns are a form of regulariser by guiding the model towards recognising valuable features. The integration of textual patterns is a well-known challenge in NLP, and extensive work has been proposed to make use of such domain knowledge. Notable examples are frameworks for their integration via soft logic rules [111, 114] and using knowledge graphs [168]. In the context of UKI, we define the requirements for the integration of textual patterns.

### Integration

We can view the integration of textual patterns as a pattern matching process. These patterns reflect some characteristics of available data that we can observe. For instance, we can detect textual patterns in small text segments in the exact form in the available data. We could then consider each textual pattern individually and verify if each input text contains that textual pattern. On the other hand, we can consider more abstract textual patterns. For instance, in the above example regarding opinion detection, we can define the following abstract textual pattern: *"A subjective expression introduces an opinion"*. In this case, the textual pattern describes a blueprint of which textual patterns like *"I think that"* and *"In my opinion"* are instances. In both cases, the unstructured knowledge integration mechanism for textual patterns is the one of *comparison*. The textual pattern encodes a linguistic property of which some given text inputs may contain an instance. We then view textual patterns as an additional input that we employ for comparison.

### Mapping

In the context of deep learning, the unstructured knowledge mapping mechanism is usually a similarity operation in the embedding space. In particular, we compare text inputs and textual patterns via vector-based similarity in a shared embedding space function like cosine similarity [213, 276]. In this case, sophisticated text embedding methods capture linguistic properties concerning syntax and semantics [58, 264]. A measure of knowledge mapping is then given by the similarity score between each (input text, textual pattern) pair. Note that we can consider other unstructured knowledge mapping mechanisms. For instance, the embedding-based similarity operation can only represent a portion of a more broad knowledge mapping mechanism that employs other comparison operators altogether.

### Validation

Regarding *consistency*, the role of textual patterns is generally not disruptive. The most common knowledge integration issue is an incomplete set of textual patterns. Thus, their integration into a deep learning model may not provide significant benefits. It is generally prohibitive to provide a set of textual patterns that fully describe a domain's properties. Usually, it is sufficient to provide a set of textual patterns to capture valuable properties of input data. For these reasons, we can usually guarantee consistency in the early stages of the problem setup, where textual patterns are defined. In particular, we define a static set of textual patterns and use it throughout the unstructured knowledge integration process. However, if we allow dynamic modifications of the knowledge (Section 2.4.1), it is necessary to evaluate the nature of such modifications. For instance, unwanted text modifications could lead to harmful text phenomena like biases, textual noise or irregular textual patterns. Concerning *coherence*, we must verify if the adopted unstructured knowledge mapping mechanism reflects the principle of pattern matching. For instance, a semantic similarity approach between (input text, textual pattern) pairs based on the cosine similarity may violate the coherence property. In this case, we define the knowledge mapping mechanism based on text similarity. However, a textual pattern might represent a small portion of the input text, i.e. a text segment. Thus, a text similarity comparison operator may deviate from the principle of pattern matching.

We can rely on the similarity operator of the unstructured knowledge mapping mechanism to evaluate *correctness*. Additionally, if we provide direct supervision between textual patterns and input text, we can then measure the degree of satisfiability of the *correctness* property. Nonetheless, we usually define textual patterns based on a preliminary experts analysis. Thus, textual patterns are hardly manually associated with each input text as they are not generally feasible. Therefore the *correctness* property for textual patterns integration is somewhat secondary compared to the *consistency* and *coherence* properties. In particular, we generally integrate textual patterns to aid a model in achieving higher performance in solving the task. In other terms, the role of knowledge is the *source of complementary information* (Section 2.5.1). Task-specific performance measures are usually sufficient to evaluate the successful integration of textual patterns. In contrast, knowledge validation assumes a primary role when considering the problem of extracting textual patterns from data to employ them as a support device for a deep learning model. In this case, knowledge validation is a critical component for applying extracted textual patterns in other domains.

### 2.6.2 Text Attributes

Text attributes represent another KT. Text attributes encode textual properties that are specific to given text input. In particular, text attributes expand the informative content encoded by their corresponding text input. Examples of text attributes are sentiment aspects[112], part-of-speech (POS) tags, stance towards a target reference, named entities and text intent. Text attributes represent a form of text enrichment. In particular, we view the input text as a potential feature set fed as input to a given deep learning model. Then, the feature set of text attributes enriches the input one.

#### Integration

We can consider text attributes as an additional model input since they are input-specific. Regarding UKI, the knowledge is like a lookup table in which there's a one to one mapping between inputs and knowledge content. Regarding the unstructured knowledge integration mechanism, an integration process is a form of input replacement: text attributes modify or enrich the original input text with their information. Unlike other KTs, text attributes only encode additional information regarding the input text. Therefore, their integration is limited to text enrichment. Thus, the integration mechanism is the one of *substitution* (Section 2.3.1).

#### Mapping

Text attributes do not require a knowledge mapping mechanism as they can be considered an additional input to the model. Alternatively, we can view text attributes as a lookup table used for each text input to retrieve the corresponding set of attributes.

#### Validation

Given the simplicity of the unstructured knowledge integration process, knowledge validation for text attributes is usually guaranteed. In particular, the *consistency* property is satisfied as text attributes are a form of text enrichment. Thus, the consistency of text attributes is guaranteed if the method behind their definition is consistent. Note that consistency does not relate to the impact of text attributes on the given task. For instance, text attributes might represent conditioning factors of the task. In this case, their correct application is entirely due to the proposed model architecture and the learning algorithm. Likewise, the *coherence* property is usually satisfied as text attributes are just a form of text enrichment. In particular, we integrate text attributes under the assumption that their informative content is incremental and not disruptive, i.e. changes the semantic of the original text input. Therefore, their integration via substitution is sufficient to guarantee *coherence*. Lastly, the *correctness* property is also ensured as there's no actual knowledge mapping mechanism.

### 2.6.3 Class Descriptors

Class descriptors represent one of the most critical KTs. Class descriptors are a set of text descriptions regarding each class of text inputs. Human annotation guidelines for a particular task are an example of class descriptors. These human annotation guidelines are usually defined to provide an extensive description of the concepts behind each class of text inputs.

Class descriptors have been widely adopted for zero- or few-shot learning [38, 280]. Class descriptors are adopted to relate classes seen during training to unseen ones in this context. This approach favours the generalisation of a model to unseen inputs by exploiting the relations depicted by the class descriptors.

In addition, we consider a more comprehensive set of applications concerning class descriptors. The classification of text inputs with just class descriptors is a notable example. Another one is the generation of class descriptors to better cluster input data. The underlying application idea remains unchanged: we integrate textual class descriptors into a model as a rich supervision signal regarding input data.

#### Integration

We can think of class descriptors as a form of abstract textual pattern. That is, class descriptors provide a conceptual description of the properties that are shared by text inputs belonging to the same class. Regarding the knowledge integration process, the integration mechanism for class descriptors is *entailment* (Section 2.3.1). In particular, the integration dynamic is to verify if the given text input is an instance of one or more class descriptors. In other terms, we view text inputs as potential conveyors of a subset of class properties. Text input can be associated with one or multiple class descriptors depending on the given problem setup. We then view the integration process as the entailment between the class descriptors, i.e. the premise, and the given text input, i.e. the hypothesis.

#### Mapping

In the context of deep learning, entailment is often formulated similarly to embedding-based similarity operators. In this case, the knowledge mapping mechanism is similar to the one described for textual patterns (Section 2.6.1). The main difference is that entailment is achieved by enforcing an asymmetric operator [4, 250]. More generally, the knowledge mapping mechanism detects the instances, i.e. the text inputs, of each class property encoded by the given class descriptors. The knowledge mapping mechanism is then evaluated based on the degree of detection for each class descriptor. To concretely motivate the mapping between text inputs and class descriptors, a corresponding set of class rationales might be required. Forestalling the definition given in Section 2.6.4, class rationales act as a support device for the reasoning process behind the detection of class property instances [32]. Without class rationales, we can consider direct supervision information between each class descriptor and text inputs for evaluating the knowledge mapping process.

#### Validation

Class descriptors should have complete coverage of the properties conveyed by each class of text inputs. Thus, the satisfaction of the *consistency* property is of particular importance. Nonetheless, defining a complete set of class descriptors might be challenging. For instance, in the case of human annotation guidelines as class descriptors, there are plenty of domains where their definition is particularly hard. In particular, annotation guidelines might be too complex, non-stationary and far too abstract to be integrated into a data-driven approach. Furthermore, their adoption might be specific for a single domain. A notable example is provided by Argument Mining [142]. In this domain, the definition of an argument for a specific domain is often non-trivial and time-consuming [156, 195]. Nonetheless, we can employ a partial set of class descriptors as well.

Regarding *coherence*, its satisfaction is tightly related to the employed unstructured knowledge mapping mechanism. For instance, consider the case in which the knowledge mapping mechanism is an embedding-based entailment operator. This operator relies on the assumption it reflects the definition of entailment. Then, we must verify if the knowledge mapping mechanism is capable of learning to extract information from text inputs that entail a given class property.

Lastly, we may measure *correctness* if direct knowledge mapping supervision is provided. This information may be available if a supervised annotated dataset also has information about which guidelines have been used to label each text input. In some cases, correctness may be hardly measurable since the same text input can belong to a particular class depending on contextual information. A notable example is represented Argument Mining [142]. In this domain, some argumentative components can be associated with the same text input depending on the given context [35]. We should adequately contain such contextual dependency in the class descriptors as well. Nonetheless, automatically determining these dependencies is non-trivial since they reflect a change in the task formulation.

### 2.6.4 Class Rationales

Class rationales explain the reasoning behind the decision-making process of a model. We identify two types of class rationales: (i) *text rationales* and (ii) *abstract rationales*. Text rationales are part of the input text as with standard Natural Language Inference (NLI) tasks [250]. On the other hand, abstract rationales are general explanations associated with their corresponding class descriptors. Text rationales are an instance of abstract class rationales as they are grounded to the given text input. Multiple class rationales might be associated with the same class descriptor as different motivations can refer to the same class property. Class rationales can then have various instances, each specific for certain text inputs.

### Integration

The integration of class rationales is inherently dependent on the underlying set of class descriptors. Thus, class rationales share the exact knowledge integration mechanism of class descriptors, i.e. the one of *entailment*. However, the entailment is conditioned on both the given text input and the employed class descriptors. In general, the knowledge integration process of class rationales comprises of two phases: (i) the entailment between text inputs and class descriptors; (ii) the entailment between (i) and class rationales. In the case of text rationales, the second phase of knowledge integration is similar to the one described for textual patterns (Section 2.6.1).

### Mapping

In the context of deep learning, we can consider traditional entailment solutions for both knowledge integration phases as with class descriptors. If class descriptors are not provided, both entailment phases are generally replaced with a unique supervised integration mechanism. That is, direct supervision about which class rationales to select is given. In both cases, we can train deep learning models to jointly or sequentially make predictions and determine the correct set of class rationales as done in NLI [32].

The above discussion is valid for both types of class rationales. However, the mapping mechanism faces different dynamics depending on the kind of class rationale. More precisely, the knowledge mapping mechanism for text rationales has a search space solely dependent on the text input. On the other hand, abstract rationales are shared by all text inputs, and their textual representation might follow linguistic properties that differ from one of the input data. In this case, we have to consider more sophisticated mapping mechanisms to account for potential linguistic gaps.

### Validation

Similar to class descriptors, *consistency* is particularly crucial to guarantee since class rationales provide information about a model's decision-making process. However, giving a set of abstract class rationales with complete coverage over class descriptors for a given task is challenging. Such an issue is further enhanced by the added complexity of the knowledge integration mechanism. In particular, two entailment integration steps are generally considered. In contrast, text rationales are more easily defined since they are part of the text input.

Likewise, *coherence* is also dependent on the knowledge mapping mechanism of class descriptors. This property is particularly demanding to guarantee as two entailment integration mechanisms have to satisfy the same property jointly. The satisfaction of coherence is a crucial requirement of a UKI process that leverages class rationales due to the role of the knowledge.

If class descriptors are not provided, it is particularly challenging to determine the correctness of the knowledge mapping mechanism. In most scenarios, direct supervision regarding class rationales may be available for NLI [32]. Generally speaking, the integration of class rationales introduces additional requirements like model interpretability and transparency. More precisely, the knowledge integration process should define a semantic frame regarding knowledge usage. Within this frame, both knowledge integration and mapping mechanisms should expose the decision-making process of a model. As a solution, sophisticated solutions proposed in the broad domain of eXplainable Artificial Intelligence (XAI) [117] can be adopted. A notable example is given by the recently proposed Argumentative XAI [48] that embraces notions of Argument Mining for explaining deep learning models.

### 2.6.5 Additional Data

One of the simplest KTs is additional data regarding available input data. Additional data is a set of text examples likely to be derived from the same data generation distribution of input data. As with data-augmentation [72], we can treat such knowledge as an additional dataset in this case. Additional data is a KT that comes with two significant advantages: (i) providing additional data to a model is generally beneficial to its learning; (ii) if the provided data comes with annotations, i.e. labels, we can use the latter information to guide the decision making process of a model.

#### Integration

The knowledge integration mechanism for additional data is *comparison*. In particular, additional data is used as a reference point for evaluating given text input. The knowledge integration process follows the same principle of traditional machine learning methods that rely on collective information to produce a prediction. Examples are Nearest Neighbours (k-NNs) [47] and Support Vector Machines (SVMs) [46].

#### Mapping

The knowledge mapping mechanism mainly relies on semantic similarity as described for textual patterns (Section 2.6.1). In particular, text inputs should be compared with the knowledge to identify knowledge candidates that are highly similar to the given input. If task-specific information is provided, the knowledge can also constrain the knowledge mapping process. For instance, if class label information is provided, we can force the knowledge mapping mechanism to select knowledge candidates that share the input text's same class label[90].

#### Validation

The *consistency* property is entirely based on the assumption that additional data is taken from the same distribution of available data. Therefore, the only requirement for the satisfaction of *consistency* is verifying such an assumption. If *consistency* has been satisfied, the *coherence* property is also guaranteed. We only require the definition of a knowledge mapping mechanism compliant with task-specific requirements. For instance, a comparison based on semantic similarity could not be adequate if matching (text input, knowledge candidate) pairs do not rely on semantic but other properties. Lastly, we can generally consider the *correctness* property as being satisfied. More precisely, there is no particular requirement concerning the knowledge mapping mechanism. Additional constraints can be added based on provided task-specific information if any.

### 2.6.6 Contextual Data

Contextual data is the set of information that describes task-specific knowledge, such as the domain and related commonsense knowledge [250]. The main difference with text attributes is that contextual data is a form of knowledge that is shared across all text inputs. Furthermore, contextual data differs from additional data as the former may cover complementary information that derives from a different data generation distribution. Nonetheless, the role of knowledge remains unchanged and is text enrichment.

#### Integration

As for text attributes, the knowledge integration mechanism of contextual data is *substitution*. In particular, contextual data is an additional source of information that we exploit to expand the informative content of each input text.

#### Mapping

Unlike text attributes, contextual data does not provide a one-to-one mapping with each text input. Generally speaking, the act of knowledge mapping resembles information retrieval as there's no particular input to knowledge mapping relation. Thus, we can cast down mapping to several mechanisms that all fall within the general information retrieval process. Some examples are given by semantic similarity, text entailment and term indexing. In general, any knowledge mapping mechanism that allows retrieving additional information to enrich the content provided by the given text input is a form of knowledge mapping.

#### Validation

The integration of contextual data does not define a particular definition of *consistency* as there's no direct relationship between knowledge and text inputs. Consistency is then limited to providing valuable information for the task of interest. Similarly, there's no particular constraint about how we should integrate contextual data. Thus, *coherence* is generally satisfied. In particular, text enrichment provides a modified version of the input via added information. Therefore, we focus on what the model extrapolates from knowledge rather than attributing a particular role to knowledge. Lastly, the *correctness* property is not of specific relevance whether it is satisfied or not. The only constraint concerns preserving the semantic meaning of the original text input.

### 2.6.7 Text Constraints

Text constraints are one of the most challenging KTs. Text constraints are any kind of behaviour or property that enforces a model. For instance, we could constrain a generative model to generate user profiles that adhere to an initial template [301]. Dialogues are then forced to exchange information that does not violate the initial user profile. Other notable examples are code comments to guide language models for program synthesis [7, 277], natural language rules and instructions to guide reinforcement learning agents [192, 238, 305] and task descriptions and annotations to guide text categorisation and question answering [104, 275].

Generally, we distinguish between two types of text constraints: (i) *external* and (ii) *internal*. First, the knowledge enforces a specific behaviour at the task level. In particular, the output of the decision-making process of a model is directly affected by text constraints. The above example of the generative model falls within this category since dialogues are evaluated on the available knowledge. On the other hand, internal text constraints affect a part of the model architecture. As a toy-like example, an internal constraint could describe how we should define a specific intermediate operation: *"The norm of each embedding input vector should be unitary"*. Note that internal constraints inherently require a form of representation mapping to evaluate their degree of satisfaction concretely. In other terms, we have to formulate text constraints so that their representation is compliant with the given model.

### Integration

The behaviour described by text constraints requires an integration mechanism that differs from comparison, entailment or substitution. In particular, the main objective of integrating text constraints is to enforce the desired behaviour. Thus, the knowledge integration mechanism of text constraints is the one of *evaluation*. Text constraints have to be integrated into a model via a mechanism that quantifies their degree of satisfaction.

### Mapping

Generally speaking, there's no actual knowledge mapping for text constraints as their described behaviour is at the model or task level. Nonetheless, we could think of instances of each text constraint as input-specific constraints. In this case, the knowledge acts as a lookup table like in text attributes (Section 2.6.2). We can reformulate traditional learning settings like supervised classification as a particular form of text constraint knowledge integration with input-specific constraints. More precisely, each text constraint instance enforces the attribution of a specific class label to its corresponding text input.

### Validation

The *consistency* property is particularly crucial as the intended behaviour to be integrated should be compliant with the task of interest. Note that consistency is implicitly related to input data. In particular, text constraints should be feasible for each text input. However, the definition of the general behaviour is directly related to the task itself rather than to the data. Another crucial property to be guaranteed is *coherence*. In particular, coherence is tightly dependent on the constraint type. The simplest scenario concerns external text constraints where the model generates text. The initial example of a generative model for dialogues constrained on user profiles is a concrete example [301]. In this context, text constraints are text properties that have to be guaranteed throughout the dialogue. Evaluating such properties can be cast down to viewing text constraints as textual patterns that we must detect in each text input. The difference with textual pattern integration is that matching text inputs and knowledge is a mandatory requirement. On the other hand, internal text constraints require an additional representation step. More precisely, we have to translate text constraints into a format to measure their degree of satisfiability. For instance, we can consider structured knowledge representations like logic rules described in Chapter 1. In this case, the validation of the *coherence* property is entirely attributed to the representation step. Lastly, *correctness* is usually guaranteed as there's no actual knowledge mapping mechanism.

### 2.6.8 Considerations on Knowledge Types

We make a few remarks on KTs concerning (i) their dependencies with other KTs (**KT dependencies**); (ii) potential shifts of a knowledge base from a KT to another depending on the given problem formulation (**KT shifts**).

#### KT Dependencies

The integration of certain KTs has implicit dependencies with other KTs. For instance, text constraints could force a model to change its classification prediction from a class descriptor to another one. In this case, the formulation of text constraints implicitly requires the definition of adequate class descriptors. A similar approach has been explored with class labels [57, 88]. Class rationales and class descriptors give another example of KT dependencies. Class rationales are an indicator concerning the reasoning process of a model. Such a reasoning process is centred on class descriptors either explicitly (if provided) or implicitly. Note that the requirement concerning the presence of class descriptors is not at the integration level but rather at the reasoning level. In other terms, we could also formulate the integration of class rationales without an explicit definition of class descriptors. For instance, consider a supervised classification setting with class labels and textual class rationales. The way class rationales are used to explain the decision-making process of a model is inherently connected to the underlying class descriptors. The latter are unified under a single class label. Generally speaking, identifying KTs dependencies is a crucial step in a UKI process. In particular, the dependencies of the KT to be integrated are part of the preliminary phase that defines how we should integrate knowledge and its corresponding role.

#### KT Shifts

A given unstructured textual knowledge can shift from one KT to another depending on its integration process into a machine learning model. In particular, the given task at hand and the desired problem formulation determine how we should employ knowledge to obtain certain benefits. The definition of a specific KT is crucial since it automatically determines the corresponding UKI process, i.e. the UKI sub-processes and related integration properties to guarantee. For instance, sentiment aspects can either act as text attributes to enrich a particular input or be formulated as text constraints to condition the decision-making process of a model. Another example is when we employ additional data as contextual information for other text inputs. For instance, consider the task of sentence classification. In this case, the sentence neighbourhood of each input sentence, i.e. adjacent sentences, is the additional data to integrate.

## 2.7 Considerations on Unstructured Knowledge Integration

Unlike structured knowledge integration, UKI in NLP has the ambitious goal of directly using text data as added knowledge. The latter is then adopted to condition the decision-making process of a model. In this section, we discuss the general advantages and challenges regarding UKI. The underlying assumption of UKI is that the knowledge should satisfy a set of requirements regarding its integration (Sections 2.3.3 and 2.4). These requirements are either directly or indirectly based on the textual representation of the knowledge. Nonetheless, the definition of a UKI system that is robust to different yet equivalent textual knowledge representations is non-trivial.

### 2.7.1 Advantages

The main advantage of UKI is the availability of unstructured textual knowledge. Contrarily to structured knowledge, no explicit knowledge representation process is required when handling unstructured textual knowledge. UKI aims at using plain natural language knowledge as it is. Thus, we do not necessarily impose an intermediate step to convert textual knowledge into a structured form. Moreover, no particular expertise is generally required to define the knowledge itself. In principle, we could define and integrate many unstructured textual knowledge representations into a model. Nonetheless, some textual knowledge representations might ease the integration process in practice as they present a higher degree of alignment with the general problem setup. Additional factors like the knowledge integration process, the model architecture and the task of interest can strongly affect the impact factor of the integrated unstructured knowledge. As a possible solution to compensate for specific issues, a more general knowledge integration process where knowledge and input data affect each other should be considered (Section 2.4.1). For instance, the data-driven learning approach can actively modify the knowledge itself, e.g. via knowledge re-writing, to achieve better performance [114].

The capability to directly integrate unstructured knowledge also translates to better adapting to domain changes. In the context of UKI, the model jointly learns to solve the task with available information and to employ provided knowledge in a goal-oriented way. From the machine learning model point of view, we can view the capability of relying on knowledge as a relaxation imposed on the model's parametric memory capacity. That is, the model learns to use provided knowledge rather than trying to implicitly store similar information in its weights in a data-driven fashion. Under this perspective, we can then consider the setting where we test a trained model on a different unseen domain where an ad-hoc knowledge base is provided. Depending on the efficacy of the knowledge integration process and on the knowledge itself, the model should better adapt to the given domain as it has learned a general approach to solve the task that relies on available information. This dynamic is entirely different from learning to solve a particular task instance. The described method touches on other active research areas concerning transfer learning [3, 199] and meta-learning [291]. Similarly, we can also evaluate model performance by varying the textual knowledge representation. We can view knowledge as a unique hyper-parameter. Depending on the assumed value, i.e. its natural language representation, the knowledge can affect a model's decision-making process with different impacts. We consider a pre-trained deep learning model and a given task-specific knowledge as a concrete example. We can evaluate the model's performance by modifying the knowledge at inference time or during a minor fine-tuning process.

### 2.7.2 Dealing with Natural Language Knowledge

In UKI, the majority of effort is centred on understanding a given task of interest and on formulating an appropriate unstructured knowledge integration process. Nonetheless, critical challenges arise when dealing with natural language knowledge. For instance, we can develop the same abstract concept in different yet equivalent textual representations. We denote this issue as *semantic equivalence*. Other challenges are given by implicit commonsense knowledge [250] and potential missing and incomplete information. The definition of a UKI process robust to the above dynamics regarding natural language is a challenging task. More precisely, it is not to exhaustively enumerate all the possible equivalent expressions of a given text. Additional factors like the given domain also affect the style of the natural language representation of the knowledge.

Another significant issue is *contextual dependence*. We denote with *contextual dependence* the set of scenarios in which we require contextual information to grasp the meaning of a given text. Contextual information might refer to commonsense or domain-specific knowledge. For instance, operations like word sense disambiguation [19, 193], where we must determine the meaning of a word from its context, might be required. Dependencies like the previous example have to be made explicit and addressed to integrate knowledge properly. Depending on the content of the knowledge, the process of contextual dependence resolution might be particularly challenging. Contextual dependence is also related to the property of *incomplete information*. In particular, we might convey implicit or abstract concepts through a textual representation. The presence of incomplete information hinders the process of knowledge integration since the gap between the knowledge and the data increases. Under this perspective, knowledge is a set of concepts, and we interpret input data as an instance of such concepts in a given domain. We should address such phenomena to provide the model with as much information as possible. Lastly, *text noise*, such as jargon expressions, abbreviations and grammatical errors, represents another major challenge for unstructured knowledge integration. In contrast to structured knowledge that accounts for such issues via an abstract representation, a UKI system must define appropriate knowledge integration processes robust to text noise.

## 2.8 Related Work

This section provides a non-exhaustive list of current broad research areas that show affinities with UKI. In particular, we focus on the usage of unstructured text knowledge to condition the decision-making process of a model when addressing a specific task. We introduce and describe related research areas and highlight their similarities and differences with UKI.

### 2.8.1 Reading Comprehension

Open or knowledge-grounded conversational systems [196] inherently require unstructured textual information to provide an answer to given questions or to, in more general, generate adequate text. We usually retrieve such textual unstructured knowledge from multiple textual sources. Traditional methods rely on fixed APIs [26] to access external knowledge in databases.

Nonetheless, these solutions are reasonably limited and simplistic and hardly scale to real-life scenarios. To this end, recent work explores autonomous information retrieval on unstructured textual documents, such as newspapers and other online verified knowledge sources. For instance, Parthasarathi & Pineau [200] discuss news articles using Wikipedia summaries as knowledge. Ghazvininejad et al. [85] discuss local businesses in two-turn dialogues using Foursquare tips as knowledge using an extended Encoder-Decoder. The decoder receives an encoding of the context along with the external knowledge encoding.

Another notable example is given by Dinan et al. [62], in which they investigate memory-augmented neural networks (MANNs) [252, 276] and transformer models [58, 264] across a large and diverse set of topics to build human-like conversational agents. Due to extensive textual information from selected knowledge sources, data is selected according to standard information retrieval methods, such as TF-IDF and bag-of-words features.

Generally speaking, reading comprehension and question answering differ from UKI. In particular, unstructured textual knowledge represents the input of the model, and no explicit external knowledge is considered. The task is then centred on determining the small portion of input data needed. However, the methods developed for handling and reasoning on textual documents for the tasks mentioned above can be useful for UKI as well. For instance, text retrieval from potentially large documents is also considered when dealing with large unstructured textual knowledge. Indeed, such a knowledge can be viewed as a textual document where individual texts may be correlated to each other or not. A substantial effort has been recently spent for defining benchmarks and suites for addressing these knowledge-intensive tasks [178, 204]. Therefore, research for UKI can certainly take inspiration from contributions in these domains.

### 2.8.2 Few- And Zero-shot Learning

Zero- and Few-Shot Learning (ZSL, FSL) are emerging hot topics of the last decade. Both research areas fall within the umbrella of Low-Shot Learning (LSL) [280]. The general problem setup concerns generalizing to unseen data examples and classes without any example (ZSL) [38, 281] or with just a few (FSL) [13, 273]. In particular, in ZSL, training data does not contain any information about unseen classes. Additionally, we distinguish between ZSL and *generalized* ZSL (GZSL). In the latter, test data includes examples from both seen and unseen classes. To allow learning, external knowledge, usually denoted as *side information*, is employed to provide relations between seen and unseen classes. Such information is then adopted to project data and class information into a shared embedding space where decision-making occurs. Similarly, in FSL, $K$ examples regarding unseen classes are provided as reference points for learning. If $K = 0$, the task is generally denoted as *one-shot learning* (OSL) [263]. Both approaches were initially proposed for Computer Vision (CV) [273, 281] concerning image classification. A large group of studies has also explored these research areas in NLP. Nonetheless, ZSL in NLP is still in its infancy [280]. Some notable contributions for ZSL have tackled text classification [218], document classification [299] and language modelling [206], entity typing [302] and intent detection [279].

Regarding FSL, several applications have been explored. Some notable examples are neural machine translation [237], sentiment analysis [294], intent classification [83], text classification [135], relation classification [103], word embedding representation ,[115] and text generation [203]. As reported in [38], ZSL and FSL mainly adopt four distinct knowledge representations with incremental semantic richness: (i) text; (ii) attributes; (iii) knowledge graphs and (iv) logic rules. Text knowledge is usually defined as class sentence descriptions extracted from sources like encyclopedia entries such as Wikipedia [6] or class names [210] or their definitions [239]. Attributes are mainly adopted in CV and refer to classes in key-value pairs of categorical, boolean or real-valued properties [137]. Knowledge graphs concern large size ontologies like WordNet [180] or ConceptNet [245]. Lastly, Horn rules [146] or first-order formulate [221] are directly encoded into the embedding space to address unseen class relations.

Regarding UKI, we view the general problem covered by ZSL and FSL as a particular instance of UKI. In particular, in the context of NLP, textual knowledge of different granularity is employed in ZSL and FSL to establish connections between seen and unseen classes of examples. The aforementioned *side information* guides the learning stage by acting as a valuable prior hypothesis regarding how data is distributed in the latent space. Under the perspective of UKI, the knowledge employed in ZSL and FSL falls under the category of class descriptors (Section 2.6.3).

A significant point of distinction is that the employed knowledge is generally static. Additionally, knowledge is an additional input that operates at the class level. In UKI, the knowledge is not generally assumed to have full coverage of input data, albeit some KTs like textual constraints may have coverage as the desired requirement. Furthermore, the knowledge natural language representation also carries uncertainty that further encourages a non-stationary view of the knowledge. In addition, in UKI, we consider many problems concerning knowledge integration, not limited to the classification or generation of unseen examples. For instance, knowledge can solely be employed as a device to guarantee model transparency (Section 2.6.4) or express a specific behaviour in the role of a regularizer (Section 2.6.7).

### 2.8.3 Knowledge-Enhanced Text Generation

One of the most challenging tasks in NLP is represented by text-to-text generation, also denoted as text generation. The task concerns generating an output sequence $Y$ from an input sequence $X$. Although a wide variety of approaches has been explored [147], the informative content stored in $X$ is not always sufficient to produce the desired $Y$. Thus, additional sources of information generally denoted as contextual knowledge are considered to overcome the lack of awareness and understanding of $X$ and its surrounding context. The integration of knowledge for text generation is known as *knowledge-enhanced text generation* (KE-TG) [295].

As reported in [295], we generally distinguish between two types of knowledge: *internal* and *external*. The first refers to information that we can extrapolate from the input text. It includes the topic, linguistic features, keyword and text related graph structures like AMR graphs. On the other hand, external knowledge refers to large knowledge bases, knowledge graphs and unstructured text. In recent years, several contributions have tackled the challenge how acquiring and integrating knowledge for text generation. In particular, research has been categorized under the same taxonomy presented in Chapter 1 regarding informed machine learning. Indeed, KE-TG is one instance of the latter: (i) model architecture, (ii) learning algorithm and (iii) additional data.

Concerning unstructured text knowledge, work in KE-TG has explored integrating topic words [298] for jointly topic modelling and text generation, keyword (terms, phrases) assignment [240, 243] and extraction [42, 144]. In the first, keywords from a fixed vocabulary are employed during the generation phase. Conversely, keyword extraction is similar to information retrieval, selecting keywords from available documents. Furthermore, linguistic features like POS tags and NER tags have been employed as forms of input enrichment [65, 191, 307].

Also raw text documents from online sources, such as encyclopedias and social media, have been integrated as an additional input to guide the generation phase [71, 116]. In particular, according to [295], we distinguish between *text retrieval-based* methods and *background knowledge comprehension* methods. The integration is cast as a candidate search problem where multiple documents are employed to retrieve fine-grained text paragraphs, denoted as candidates. Such an approach is labelled as *retrieval-augmented* generation (RAG) [133, 143].

Additionally, RAG methods can include intermediate candidate re-rank and rewrite phases to retrieve a particular reference set of additional information. RAG is declined as *retrieve, re-rank and rewrite* ($R^3$) [34, 97, 270]. Lastly, background knowledge integration can be viewed as a particular case of text retrieval-based methods when the retrieval phase is centred on a single document. In particular, a more extensive comprehension of the document, rather than limiting to candidate retrieval, is required to integrate the background knowledge into the generation phase properly. Notable examples are mainly in the fields of conversational agents [175, 209].

Within the scope of UKI, KE-TG can be viewed as a particular instance of UKI when considering unstructured text knowledge integration. In particular, the knowledge encodes two primary intents: (i) explicit information provision and (ii) generation conditioning. First, the integrated knowledge for text generation can be associated with KTs like text attributes, additional data and contextual data. The knowledge integration mechanism is *substitution* to overcome the limitations of generating text $Y$ solely relying on input text $X$.

Current research mainly explored embedding-based knowledge mapping mechanisms where knowledge is encoded and fused with the input text [20, 98, 175]. Moreover, unstructured knowledge integration via substitution is also modelled to explicit condition the generation phase. In this case, knowledge is related to the KT of text constraints as it imposes a specific behaviour. Nonetheless, most applications rely on more structured knowledge like knowledge graphs as an evaluation criterion of the described conditioning behaviour [131].

In UKI, we consider additional KTs that can be considered for text generation. For instance, class descriptors can encode prior information or act as evaluation criteria. Additionally, we also consider more general knowledge integration and mapping mechanisms for introducing unstructured integration. For instance, due to the ample search space, knowledge retrieval in KE-TG is often limited to traditional lexical and syntactic methods like Tf-Idf [42, 240]. Thus, the problem of knowledge mapping is usually limited to direct supervision (if available).

# Chapter 3

# Knowledge Extraction from Text

A general consideration concerning unstructured knowledge integration (UKI) is the definition of adequate knowledge integration processes capable of satisfying the requirements of a given knowledge type (KT). In particular, each KT defines its requirements for knowledge integration. For instance, text constraints require an explicit evaluation method to assess their degree of satisfiability (Section 2.6.7). To satisfy knowledge integration requirements like the above, structured knowledge representation may be required as an intermediate knowledge representation. For instance, suppose the problem of integrating class rationales into a deep learning model. To verify properties like *coherence*, we require a method that exposes the connection between particular text input and the selected class rationales subset. In other terms, we need a semantic frame that allows answering the following question: *"For what reasons was input X connected to knowledge K?"*. A possible solution is to organise class rationales and input data into a knowledge graph. In particular, we can use the provided graph edge semantic to motivate a connection from input data to knowledge. Such reasoning is possible as the knowledge graph provides a known edge semantic.

The knowledge graph in the above example is the product of a corresponding structured knowledge extraction phase (KE). Structured knowledge represents a valuable source of information as it provides its own interpretation semantic. It also defines a high-level representation with several advantages like textual noise filtering and informative content enrichment. These two properties of KE, i.e. the representation and the related semantic frame, can be employed to guarantee specific properties of a UKI system as described in the previous example. We view KE as a sub-process of UKI as it provides valuable tools for satisfying specific integration properties of KTs. The derived knowledge representation and semantic frame may differ from the more general-purpose structured representations as shown in Section 1.2 since they are tightly dependent on the given KT.

In this Chapter, we consider two examples of KE. First, knowledge is cast down to structured textual patterns derived from input intermediate representations. In the second, knowledge is a supplementary high-level annotation layer employed to bridge multiple inputs into a single structured one. We experimentally assess to what extent these knowledge representations improve the learning process of deep learning models. Such improvement could be in terms of model performance level (Section 3.3) or the amount of information required for learning (Section 3.4). Lastly, we conclude by drawing some considerations about the applicability of such valuable knowledge extraction methods.

## 3.1 Knowledge Representation

This chapter considers two distinct knowledge types in two different domains, respectively. First, we explore the task of argument detection [225], i.e. subjective claims and related supporting facts, in textual documents. In this scenario, we use existing parsing models to automatically retrieve the constituency parse tree of a given input text. Then, we define our knowledge as the set of sub-tree structures that act as characteristic features of the given input text. In other terms, we view the knowledge as a high-level textual and structured feature. We can label the constituency tree as a first-order knowledge type, automatically derived from the given input text. On the other hand, the automatically extracted sub-tree structures from the input constituency tree represent a second-order knowledge type.

Second, we introduce a conversational dataset on scientific papers. We explore the tasks of dialogue generation and content retrieval on scientific papers. In this domain, we employ argumentative models to annotate scientific papers automatically. Scientific papers are enriched with information about their claims and related supporting hypotheses. We then view these argumentative annotations as our knowledge base to guide the content retrieval task without using ad-hoc supervised annotations. In particular, the argumentative graph obtained from extracted arguments and their relations is employed to connect the dialogue with the associated scientific paper.

In both scenarios, the added knowledge is an auxiliary source of information, either as a high-level representation or an additional annotation layer. Such auxiliary property of knowledge reflects the main objective of KE: obtaining a refined input representation that is less subdued to textual noise sources and that provides task-specific valuable information. In particular, the latter is not provided by the original text input.

## 3.2 Challenges

We identify two significant challenges regarding KE for the introduced case studies: (i) *knowledge validation*; (ii) the *availability of labelled data* to train a KE model. The first challenge arises for argument detection with automatically extracted sub-tree structures. On the other hand, the automatic labelling of scientific papers requires the pre-training of argumentative models on accurate data.

In the case of automatically extracted sub-tree structures, we require the definition of a criterion for assessing the validity of such structures. More precisely, our evaluation criterion covers: (i) *structure-specific constraint satisfaction*, i.e. the extracted sub-tree structures are valid according to their definition; (ii) *knowledge importance*, i.e. to what extent does knowledge impact the overall model task performance; (iii) *knowledge associative mapping*, i.e. if the extracted knowledge is selectively employed based on the given input characteristics. Furthermore, we might require an adequate semantic frame for explaining how extracted sub-tree structures are associated with their corresponding input text.

Another common issue regarding KE is the availability of adequate data for training knowledge extraction models. In the context of conversational agents for scientific papers (Section 3.4), the lack of in domain datasets for argument annotation makes KE particularly challenging. Generally speaking, this issue translates to adopting a KE step or not. In many domains, abstract text representations are important to address existing difficulties, such as the low amount of data, noisy data and an overall challenging task. However, the possibility of automatically extracting an adequate knowledge-enriched input representation is not always guaranteed.

## 3.3 Argument Detection with Automatically Extracted Sub-tree Structures

We tackle the task of argumentative sentence detection, a sub-task of Argument Mining (AM) [156, 161]. We formulate the task as a classification problem where a sentence can either be argumentative or not. In some scenarios, the argumentative label is split into the argument individual components. Thus, the task is like argumentative component detection [142]. To solve this challenging task, we inspire an existing solution [155, 157]. In this work, authors exploit SVM classifiers with bag-of-words features and structured ones based on Tree Kernels (TKs) [186]. In particular, TKs are applied on the corresponding constituency parse tree of the given input sentence. The TK measures similarity based on each sentence constituency parse tree. In this way, the sentence constituency parse tree is considered in addition to word-level information. Such a conjoined approach offers models the capability of capturing both syntactic- and semantic-level textual properties. We consider structured textual patterns a valuable indicator of labelling a given sentence. Indeed, in the context of AM, recent works have explored the task of retrieving textual features that act as indicators of a given argument category [52, 96].

On this idea, we seek to apply the concept of TKs to deep learning models. We assume to have the constituency parse tree of the given sentence as an intermediate text representation. Such requirement is not particularly tight as a wide variety of sequential tagging models exist for this purpose [167]. Given the structured input type, we leverage Graph Neural Networks (GNNs) [128, 229] to handle the input constituency parse tree. We do not want to propose a solution that is restricted to tree-like structures. Instead, we use GNNs to extend our method to graph-like structures without much effort. Similarly to TKs, we condition the GNN architecture to extract node groups, i.e. words and POS tags, compliant with the given TK definition. Additionally, we enforce the model to retrieve the minimum set of structured features relevant to the task.

We initially provide some background about GNNs and TKs. Then, we offer a detailed description of our approach to integrate the concept behind TKs into GNNs. Lastly, we report experimental results and future work directions to demonstrate the importance of extracting structured features. For more details about this work, please refer to [225].

### 3.3.1 Background

#### Graph Neural Networks

Graph neural networks (GNNs) were first introduced in [229] and have become a key component for handling structured data [278]. Given their expressive power [284], GNNs have been applied to various domains [278]. Several architectures have been proposed, ranging from simple aggregating methods such as Graph Convolutional Network (GCN) [128] and Message Passing Neural Network (MPNN) [66, 86] to advanced architectures like Graph Attention Network (GAT) [265] and GraphSAGE [102]. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \{v_1, \dots, v_n\}$ vertices and $\mathcal{E} = \{e_1, \dots, e_m\}$ edges, a GNN generally employs the following message-passing aggregation function:

$$H^{t+1} = f(A, H^t; \theta^{t+1}) \tag{3.1}$$

where $H^t = \{h_1^t, \dots, h_n^t\}$ is the node representation matrix $n \times d$ at time $t$, where $d$ is the embedding dimensionality. Each node $h_i^t$ is the node embedding vector and $A$ is the binary adjacency matrix given $(\mathcal{V}, \mathcal{E})$ and $\theta^t$ are model parameters at time $t$. Our methodology is based on the GCN architecture due to its simplicity and effectiveness. However, we are not limited to this architecture. Nonetheless, the proposed setup applies to any GNN like a plug-and-play module. In the case of a GCN, the aggregation function takes the following grounding:

$$H^{t+1} = ReLU(\tilde{A}H^t W^t) \tag{3.2}$$

where $D$ is the degree matrix, $\tilde{A} = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ is the degree normalized adjacency matrix, $W^t$ is a trainable weight matrix. Such aggregating operation can be stacked multiple times to take into account distantly connected nodes progressively.

### Differentiable Pooling

For particular tasks, e.g. graph classification, a hierarchical representation of the input graph in terms of the constituting nodes is adopted for nodes cluster learning. Such operation is formally denoted as node clustering and can be iteratively applied similarly to message-passing to a graph to determine its building node groups. Differentiable Pooling (DiffPool) [292] is the first example of differentiable node clustering in neural networks. In particular, DiffPool views node clustering as a soft node assignment task, where $n$ input nodes are associated with $k$ node clusters. Given input node embeddings $H \in \mathbb{R}^{n \times d}$, the pooling layer is defined as follows:

$$P = softmax(HW_P) \tag{3.3}$$

where $W_P \in \mathbb{R}^{d \times k}$ is a learnable weight matrix and $P \in \mathbb{R}^{n \times k}$ is the pooling soft assignment matrix. Lastly, node cluster embeddings $\tilde{H} \in \mathbb{R}^{k \times d}$ are determined as follows:

$$\tilde{H} = P^T H \tag{3.4}$$

Likewise, the adjacency matrix $\tilde{A} \in \mathbb{R}^{k \times k}$ for new node clusters is built using previous node adjacency matrix $A$:

$$\tilde{A} = P^T AP \tag{3.5}$$

### Tree Kernels

In the framework of kernel machines, a decision function $f(x)$ is defined as the weighted combination of the contributions of a subset of training elements, named support vectors:

$$f(x) = \sum_{i=1}^{m} \alpha_i y_i K(x, x_i) \tag{3.6}$$

The kernel function is a symmetric, positive-definite function that captures the degree of similarity between two examples. In the case of a TK, we typically measure the similarity by

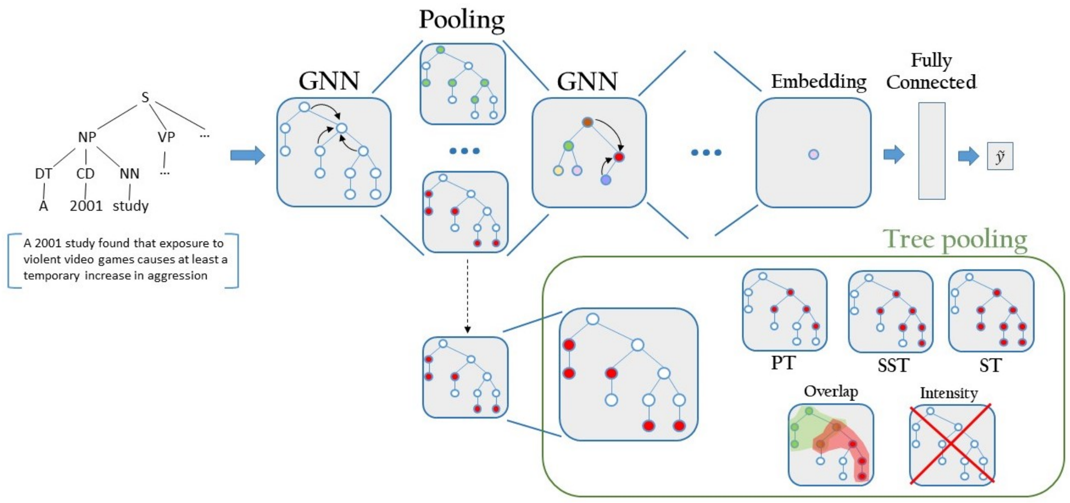


Figure 3.1: Tree-constrained GNN architecture. The constituency tree of a given sentence is fed as input to the network. The latter is defined as a stack of GCN and DiffPool layers, where the last DiffPool block has $k = 1$ to determine a final tree embedding. The retrieved tree embedding is fed to a multi-layer perceptron for classification. In the case of tree regularizations, each pooling node cluster is subject to such constraints for structured learning.

considering the number of substructures that the two trees have in common. Such substructures are usually named fragments. We can think of each fragment as a feature upon which the similarity is evaluated. Thus, it is clear that different fragments induce different TK functions, and complex definitions of fragments can produce vibrant feature sets. Given two trees $T_x$ and $T_z$, a TK is defined as:

$$K(T_x, T_z) = \sum_{n_x \in N_{T_x}} \sum_{n_z \in N_{T_z}} \Delta(n_x, n_z) \tag{3.7}$$

where $N_{T_x}$ and $N_{T_z}$ the set of nodes in $T_x$ and $T_z$, respectively, and $\Delta(\cdot,\cdot)$ computes the similarity between two such nodes, by exploiting the chosen fragments. We will consider three of the most widely used TKs for NLP: namely, the SubTree Kernel (STK), the SubSet Tree Kernel (SSTK) and the Partial Tree Kernel (PTK). In STK, a fragment is any subtree of the original tree. SSTK is very similar to STK, but the subtree can also terminate at pre-terminals. PTK is much more general since any portion of a subtree is a possible fragment. For all the technical details of the implementation of these kernels, see [186] and references therein.

### 3.3.2 Tree-Constrained GNNs

Inspired by the effectiveness of tree kernels, we investigate the problem of integrating such structured notions into deep learning models. Conversely, to well-known tree compliant solutions, such as Recursive Neural Networks (RecNN) [246] and Tree-LSTM [255], we formulate the task of learning structured node clusters as the building block features of the input graph. In particular, we view the problem of extracting kernel specific tree fragments as a node soft assignment task by constraining the pooling operation. Each pooled node cluster should comply with a particular tree definition dictated by a given tree kernel. Unlike node clustering, tree extraction does not necessarily impose mutually exclusive assignments. In most cases, node sharing between extracted trees is unavoidable due to the given fragment definition (e.g., for the STK).

### Naive Solution

As a first proposal, we present a naive solution that determines all possible tree fragments of a given input tree beforehand. Subsequently, the pooling layer learns to extract the most relevant fragments for the task during training. Formally, we introduce a tree fragments matrix $T \in n \times s$ like the pooling matrix $P$, where $s$ is the number of previously identified tree fragments. Subsequently, $P$ is constrained to match *only one* column on $T$. Intuitively, such an approach can only be considered when the number of nodes $n$ is limited. In particular, specific tree definitions like PTK and SSTK are of combinatorial complexity. Therefore, complete tree fragments enumeration cannot stand as a general-purpose solution.

### Tree Structure Regularization

Rather than exhaustively enumerating tree fragments, a more general approach involves injecting each tree definition into the model itself. In this way, learning tree fragments is declared as a constrained differentiable operation that does not alter model architecture. It is then possible to automatically extract the most relevant fragments according to a specific kernel definition without the need to exhaustively list all of them. Therefore, we only need to define appropriate regularizations that reflect the structured output we would like to obtain. Formally, we constrain Eq. 3.3 so that each node cluster minimizes all the constraints associated with the given fragments:

$$P = softmax(HW_P) \quad s.t.\ \mathbf{C}^{frag}(P) = 0 \tag{3.8}$$

where $\mathbf{C}^{frag} = \{C_1^{frag}, \dots, C_\ell^{frag}\}$ is the set of constraint functions associated with a specific fragment definition. Formally, we label Tree-Constrained GNN (TC-GNN) a GNN model with a constrained differentiable pooling layer as in Eq. 3.8.

### Kernel-induced Constraints

We require decomposing a tree's concept into its constituting properties to define appropriate tree constraints according to a given tree kernel.

1. [*Contiguity*] A tree is a connected set of nodes.
2. [*Root node*] A tree has only one node at the lowest depth, denoted as the root node.
3. [*No cycles*] A tree allows no cycles.
4. [*Root span*] All nodes in a tree belong to the root node span, i.e. the set of children nodes, either directly or indirectly, of the root node.

Additionally, we consider tree kernel-specific definitions:

1. [*ST*] for ST, all leaf nodes in the root node span must be considered;
2. [*SST*] for SST, all non-leaf nodes of a node $x$ in the root node span have to be considered if $x$ is considered.

Note that in the case of tree-structured inputs, the satisfaction of the *Contiguity* property inherently satisfies *Root node*, *No Cycles* and *Root span* properties. Additionally, for PTK, it is only required to fulfil the connected set of node properties in tree-structured inputs.

### Contiguity Constraint

Inspired by [22], the constraint solely relies on the graph topology as follows:

$$C^{cont} = 1 - \frac{Tr\left(P^T(\overrightarrow{A} - diag(A))P\right)}{Tr\left(P^TP - \frac{1}{\mathbb{1}_{P_i^2 \geq \varepsilon}} P^TP\right)} \tag{3.9}$$

where $\overrightarrow{A}$ is a masked version of $A$ that only considers forward connections, i.e., from root to leaves, and $\varepsilon \ll 1$. The numerator computes node-to-node intensities, i.e., edges. Note that the numerator excludes self-intensities, i.e. the main diagonal is removed. On the other hand, the denominator just computes self-node intensities minus the mean self-node intensity value. In simple terms, the constraint could be considered the ratio between pooled edges and pooled nodes minus 1, since, in a tree, the number of edges is equal to $n-1$, where $n$ is the number of nodes. Technically, we could replace the mean self-node intensity with a max as a more greedy estimate.

Experimentally, we chose a masked mean to address large input trees as a more stable solution. An additional side effect of this constraint is to ensure pooling intensity uniformity. Granting more flexibility in terms of pooling intensity is possible but requires some sort of node intensity capping, which, however, may lead to harder optimization[1].

[1]The original formulation of tree constraints followed this perspective, but we later substituted it due to their difficulty of optimization.

### ST Constraint

To be an ST, a connected set of nodes requires considering all the nodes under the root node span. We can impose this requirement locally by enforcing pooling children nodes of node $x$ when $x$ is pooled. The ST constraint leverages graph topology in adjacency matrix $A$ and corresponding degree matrix $D$, which simply associates its branching factor to a node. We can obtain $D$ from $A$ by summing row-wise. Formally, the ST constraint is defined as follows:

$$C^{ST} = 1 - \frac{Tr\left(P^T(\overrightarrow{A} + \overleftarrow{A}(1-L))P\right)}{Tr\left(P^T(\overrightarrow{D} + \overleftarrow{D}(1-L))P\right)} \tag{3.10}$$

where $\overrightarrow{A}$ and $\overleftarrow{A}$ are masked versions of $A$ that only consider forward and backward connections, respectively. $L$ is the leaf mask vector, i.e., it is 1 for leaf nodes and 0 otherwise. It can be computed by summing $\overrightarrow{A}$ row-wise (leaf nodes are the only nodes with no children). The numerator computes node-to-node intensities, i.e., edges, but masks some. More precisely, the first term of the numerator, $P^T\overrightarrow{A}P$ computes forward edges only. On the other hand, the second term, $P^T(\overleftarrow{A}(1-L))P$ computes backward edges only for leaf nodes. The denominator follows the same perspective but multiplies self-node intensities by corresponding node degree.

### SST Constraint

The SST constraint can be considered a particular case of the ST constraint.

$$C^{SST} = 1 - \frac{Tr\left(P_L^T\overrightarrow{A}P_L\right)}{Tr\left(P_L^T\overrightarrow{D}_L P_L\right)} \tag{3.11}$$

where $P_L = PL$ and $D_L = \sum_{\{j,L_j \neq 0\}} A_{i,j}$. The main difference with ST is that we are just filtering out leaf nodes in SST. In particular, the numerator considers forward node-to-node intensities between non-leaf nodes. In contrast, the denominator multiplies non-leaf self-nodes intensities by their forward masked degree value (leaf nodes are not considered).

## 3.3.3 Issues

### Sigmoid-based Pooling

Tree regularized pooling does not necessarily enforce mutually exclusive assignments. In other terms, a node might be assigned to multiple tree fragments. This scenario is prevalent when considering tree definitions like ST and SST. On the other hand, differentiable pooling (Eq. 3.3) applies a softmax operation, hindering constraint satisfaction, especially when considering large values of $k$. To this end, we replace softmax with a sigmoid activation function. On the one hand, we lose the softmax advantage of automatically building different node clusters. On the other hand, we obtain some desired properties. For instance, a low value of $k$ forces a softmax-based pooling to define large tree fragments. However, the sigmoid activation function allows having a small number of relevant sub-trees while maintaining freedom of choice regarding their composition. Furthermore, sigmoid-based pooling potentially allows node filtering by not assigning a node to any node cluster.

### Avoiding Degenerated Solutions

Degenerate scenarios inherently accompany the attained flexibility of tree constraints due to the soft assignment operation. Similarly to softmax-based pooling, we have to deal with two degenerate behaviours: i) all node clusters reflect the same tree structure; ii) nodes are assigned to a single node cluster while the remaining node clusters are ignored. Albeit there exist reasonable solutions as in [22], some modifications are required since softmax properties no longer hold.

### Overlap Constraint

We require regularising the degree of overlap among node clusters to overcome uniform node clusters. Differently from the orthogonality loss used in [22], we allow tree fragments to overlap up to a controlled threshold without enforcing node clusters to be of the same size. Thus, we devise the overlap constraint solely on the ratio between intra and self node clusters intensities as follows:

$$C^{overlap} = \left\| \max\left( \frac{\tilde{P}}{\sum_j diag(P^T P)_j} - \delta, 0 \right) \right\|_F \tag{3.12}$$

where $\tilde{P} = P^T P - diag(P^T P)$ and $\delta \in [0,1]$ is the overlap threshold. In other terms, Eq. 3.12 computes the intensities of shared nodes between pairs of node clusters, normalized by each node cluster self-intensity. Such derived ratio must not exceed the overlap threshold $\delta$.

### Minimum Intensity Constraint

The second degenerate scenario regards empty node clusters. Similarly to the overlap constraint, we introduce a regularization term that penalizes node clusters with low intensity. Switching to sigmoid-based pooling loses the convenient softmax property of preserving the cumulative node intensity between input and the pooled graphs. To overcome this issue, we enforce a minimum node cluster intensity as follows:

$$C^{intensity} = \left\| \max\left( (\alpha \frac{n}{k})I - diag(P^T P), 0 \right) \right\|_F \tag{3.13}$$

where $\alpha \in [0,1]$ is a hyper-parameter controlling the minimum intensity threshold. Such a constraint enforces each un-normalized node cluster self-intensity to be greater than or equal to $\alpha \frac{n}{k}$. In other terms, each node cluster self-intensity should at least match the intensity of a node cluster of $\frac{n}{k}$ nodes whose self-intensity is equal to $\alpha$. Note that the presented constraint formulation is just one instance out of many solutions that work well for our case study. For example, one could ignore $k$ and define a specific node cluster threshold.

### 3.3.4 Adaptive Learning

A major problem with learning soft tree constraints regards the nature of the learning problem. More precisely, the proposed tree regularizations are not always concurrent throughout training. This issue can lead to an unwanted locally optimal solution where some tree constraints are highly violated. We should avoid such intermediate results since we explicitly require different tree fragments. A well-defined structured representation also increases model interpretability by directly inspecting the obtained node pooling. As an example of conflict between constraints, consider the overlap and minimum intensity constraints (Eqs. 3.12 and 3.13). Both of them could be simply minimized by pooling sparse node groups.

However, we would primarily violate the contiguity constraint (Eq. 3.9) as a consequence. Likewise, kernel specific constraints (ST and SST) cannot be satisfied if inter-node clusters constraints prevail. As a naive approach, one could calibrate ad-hoc weighting coefficients for constraints to find their best linear combination. However, such antagonist dynamics among constraints might occur multiple times during the learning process and depend on the model initialization.

Thus, an approach based on fixed coefficients hardly leads to a solution where most of the constraints are satisfied. To this end, we propose to adopt the dual Lagrangian framework [74] to define dynamic coefficients that are periodically updated during learning proportionally to the degree of constraints violation. Under this perspective, a constraint-based problem is relaxed as follows:

$$\mathscr{L} = \arg\min_{\theta} f_\theta(x) + \lambda g(x) \tag{3.14}$$

where $\lambda \geq 0$ are the Lagrangian multipliers. In order to obtain the best Lagrangian relaxation, the Lagrangian dual can be used to find the best $\lambda$ value:

$$\mathscr{L}^{dual} = \arg\max_{\lambda \geq 0} (\mathscr{L}) \tag{3.15}$$

In our experimental scenario, the model is trained to optimize the following relaxed constrained objective:

$$\mathscr{L} = \mathscr{L}^{CE} + \sum_i \lambda_i C_i^{frag} \tag{3.16}$$

where $\mathscr{L}^{CE}$ is the cross-entropy loss for classification.

### 3.3.5 Experiments

As a testbed for our approach, we consider argumentative sentence detection a sub-task of argument mining. The goal of argument mining is to extract arguments for text collections automatically. One of the major tasks is to identify argument components, such as claims or premises, within a sentence.

The problem is formulated as a binary classification task where a sentence can be labelled as a particular argument component or evaluated as non-argumentative [156]. Depending on the annotation schema used in a given corpus, the task is slightly different, as it involves identifying claims, premises, or both. Several approaches have tried to shed light on the critical properties of arguments to determine common patterns, e.g., syntactic and rhetoric, among single or multiple domains [142, 156]. Primarily inspired by the work by Lippi and Torroni [155], we seek to extract tree-structured word patterns that, in combination with automatically inferred semantic features, may help identify argumentative phrases. Additional details regarding employed GNN architectures are reported in Appendix A.

#### Data

We carry out an extensive experimental evaluation on four distinct datasets for argumentative sentence detection.

- **IBM2015 [217]**: an argumentative corpus built in the context of the Debater project [241]. The data set consists of Wikipedia pages grouped into 58 topics. Annotations consist of context-dependent claims (i.e., with the underlying assumption that the topic is given) and evidence (i.e., premises) supporting such claims.

- **UKP Sentential Argument Mining Corpus (UKP-Sent) [248]**: it contains a large variety of heterogeneous sources, including news reports, editorials, blogs, debate forums

Table 3.1: Corpora for argumentative sentence detection.

| Corpus | No. Sentences | Class Distribution | Task |
|---|---|---|---|
| IBM2015 | 82,718 | 2,388 claim, 4,614 evidence, 940 claim and evidence, 76,656 non-argumentative | Claim vs. non-claim, Evidence vs. non-evidence |
| UKP-Sent | 25,492 | 6,195 argument against, 4,944 argument for, 14,353 non-argumentative | Argument vs. non-argument |
| PE | 6,826 | 751 major claims, 1,506 claim, 3,832 premise | Merged claims vs. premise, |
| AbstRCT | 35,012 | 1,390 claim, 2,808 premise | Claim vs. premise |

and encyclopedia articles, on eight controversial topics. The argument annotation schema distinguishes arguments attacking or supporting a given stance from not argumentative sentences.

- **Persuasive Essays Corpus (PE) [247]**: a collection of 402 documents extracted from an online community regarding essays discussion and advice. Documents are annotated at token-level, and the argument annotation schema distinguishes the following argumentative components: major claim, claim, and premise.

- **PubMed RCT Abstracts (AbstRCT) [172, 173]**: a collection of abstracts from scientific papers regarding randomized control trials for the treatment of specific diseases. The annotation schema reflects the well known claim-premise model [156] and is applied at token-level.

Table 3.1 provides a detailed summary of the corpora. In particular, since we are mainly interested in detecting arguments at the sentence level, we only consider argumentative components for classification for token-level annotated datasets (such as PE and AbstRCT) [76].

### Experimental Setup

To ensure direct comparison with existing baselines, we follow, whenever possible, the same evaluation methodology employed in the literature. As mentioned in Section 3.3.5, for PE and AbstRCT, we focus on argumentative component categorization to define a sentence-level argumentative classification properly. This simplification avoids a multi-label setting where phrases could contain multiple argumentative components. Such a setting occurs for IBM2015 where, in some cases (see Table 3.1), a sentence includes a claim and an evidence.

To solve this issue, as in [157] we carry out two distinct binary classification tasks concerning identifying a single argumentative component at a time. Additionally, for PE, the distinction between major claim and claim is mainly determined by their position in the essay. In particular, there are cases in which the same sentence is distinctly labelled belonging to both classes within the same paragraph. To better focus on identifying unique structured features for each class, we merge the two claim classes into a single one.

For UKP-Sent, we consider a binary classification task that discriminates between argumentative and non-argumentative sentences. The motivation lies in the employed annotation schema, which focuses entirely on the stance information (for/against) rather than on distinct argumentative components. To guarantee the same task, i.e., identifying argumentative components, we merge labelled arguments into a single class.

As competitors, we consider the TK-SVM classifier with SSTK as in [157], as well as LSTM-based models of increasing complexity: (i) a bi-directional LSTM that works on the input sentence (BLSTM); (ii) a bi-directional LSTM that sees the constituency tree as a sequence[2] (Syn-BLSTM); (iii) a combination of (i) and (ii) via a self-attention layer (Dual-Syn-BLSTM). Additionally, we also consider standard tree-aware LSTM models like Tree-LSTM [255]. All neural models are regularized at training time by applying early stopping on the validation F1-score.

We hereby report additional evaluation details for each data set:

- **IBM2015**: The original corpus is organized into two splits: 19 topics are used as a held-out set for model selection and hyper-parameter tuning, whereas the remaining 39 topics are used for the final training and evaluation. We adopt a 5-fold cross-validation on the latter set of 39 topics while we calibrate the hyper-parameters of our architecture using the available held-out set.

- **UKP-Sent**: We consider the same leave-one-out (LOO) test, repeated ten times, as in [214]. We do not compare with strong LSTM-based and BERT-based baselines as reported in [214] due to the different training setup. However, in early experiments with the same classification task as in [214] the performance of GNN models nearly matched that of LSTM-BERT and BERT-base.

- **PE**: The original corpus is organized into two splits: 322 essays for training and 80 for test. We perform a 5-fold cross-validation on the training data for model selection, and

[2]Nodes are listed as a depth-first enumeration starting from the root node.

we finally evaluate performance on the test set. Due to the limited size of the corpus, for neural architectures, we also employ a multi-start procedure, choosing the best model on the validation set via early stopping.

- **AbstRCT**: We follow the same evaluation methodology as in [79], where a repeated train-and-test procedure is considered both for tuning and testing. In addition to the neural baselines introduced for the other corpora, we consider state-of-the-art models for this task as reported in [79]. In particular, these baselines are complex neural architectures mainly based on residual networks and the attention mechanism [77].

### Results

**IBM2015** Table 3.2 reports classification performance for the IBM2015 corpus. GNN models outperform their recurrent counterparts by a large margin in both cases. Such results corroborate the hypothesis that the information encoded by constituency trees is better handled by GNNs rather than recurrent models. The advantage is further emphasized when considering the more extended input sequence defined by the constituency tree for the related sentence, which recurrent baselines struggle to handle appropriately.[3] The introduction of the pooling layer brings particular benefits to claim detection. Tree-regularised GNNs consistently improve performance to their standard pooling-based counterpart for both classification tasks while satisfying the imposed structured constraints. Such a result is aligned with our initial hypothesis that arguments of the same class share common underlying linguistic patterns.

**UKP-Sent** Table 3.3 reports classification performance for UKP-Sent corpus. GNN models significantly outperform other neural baselines by a large margin. Additionally, tree-aware GNNs consistently improve over the standard and the pooling-based models. Such results, coupled with the notable performance of TK-SVM, which matches tree-GNN models, suggest that structured patterns are valuable indicators for detecting argumentative phrases.

**PE** Table 3.4 reports classification performance for PE corpus. GNNs perform better than recurrent baselines. Differently to IBM2015, all models differ by 1-2 percentage points when considering the macro F1-score. Nonetheless, when considering the F1@C column, GNNs significantly outperform all the other models. In particular, tree-regularized GNNs achieve the highest performance for both argument classes.

[3]The 99% quantile concerning the number of nodes per tree is 160.

Table 3.2: Classification performance on IBM2015 corpus. For each argumentative class, i.e. claim and evidence, we report achieved F1-score on the test set.

| Model | F1@C | F1@E |
|---|---|---|
| BLSTM | 15.0 | 16.4 |
| Syn-BLSTM | 14.0 | 14.7 |
| Dual-Syn-BLSTM | 14.7 | 15.3 |
| Tree-LSTM | 11.5 | 12.4 |
| TK-SVM | 12.7 | 8.9 |
| GNN | 18.5 | 18.8 |
| GNN + DiffPool | 19.7 | 18.4 |
| TC-GNN (STK) | 18.3 | 18.5 |
| TC-GNN (SSTK) | <u>20.1</u> | **19.2** |
| TC-GNN (PTK) | **21.3** | <u>19.1</u> |

Table 3.3: Classification performance on UKP-Sent corpus with merged argument classes: class Arg contains both pros and cons for each stance.

| Model | F1@Arg |
|---|---|
| BLSTM | 0.513 |
| Syn-BLSTM | 0.572 |
| Dual-Syn-BLSTM | 0.544 |
| Tree-LSTM | 0.517 |
| TK-SVM | 0.605 |
| GNN | 0.595 |
| GNN + DiffPool | 0.590 |
| TC-GNN (STK) | **0.610** |
| TC-GNN (SSTK) | **0.610** |
| TC-GNN (PTK) | <u>0.608</u> |

**AbstRCT** Table 3.5 reports classification performance for AbstRCT corpus. As in [79], we report classification metrics for each test set. GNN models manage to reach satisfying performance compared to ResArg and ResAttArg strong baselines. The GNN extension with DiffPool and, in particular, TC-GNNs consistently improve performance for the base GNN. Considering recurrent baselines, the structured representation of the constituency tree is beneficial for the BLSTM baselines, where both the Syn-BLSTM and the Dual-Syn-BLSTM nearly match the ResArg model.

Table 3.4: Classification performance on PE corpus with merged claim classes: class C contains both claims and major claims.

| Model | F1@C | F1@PR | F1@Macro |
|---|---|---|---|
| BLSTM | 62.05 | 78.16 | 70.10 |
| Syn-BLSTM | 60.69 | 77.61 | 69.14 |
| Dual-Syn-BLSTM | 59.79 | 79.10 | 69.44 |
| Tree-LSTM | 54.78 | 57.48 | 56.13 |
| TK-SVM | 62.11 | 77.24 | 69.68 |
| GNN | 64.17 | 78.07 | 71.12 |
| GNN + DiffPool | 64.08 | 78.03 | 71.05 |
| TC-GNN (STK) | 65.42 | 77.82 | **71.62** |
| TC-GNN (SSTK) | 64.40 | 78.02 | 71.21 |
| TC-GNN (PTK) | 64.36 | 78.74 | <u>71.55</u> |

Table 3.5: Classification performance on AbstRCT. We report per class F1-score and macro F1-score for each reported test set.

| | Neoplasm | | | Glaucoma | | | Mixed | | |
|---|---|---|---|---|---|---|---|---|---|
| Model | F1@C | F1@PR | F1 | F1@C | F1@PR | F1 | F1@C | F1@PR | F1 |
| BLSTM | 78.80 | 88.61 | 83.71 | 77.04 | 91.04 | 84.04 | 77.39 | 89.33 | 83.36 |
| Syn-BLSTM | 82.74 | 90.53 | **86.63** | 79.37 | 91.43 | 85.40 | 82.81 | 91.15 | <u>86.98</u> |
| Dual-Syn-BLSTM | 81.58 | 89.70 | 85.64 | 79.86 | 91.31 | <u>85.59</u> | 82.32 | 90.81 | 86.56 |
| TK-SVM | 79.25 | 90.31 | 84.78 | 71.01 | 90.02 | 80.52 | 77.62 | 90.04 | 83.83 |
| Tree-LSTM | 71.79 | 76.67 | 74.23 | 72.22 | 82.88 | 77.55 | 77.09 | 83.17 | 80.13 |
| ResArg | 82.04 | 90.31 | 86.18 | 79.48 | 91.59 | 85.53 | 82.35 | 91.13 | 86.74 |
| ResAttArg | 82.27 | 90.11 | 86.19 | 80.72 | 91.79 | **86.26** | 83.74 | 91.27 | **87.51** |
| GNN | 80.72 | 89.97 | 85.34 | 77.14 | 90.98 | 84.06 | 80.78 | 90.56 | 85.67 |
| GNN + DiffPool | 81.17 | 90.06 | 85.61 | 77.09 | 91.03 | 84.06 | 81.25 | 90.58 | 85.91 |
| TC-GNN (STK) | 82.35 | 90.50 | <u>86.43</u> | 78.10 | 91.16 | 84.63 | 82.45 | 91.07 | 86.76 |
| TC-GNN (SSTK) | 80.97 | 89.90 | 85.44 | 77.25 | 91.06 | 84.15 | 81.24 | 90.60 | 85.92 |
| TC-GNN (PTK) | 81.90 | 90.30 | 86.10 | 77.17 | 91.00 | 84.08 | 81.74 | 90.70 | 86.22 |

### 3.3.6 Discussion

The positive results described so far corroborate our initial hypothesis that structured linguistic patterns are significant for identifying arguments in a sentence. In particular, TC-GNNs reach increased generalization capabilities over the non-regularized counterpart model while satisfying tree constraints to a certain degree. We hereby discuss: (i) how the GNN models can minimize the imposed regularizations; (ii) whether recognizable structured patterns can be extracted for each argumentative class; (iii) the overall introduced overhead.

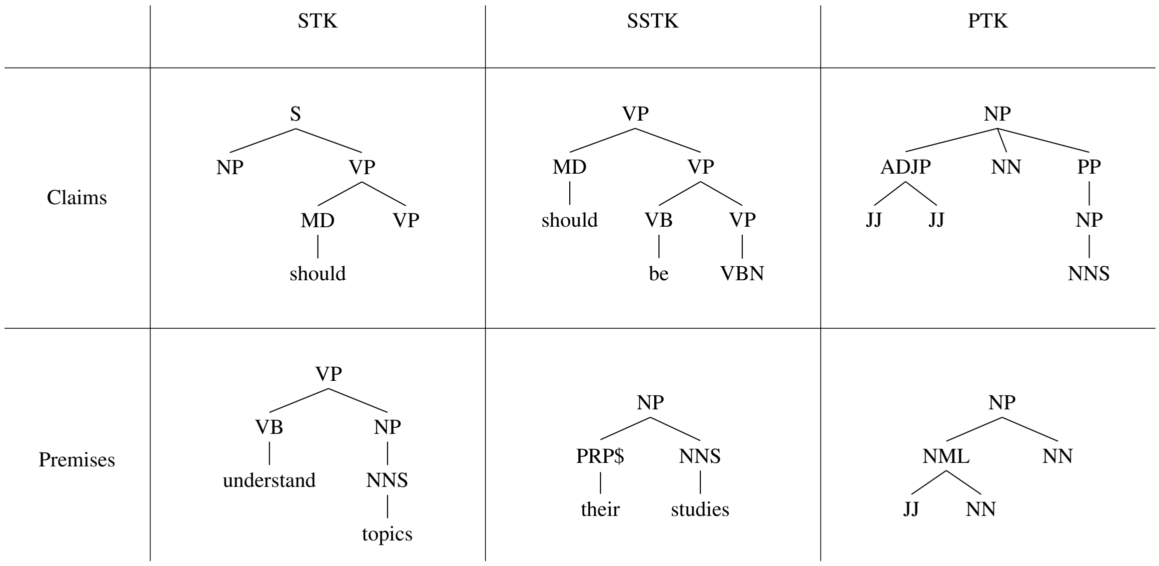


Figure 3.2: Some of the most frequent class-specific tree fragments extracted from TC-GNN models on the PE corpus (top: claims, bottom: premises).

First of all, we start by inspecting the difficulty of extracting tree fragments by regularizing the pooling layer of the GNN architecture. To do so, we compare the dual Lagrangian methodology with a fixed coefficient baseline, i.e., the $\lambda$ value for each constraint is initialized to a specific value and kept fixed throughout training. Table 3.7 reports classification performance and the average constraint value (the lower, the better) for claim detection in the IBM2015 corpus. We can see how using the dual Lagrangian method is crucial both to reach higher performance and achieve lower constraints violation. Moreover, an additional advantage of the approach relies on its simplicity and scalability. Fixed coefficients baseline require an extensive calibration phase to reach satisfying performance.

In particular, most of the evaluated configurations failed to reach significant performance. On the other hand, even by initializing all $\lambda$ values to 0, the dual Lagrangian method enforces constraints alignment and reduces the occurrence of degenerate scenarios where one of them is significantly violated. Further calibration of initial values might improve performance. Lastly, in the case of a deeper model with stacked pooling layers, the dual Lagrangian method scales linearly with the number of constraints and of pooling layers[4].

From a qualitative point of view, Figure 3.2 reports some of the most frequent tree fragments for each TC-GNN model on the PE corpus. In particular, we consider fragments unique to a specific argumentative class, i.e., are not shared with other classes. By doing so, we can inspect structured patterns that are valuable indicators of an argumentative component, e.g., a

[4]We consider a distinct set of multipliers for each pooling layer.

Table 3.6: Training and inference times on the IBM2015 corpus. We report the average batch time in seconds for training, computed over 50 epochs. The standard deviation is in brackets.

| Model | Training | Inference |
|---|---|---|
| GNN + DiffPool | 0.065 (0.004) | 0.046 (0.032) |
| TC-GNN (STK) | 0.075 (0.008) | 0.048 (0.077) |
| TC-GNN (SSTK) | 0.119 (0.010) | 0.042 (0.061) |
| TC-GNN (PTK) | 0.118 (0.009) | 0.060 (0.105) |

claim or a premise. Due to the soft pooling assignment, we apply an activation threshold to discriminate ‘active’ nodes. We set the activation threshold to 0.3 to include a large pool of fragments. Considering PTK fragments, we can see how the model focuses more on part-of-speech tags, i.e., lower levels of the constituency tree. On the other hand, for SSTK and STK, most fragments are centred near leaf nodes to satisfy kernel-specific constraints. In both cases, claims are mainly identified by expressions with modal verbs, such as ‘should’, whereas noun phrases often characterize premises.

We also inspect the added overhead of described tree regularizations for the base GNN model. Table 3.6 reports average training and inference times for all pooling based GNNs on the IBM2015 corpus. We considered the IBM2015 corpus as it has the highest number of samples, thus, representing a valuable benchmark setting. Compared to the GNN + DiffPool model, TC-GNNs have comparable training and inference times, meaning that the introduced regularization overhead is negligible.

We have shown that the extraction of structured knowledge in the form of sub-tree node clusters can be beneficial to the task of argument sentence detection. Additionally, we can use node pooling attention values to inspect the extracted sub-tree node clusters for analysis. Nonetheless, the extraction of sub-tree node clusters is non-trivial and subject to degenerate solutions if no adequate counter-measures are employed. The method we have presented in this work is just one possible solution to integrating structured features into a deep learning model.

The capability of extracting portions of an input parse tree or graph can be particularly beneficial for UKI. For instance, we could formulate a sophisticated knowledge integration process for textual patterns (Section 2.6.1). In particular, the knowledge mapping mechanism could compare textual patterns with given input texts both from the semantic and syntax-level point of view. To do so, approaches like TC-GNNs can be a valuable solution. The capability of extracting structured features opens to a wide variety of research directions. Some research directions we repute worth exploring are: (i) architecture extensions to handle graph-like inputs, (ii) multi-layered TC-GNNs, (iii) structured text similarity, (iv) text structure injection and (v) an online tool for AM. We briefly outline these research areas in the following sections.

Table 3.7: Constraint satisfaction for claim detection on the IBM2015 corpus. We report classification performance and the average constraint violation (the lower, the better). Standard deviation is reported in brackets.

| Kernel | Weights | F1@C | $C^{frag}$ |
|---|---|---|---|
| STK | Fixed | 11.8 | 0.62 (0.01) |
| STK | Lagrangian | 18.3 | **0.19** (0.01) |
| SSTK | Fixed | 17.2 | 0.70 (0.04) |
| SSTK | Lagrangian | 20.1 | **0.22** (0.02) |
| PTK | Fixed | 15.2 | 0.54 (0.03) |
| PTK | Lagrangian | 21.3 | **0.50** (0.28) |

### Extension to Graphs

The primary assumption behind the described TC-GNN models concerns tree-like structured inputs. Nonetheless, tree structures, such as constituency parse trees and dependency trees, have been widely adopted in NLP as they provide an abstract text representation [73, 101, 127, 148, 164]. The extension to graph inputs is compatible with TC-GNN constraints, provided that each pooled node cluster is at least a Directed Acyclic Graph (DAG) or a tree. Note that the method's applicability is not subject to the described constraints and can be employed for more graph-specific constraints.

Furthermore, the extension to graphs requires learning an adjacency matrix $A$ for each pooled node cluster. In particular, each pooled node cluster is also needed to satisfy the *Root Node*, *No Cycles* and *Root span* properties (Section 3.3.2). In other terms, we required the adjacency matrix $A$ to define a DAG or a tree in each pooled node cluster [296]. Then, it is possible to jointly retrieve pooled node clusters that follow the given tree kernel while overcoming graph topology issues.

Working at the graph level allows extending learnable structured features to graph structures like Abstract Meaning Representation (AMR) graphs. For instance, we could consider the union of constituency and dependency parse tree, i.e. nodes and edges of both trees are merged into a single unified representation, as a valuable graph structure for learning.

### Multi-layered TC-GNNs

A straightforward extension of TC-GNNs regards the complexity of their architecture. Similarly to other pooling layers, the tree-constrained pooling layer can be stacked repeatedly to define more complex TC-GNN models. As noted in Section 3.3.6, the main advantage of injecting constraints via the Dual Lagrangian framework [74] is that the complexity scales linearly with the number of pooling layers. In other terms, it is only necessary to consider a set of $\lambda$ multipliers for each tree-constrained pooling layer. It is worth exploring stacked TC-GNNs as the expected behaviour is structured feature composition. In particular, early layers capture node clusters of limited size while late layers learn how to compose the early ones to obtain more advanced node clusters.

### Structured Text Similarity

We could also employ the extracted structured features to compare text inputs for semantic and syntactic similarity. Recent work has explored efficient similarity evaluation methods via GNNs [149] or BERT-based models [213]. In this context, text similarity is generally formulated to compare two text inputs. Models trained for this task learn similar embedding representations for text inputs labelled as similar.

Similarly to the original application of TKs for argument detection [155], we aim at comparing text inputs using their extracted sub-tree structures. Then, graph matching mechanisms introduced for GNNs [149] can be employed to compare pooled node clusters belonging to two different text inputs. More precisely, two text inputs are similar based on the number of shared pooled node clusters.

Such an approach assumes that texts with similar structures, as well as similar semantic meaning, should be labelled as being similar. Note that, in the case of abstract representations like constituency parse trees, the content of pooled node clusters can be independent of the semantic of the given input text. For instance, a pooled node cluster can only describe a sub-tree comprised of POS tags. In this case, the extracted structured feature can be shared across texts of different meanings, highlighting a syntactic pattern.

Indeed, in the context of AM, arguments and their components have been frequently investigated to extrapolate descriptive patterns [52]. Nonetheless, depending on the given context, the same text can be labelled as a different argument component [35]. Thus, AM stands as a particularly challenging domain for knowledge extraction as the knowledge itself is strongly affected by contextual information. In other terms, the set of features that characterizes a given argument component is highly dependent on (i) the domain, (ii) the argument definition itself and (iii) the given textual context.

### Structure injection

We could employ the extracted structured features in a generative setting as well. Conditional generation based on a set of attributes has been widely explored in recent years [147, 295]. The underlying concept is injecting a conditioning factor within the generative process to control the generation process and ensure desired properties. A notable example is given by style transfer in NLP [118].

In the context of AM, we consider the structured features as the conditioning factor of the generation process. We are mainly interested in affecting the syntactic structure of a text in conjunction with its semantic meaning. Alternatively, it is also worth exploring the capability of solely modifying the text structure without affecting its meaning. Indeed, in natural language, we can express the same concept differently. Such an approach could be beneficial to a wide variety of tasks in AM [142], such as argument style transfer, argument completion and argument quality improvement.

In the first, structure injection is applied to modify a text input via a given conditioning factor. For instance, we might want to paraphrase a given argument by emulating the style of another one. Such style information is stored in extracted pooled node clusters as our structured feature set. Argument completion is particularly relevant in domains subject to noisy and jargon expressions like social media and online blogs.

The difficulties regarding AM in these scenarios are further enhanced by arguments with missing information like enthymemes [230]. In this domain, one valuable objective follows the completion of arguments by injecting structured information.

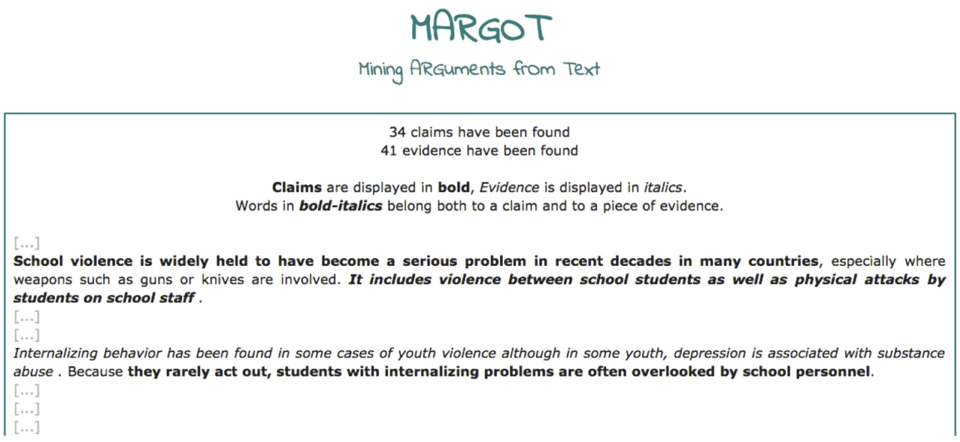
As with argument completion, the integration of structured knowledge is of particular relevance for argument correction. The problem is formulated as modifying an original argument to achieve a task-specific goal. One such goal is the quality of the argument. For instance, argument quality may refer to its persuasiveness for a given topic or the capability of an argument detection model to identify it with high confidence.

#### An Online Tool for Argument Mining

We aim at extending the Mining ARGuments frOm Text (MARGOT) tool [157] to integrate the capabilities offered by TC-GNNs for several purposes. MARGOT allow users to extract argument components from provided text documents quickly. Figure 3.3 provides a concrete example of the MARGOT online tool. First, we integrate pre-trained TC-GNN models as an online service. Second, we plan to cover various applications like argument similarity and argument generation. The general purpose of such a suite of services is to offer to the research community and non-expert users easily accessible argumentative annotations.

## 3.4 Arguments for Scientific Dialogues

We consider the task of defining conversational agents on scientific papers. In this setting, dialogue partners exchange information about a scientific paper to reach a common goal. We



34 claims have been found
41 evidence have been found

**Claims** are displayed in **bold**, *Evidence* is displayed in *italics*.
Words in ***bold-italics*** belong both to a claim and to a piece of evidence.

[...]
**School violence is widely held to have become a serious problem in recent decades in many countries**, especially where weapons such as guns or knives are involved. ***It includes violence between school students as well as physical attacks by students on school staff*** .
[...]
[...]
*Internalizing behavior has been found in some cases of youth violence although in some youth, depression is associated with substance abuse* . Because **they rarely act out, students with internalizing problems are often overlooked by school personnel**.
[...]
[...]
[...]

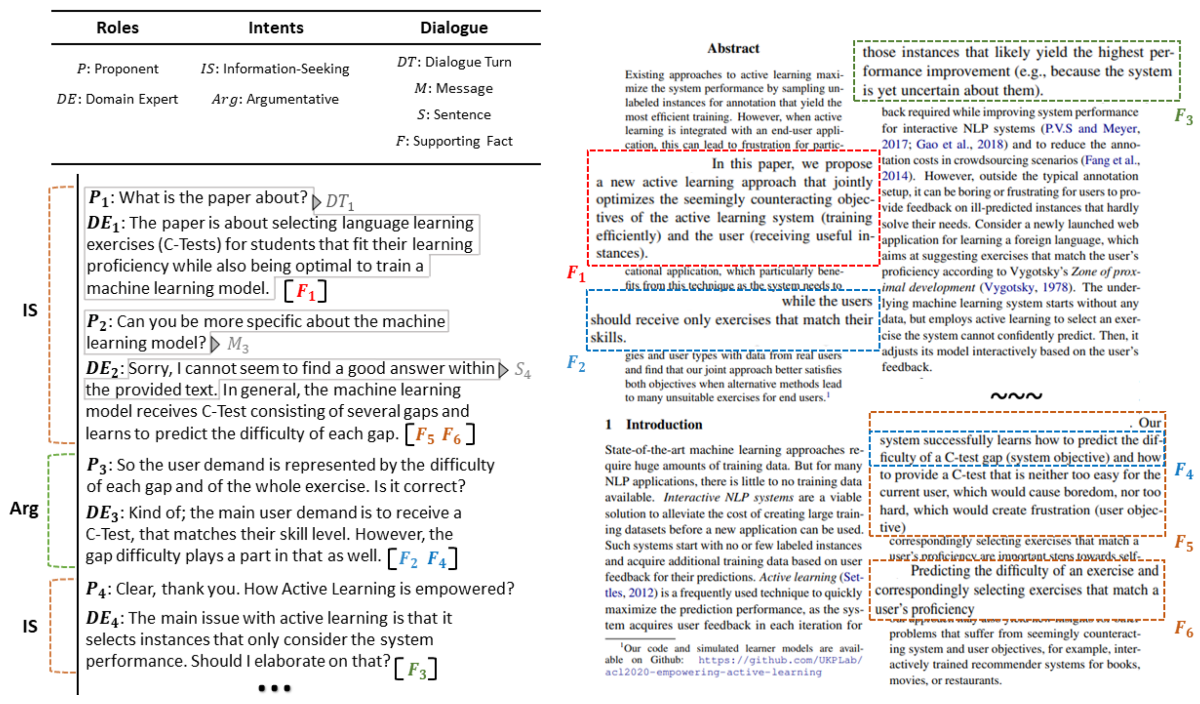
Figure 3.3: An example of execution of MARGOT. Detected claims are in bold, while evidence are displayed in italics. Sentences that both contain a claim and a evidence are in bold italics.

identify two dialogue roles, the Domain Expert (DE) and the Proponent (P). Dialogue partners sustain a conversation to convey the content of the paper from one DE to P. Differently from DE, P has no access to the scientific paper. Thus, P has to rely on DE to acquire information. In this experimental setting, we evaluate conversational agents on two specific tasks: (i) *DE message generation*, i.e. the capability of generating a text in response to a P's dialogue message; (ii) *supporting fact selection*, i.e. the capability of selecting text spans (supporting facts) in scientific papers to support the DE's generated response.

Scientific papers are rich in argumentative content due to their inherent high persuasive intent [95, 129]. Therefore, we investigate how automatically detected arguments can aid the conversational agent in addressing the tasks mentioned above. In particular, the lack of adequate scientific argumentative data enforces the appliance of pre-trained argumentative models to extract the argumentative graph of a scientific paper. More precisely, we define the argumentative graph as the graph where detected arguments stand as nodes and their inferred relations stand as edges.

Such structured information is of particular importance given the overall organization of scientific papers as it bridges connections between texts independently of their spatial location in the paper. More precisely, concepts in scientific papers are presented over multiple paper sections with incremental details. Thus, the concepts and contributions of the paper follow a fragmentary organization throughout the paper. We view the extracted argumentative graph as structured knowledge that provides high-level information about a scientific paper. The conversational agent then employs such added knowledge to address its tasks better.

We briefly describe our data collection framework for acquiring dialogues on scientific papers. In particular, several challenges arise since adequate expertise is required to sustain dialogues on scientific papers. For instance, sufficiently motivating participants to produce


| Roles | Intents | Dialogue |
|---|---|---|
| $P$: Proponent | $IS$: Information-Seeking | $DT$: Dialogue Turn |
| $DE$: Domain Expert | $Arg$: Argumentative | $M$: Message |
| | | $S$: Sentence |
| | | $F$: Supporting Fact |

IS

$P_1$: What is the paper about? $DT_1$

$DE_1$: The paper is about selecting language learning exercises (C-Tests) for students that fit their learning proficiency while also being optimal to train a machine learning model. $[F_1]$

$P_2$: Can you be more specific about the machine learning model? $M_3$

$DE_2$: Sorry, I cannot seem to find a good answer within the provided text. $S_4$ In general, the machine learning model receives C-Test consisting of several gaps and learns to predict the difficulty of each gap. $[F_5\ F_6]$

Arg

$P_3$: So the user demand is represented by the difficulty of each gap and of the whole exercise. Is it correct?

$DE_3$: Kind of; the main user demand is to receive a C-Test, that matches their skill level. However, the gap difficulty plays a part in that as well. $[F_2\ F_4]$

IS

$P_4$: Clear, thank you. How Active Learning is empowered?

$DE_4$: The main issue with active learning is that it selects instances that only consider the system performance. Should I elaborate on that? $[F_3]$

•••

**Abstract**

Existing approaches to active learning maximize the system performance by sampling unlabeled instances for annotation that yield the most efficient training. However, when active learning is integrated with an end-user application, this can lead to frustration for partic-

$F_1$: In this paper, we propose a new active learning approach that jointly optimizes the seemingly counteracting objectives of the active learning system (training efficiently) and the user (receiving useful instances).

cational application, which particularly benefits from this technique as the system needs to

$F_2$: while the users should receive only exercises that match their skills.

gies and user types with data from real users and find that our joint approach better satisfies both objectives when alternative methods lead to many unsuitable exercises for end users.[1]

**1 Introduction**

State-of-the-art machine learning approaches require huge amounts of training data. But for many NLP applications, there is little to no training data available. *Interactive NLP systems* are a viable solution to alleviate the cost of creating large training datasets before a new application can be used. Such systems start with no or few labeled instances and acquire additional training data based on user feedback for their predictions. *Active learning* (Settles, 2012) is a frequently used technique to quickly maximize the prediction performance, as the system acquires user feedback in each iteration for

[1]Our code and simulated learner models are available on Github: https://github.com/UKPLab/acl2020-empowering-active-learning

$F_3$: those instances that likely yield the highest performance improvement (e.g., because the system is yet uncertain about them).

back required while improving system performance for interactive NLP systems (P.V.S and Meyer, 2017; Gao et al., 2018) and to reduce the annotation costs in crowdsourcing scenarios (Fang et al., 2014). However, outside the typical annotation setup, it can be boring or frustrating for users to provide feedback on ill-predicted instances that hardly solve their needs. Consider a newly launched web application for learning a foreign language, which aims at suggesting exercises that match the user's proficiency according to Vygotsky's *Zone of proximal development* (Vygotsky, 1978). The underlying machine learning system starts without any data, but employs active learning to select an exercise the system cannot confidently predict. Then, it adjusts its model interactively based on the user's feedback.

~~~

$F_5$: . Our $F_4$: system successfully learns how to predict the difficulty of a C-test gap (system objective) and how to provide a C-test that is neither too easy for the current user, which would cause boredom, nor too hard, which would create frustration (user objective)

correspondingly selecting exercises that match a user's proficiency are important steps towards self-

$F_6$: Predicting the difficulty of an exercise and correspondingly selecting exercises that match a user's proficiency

problems that suffer from seemingly counteracting system and user objectives, for example, interactively trained recommender systems for books, movies, or restaurants.

Figure 3.4: An example dialogue from our ArgSciChat dataset. Supporting facts $\{F_1, \dots, F_6\}$ are highlighted in the scientific paper. Dialogue partners alternate between information-seeking (IS) and argumentative (Arg) intents.

dialogues given their tight schedules is a demanding and non-negligible challenge. We provide an in-depth analysis of our collected conversational dataset, namely *ArgSciChat*. In particular, we overview acquired dialogues and present a qualitative study regarding automatically retrieved arguments. Lastly, we report experimental results regarding `DE` *message generation* and *supporting fact selection* tasks. We evaluate the efficacy of automatically extracted arguments when no information about supporting facts is provided. More details are reported in Ruggeri et al. [226].

## 3.4.1 Dataset Collection

In this section, we first formulate the dialogues we aim to collect. Then we propose our data collection framework to acquire dialogues, a multi-step pipeline that is optimized to allow alignment between participants. We provide details about our implementation in Appendix B.

### Dialogue Formulation

We aim at collecting argumentative and information-seeking dialogues between scientists. We define two roles for participants: `DE` and `P`. `P` starts a dialogue with `DE` on a scientific paper of
~~~

Table 3.8: The set of actions we define to ensure dialogues contain information-seeking (IS) and argumentative (Arg) interactions.

| | |
|---|---|
| | **IS** |
| Ask Info (AI) | A question regarding the paper to know more about its content. |
| Reply Info (RI) | A reply that provides information in response to an initial request. |
| Ask Suggestion (S) | A request or proposal regarding which topic to discuss next. |
| | **Arg** |
| Give Opinion (GO) | A writer's opinion about some paper content. |
| Ask Rebuttal (AR) | A request to the other dialogue partner to express their own opinion on a particular subject. |

which `P` only knows the title. `DE` is given the abstract and introduction sections of the same paper. Thus, `P` is encouraged to formulate initial questions to learn more about the paper. `P` and `DE` also argue about the claims in the paper. We limit paper content to abstract and introduction sections to limit participants' effort to act as `DE` while providing enough information to sustain a dialogue. We also define a set of actions for dialogue partners to ensure their interactions follow IS and Arg intents. As shown in Table 3.8, these actions are grouped by their intent type. We instruct `DE` to select up to two supporting facts to ground their messages. Lastly, we instruct `DE` to provide suggestions about available paper content to limit the occurrences of unanswerable questions.

### Notation

We define as a *message* each text that a dialogue partner generates. An example of a message is $\mathtt{P}_2$ as shown in Figure 3.4. A message can contain one or more *sentences*. Each sentence is associated with a dialogue action (Table 3.8). Lastly, we define a *dialogue turn* as a pair of adjacent `P` and `DE` messages. For instance, the pair ($\mathtt{P}_1$, $\mathtt{DE}_1$) as shown in Figure 3.4 is a dialogue turn.

### Data Collection Framework

We define a novel framework to collect dialogues between scientists as domain experts. The well-used crowdsourcing frameworks like AMT and Upwork are not suitable for dialogue collection in expert domains due to their shortcoming in (1) letting experts suggest and choose the topics of dialogue, which helps in encouraging experts to participate in a conversation, and (2) being flexible with the time slots in which experts are willing to participate in a conversation, which is required as experts have a limited time budget, and (3) supporting synchronous participation, which is required to generate natural dialogues.

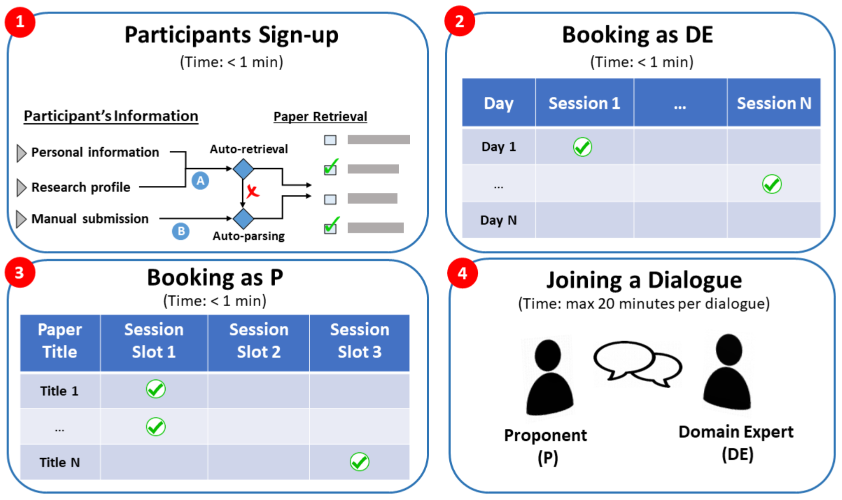


Figure 3.5: The data collection framework we propose to acquire argumentative scientific dialogues.

Our framework consists of four major steps (Figure 3.5). The step *"Participant Sign-up"* allows domain experts to sign up in the system and propose the topics at which they are experts. The step *"Booking as DE"* lets experts propose the time slots at which they are available to participate in a dialogue with a DE role. The step *"Booking as P"* lets experts choose the topics about which they are willing to learn and also the time slots at which they can participate with a P role. Finally, step *"Joining a Dialogue"* provides a synchronous online session in which two experts chat about the topic they agreed on. We describe the method behind each step in the following paragraphs and report their implementation details in Appendix B.

**Participants Sign-up** Initially, participants have to fill out a sign-up form. The form consists of two sections. In the first section, we ask for participants' personal information. The second section allows participants to suggest their papers to be used in a dialogue. Participants' most recent papers are automatically retrieved to save participants' time. Participants then have to select a few of them as the topics they are experts on. Participants can also submit their papers manually to include topics other than what we retrieved.

**Booking as DE** Upon papers confirmation, as the next step, we ask participants to select a *dialogue session* for each selected paper. We refer to these sessions as "DE dialogue session" as participants agree to take the DE role on that paper.

**Booking as P** Participants have to book dialogue sessions in which they should act as P. Consequently, participants are provided with summary information about all DE dialogue sessions. To collect several dialogues for each paper, we split each DE dialogue session into smaller slots. Participants then select a few slots to match with a dialogue partner. This procedure provides flexibility for participants to decide on the topic of dialogues while ensuring enough coverage of scientific papers.

**Joining a Dialogue** Our framework notifies participants before each dialogue slot begins. The goal is to maximize the number of dialogue slots in which two participants, acting as `P` and `DE` respectively, are present to talk about a paper.

#### Collecting Dialogues about NLP Papers

This study uses our framework to collect dialogues about NLP papers. While this choice opens a common background between participants, it also brings enough topic diversity for dialogues. We invite senior and junior researchers from two large NLP groups in Europe as a candidate pool for participation. We organize the data collection phase into multiple short data collection rounds. Such short term rounds encourage scientists participation due to their limited time budget and facilitate participants alignment. Dialogues were collected within two distinct data collection rounds over two weeks. The dialogues collected in these two rounds construct our ArgSciChat dataset.

### 3.4.2 Analysis

We carry out extensive analysis on (i) data collection framework; (ii) collected dialogues; (iii) action set, and (iv) arguments.

#### Framework Analysis

Table 3.9 reports general statistics of the effectiveness of the data collection framework. At the end of the last data collection round, we identified 68 dialogues. We filtered out dialogues that were too short, i.e., had less than eight messages or ended abruptly. At the end of this process, we obtained 41 dialogues. Considering the number of participants and collected dialogues given the short data collection period, we notice that our framework successfully encourages the participation of scientists in the study. The devised short data collection rounds (Section 3.4.1) impose a limited workload to participants, while the lightweight data collection pipeline lifts the burden of participants alignment.

Table 3.9: Data collection framework statistics concerning completed dialogue sessions.

| Property | Value |
|---|---|
| # participants | 23 |
| # scientific papers | 20 |
| # dialogues | 41 |
| # messages | 498 |
| # sentences | 1034 |

### Dialogues Analysis

In this section, we elaborate on collected dialogues. First, we compare ArgSciChat with related datasets. Second, we investigate to what extent our dialogue formulation yields diversified dialogues on the same paper. Third, we analyze role-specific behaviours concerning our action set.

**Comparison with other datasets** Table 3.10 reports detailed statistics about collected dialogues. We notice that dialogues in ArgSciChat are comparable in dialogue turns with other conversational datasets. However, messages in ArgScichat are far more articulated. This property is reflected by the higher percentage of multi-sentence messages and the higher message token length. Examples of multi-sentence messages are $\texttt{P}_4$ and $\texttt{DE}_4$ as shown in Figure 3.4.

**Dialogue Diversity** We investigate if dialogues on the same paper present low diversity due to the limited paper content. Indeed, our data collection formulation allows various dialogues on the same paper. On average, ArgSciChat contains two dialogues per scientific paper. We evaluate if messages belonging to dialogues on the same paper show high textual similarity as a diversity measure. The higher the rate of similar messages, the lower is the dialogue diversity.

We define some indicators to measure dialogue diversity: (i) similar P questions percentage; (ii) similar DE messages given a question cluster; (iii) DE message to supporting fact similarity. These indicators are chosen to reflect the two main behaviours of our dialogues: a) dialogue intents (IS and Arg) and b) DE supporting fact selection.

As acquiring experts to annotate the dialogues is expensive, we rely on an automatic analysis via SentenceBERT [213][5]. In particular, we consider the cosine similarity between encoded sentences as the similarity measure to compute the described indicators. Figure 3.6 reports results for various similarity thresholds. In particular, we report indicators as the average frequency of similarity matches based on the given similarity threshold.

[5]We use `all-mpnet-base-v2` as the current state of the start model on sentence similarity.

We initially identify sentences from P messages as questions. Then, we apply the Sentence-BERT model to compute the similarity between pairs of questions from dialogues regarding the same scientific paper. Similarities are rounded to $0, 1$ binary values given a fixed similarity threshold. A pair match is then defined when the rounded similarity value equals 1. The first indicator is computed as the average percentage of unique pair matches over all papers with multiple dialogues.

Pair matches are also employed to extract question clusters. We define a question cluster as a group of questions that report mutual matching. For instance, if question A is matched to question B and the latter is matched to question C, then $\mathtt{A}, \mathtt{B}, \mathtt{C}$ defines a question cluster. From each identified question cluster, we retrieve the corresponding cluster of DE messages. We then apply the SentenceBERT model to compute the similarity between pairs of DE messages belonging to the same cluster. The second indicator is computed as the average percentage of unique pair matches over all papers with multiple dialogues.

Lastly, we apply the SentenceBERT model to compute the similarity between each DE message and its corresponding supporting facts. Then, we compute pair matches as done for the other indicators by fixing the similarity threshold. The third indicator is then computed as the average percentage of pair matches over all DE messages.

We observe that all indicators show moderate to low frequencies as the similarity threshold increases. In particular, even at very low similarity threshold values, we notice that about 35% of DE messages, 60% of messages to facts and 10% of P messages are similar. Such similarity rates decrease as the similarity threshold increases. This behaviour is mainly attributed to the high rate of multi-sentence messages that hinders high similarity values. Although an annotation study is recommended, these results suggest that our dialogue formulation and action set provide enough flexibility to foster diversified dialogues. Such property encourages data collection even when a limited amount of papers is considered.

**Role-specific Behaviours** Moreover, we analyse whether dialogue roles in collected dialogues reflect our dialogue formulation. First, we study if P interactions with DE follow multiple intents. Second, we evaluate DE fact selection.

Regarding P interactions, we focus on P questions. Table 3.11 reports the occurrence of each question starting tri-grams over 293 identified questions. We can distinguish expressions that reflect request-type actions in our action set (Table 3.8). Notable examples are *{(what is the), (what kind of)}* for the Ask Info action, *{(would you like)}* for Ask Suggestion, and *{(do you think), (how do you)}* for Ask Opinion. Thus, we can see how P questions both reflect the IS and the Arg dialogue intents.

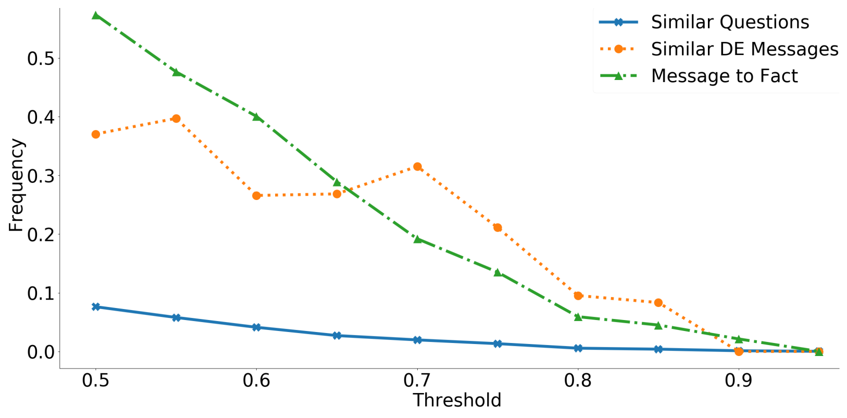


Figure 3.6: Dialogue diversity indicators based on SentenceBert.

Table 3.10: Dialogue-level statistics of ArgSciChat. With a slight abuse of notation, we consider sequential QA as a form of dialogue to include QA datasets like CoQA.

| Dataset | Turns | Multi sentence messages | Message token length | Sentence token length |
|---|---|---|---|---|
| CoQA | 15.5 | 0.2% | 4.7 | 4.7 |
| QuAC | 7.3 | 4.0% | 11.4 | 10.9 |
| Doc2Dial | 6.4 | 17.8% | 16.3 | 13.5 |
| ArgSciChat (ours) | 6.3 | 50.8% | 38.2 | 12.1 |

Concerning DE, Table 3.12 reports relevant statistics about fact selection. The high rate of 2-fact DE messages indicates that dialogue turns are articulated and that multiple paper text spans are used to provide exhaustive content to P. This property is further stressed out when considering the fact selection distribution over paper sections and the rate of fragmented facts, i.e. facts from the same DE message that belong to different paper sections. $DE_3$ from Figure 3.4 provides an example of a message with fragmented facts.

Furthermore, the limited paper content does not significantly hinder dialogue partners interactions. Out of 212 DE messages, 10.4% of them are associated with an unanswerable question. Compared to QuAC (20.2%), the low percentage is due to how DE is instructed to provide suggestions based on available paper content to guide the dialogue. One such example is represented by $DE_2$ as shown in Figure 3.4.

### Action Set Analysis

We evaluate dialogue roles based on our action set. We carry out a manual annotation study of ArgSciChat dialogues to do so. Compared to dialogue analysis (Section 3.4.2), annotating dialogues with dialogue actions is a low-demanding task as no in-depth understanding of paper

Table 3.11: Fifteen most frequent tri-grams question starters in ArgSciChat.

| | | | | | |
|---|---|---|---|---|---|
| 1. | do you want | 6. | can you tell | 11. | how do you |
| 2. | do you think | 7. | would you like | 12. | which kind of |
| 3. | what is the | 8. | can you give | 13. | what type of |
| 4. | what do you | 9. | do you have | 14. | how do the |
| 5. | what is this | 10. | what kind of | 15. | does it make |

Table 3.12: ArgSciChat statistics regarding fact selection.

| Property | Value |
|---|---|
| 1-Fact Turns | 129 |
| 2-Fact Turns | 82 |
| Avg Facts | 1.4 |
| Avg Fact Distance | 5.8 |
| % of Title Facts | 1.0% |
| % of Abstract Facts | 38.2% |
| % of Introduction Facts | 60.8% |
| % of Fragmented Facts | 15% |

content is required. We show that actions are easily recognizable and, thus, widely applicable to other domains.

**Manual Annotation** We manually annotate dialogue messages at sentence level using our action set (Section 3.4.1). We consider four NLP practitioners as annotators. We use GLOSS[6], a lightweight text annotation tool, to collect annotations. The inter-annotator agreement is computed on a random set of 10 dialogues. Table 3.13 reports Cohen's Kappa for each annotator pair. The inter-annotator agreement is computed by transferring sentence-level annotator labels at the token level. The average Cohen's Kappa is 0.89, and the Krippendorf's alpha is 0.83. These results indicate a high inter-annotator agreement. Such high agreement corroborates our hypothesis that provided actions are fairly distinguishable in dialogues in our dataset.

**Dialogue Roles** Table 3.14 reports actions distribution per role computed over the whole dataset. We notice that specific role behaviours are reflected in the action frequencies. For instance, `P` is more inclined to ask questions (AI), whereas `DE` reports about facts from paper content (RI). Additionally, by grouping up actions according to their corresponding dialogue

[6]https://github.com/jsavelka/gloss-server

Table 3.13: Cohen's kappa computed for each annotator pair on ArgSciChat.

| | **A1** | **A2** | **A3** | **A4** |
|---|---|---|---|---|
| **A1** | - | 0.94 | 0.88 | 0.92 |
| **A2** | 0.94 | - | 0.87 | 0.89 |
| **A3** | 0.88 | 0.87 | - | 0.86 |
| **A4** | 0.92 | 0.89 | 0.86 | - |

Table 3.14: Action set distribution on ArgSciChat.

| | AI | S | RI | GO | AR | IS | ARG |
|---|---|---|---|---|---|---|---|
| P | 49.2 | 0.9 | 11.3 | 17.2 | 4.7 | 61.4 | 21.9 |
| DE | 0.8 | 6.6 | 63.9 | 10.2 | 5.9 | 71.3 | 16.1 |
| All | 24.3 | 3.8 | 38.4 | 13.6 | 5.4 | 66.5 | 18.9 |

intent, we observe that the argumentative intent represents nearly 20% of each dialogue. This fact indicates that dialogue partners exchange information and spend time arguing about it.

### Argumentative Analysis

We investigate to what extent selected supporting facts are also argumentative. We use pre-trained models to automatically detect arguments from papers due to the high cost of manual annotation. We then evaluate the applicability of an argument-based fact selection.

**Argument Extraction** We employ two argumentative models to detect arguments and links from a scientific paper automatically. This process defines an argumentative graph. Such structured representation is precious for fact selection due to linking fragmented text spans. We apply two SciBERT models [11] for argument detection and argument linking, following a two-step pipeline as in [174]. The SciBERT models are trained on the DrInventor [141] dataset as this dataset is the largest with argumentative annotations on 40 scientific papers. We formulate argument detection as a sequential tagging task alike [140]. We follow the same training setup of [140]. On the other hand, argument linking is viewed as a binary classification problem as described in [79]. In contrast to [79], we consider a more simplified formulation. More precisely, we only consider the task of binary link prediction, i.e. determining whether two arguments are linked or not, rather than a multi-task scenario. In the latter, the model is trained to identify both arguments and determine the presence of a relation and the corresponding relation type.

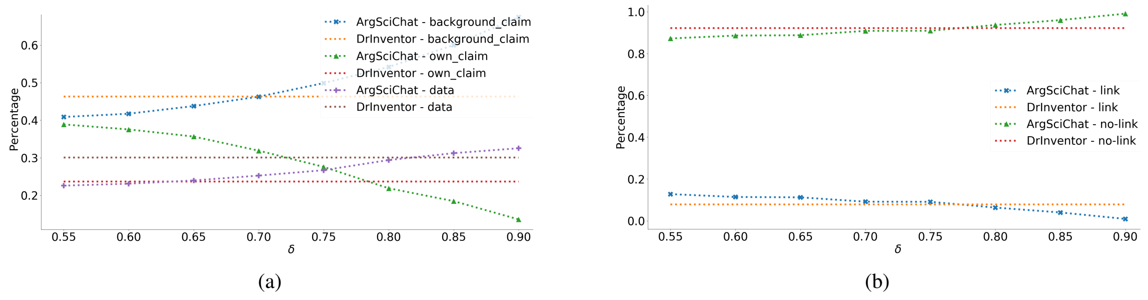


Figure 3.7: Argumentative models class distribution for (a) the task of argument component detection and (b) argument linking when varying the confidence threshold $\delta$.

Table 3.15: Model performance on Argument Mining tasks. We report Token-F1 for sequential argument component detection and Link-F1 for argument linking.

| Model | Token-F1 | Link-F1 |
|---|---|---|
| SciBERT | 62.6 | 36.9 |
| SOTA | 44.7 [140] | 43.7 [79] |

Table 3.15 reports results on argument detection and argument linking tasks regarding model evaluation. Our models work on par with or outperform the state of the art (SOTA) in both tasks. As evaluation metrics, we employ token-level F1-score, i.e. *Token-F1*, for argument detection and binary F1-score, i.e. *Link-F1*, for argument linking. As expected, the simplified argument linking formulation achieves lower performance than the SOTA model. We opted for a streamlined model for our preliminary study.

Since ArgSciChat scientific papers are not provided with argument annotations, we rely on DrInventor to control the degree of the automatic annotation process. Figure 3.7a and 3.7b report performance results regarding the two datasets. DrInventor argumentative schema distinguishes among *background_claim*, *own_claim* and *data*. These annotations are at the sub-sentence level, where links could exist within a single sentence. We notice that in the $[0.70, 0.75]$ confidence threshold interval, there's a high similarity between class distributions in the two datasets for both argumentative tasks. Lastly, we apply these SciBERT models to annotated ArgSciChat scientific papers.

**Arguments Evaluation** Figure 3.8 reports statistics concerning the quality of inferred argumentative graphs when varying the model prediction confidence threshold $\delta$. For values of $\delta \in [0.70, 0.75]$, around 40-60% of facts are argumentative, and 20-15% are linked on average per dialogue. These graphs have less than $\sim$30% of isolated nodes while having limited edge intensity [194] ($\sim$10% of connected node pairs). These results and, in particular, the high rate of argumentative facts suggest that we could employ arguments to guide fact selection.

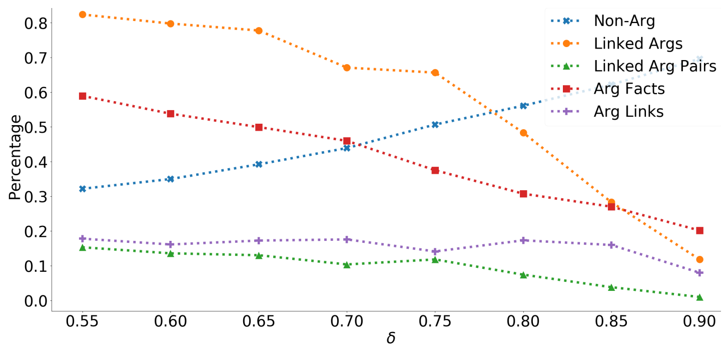


Figure 3.8: Argumentative graph measures when varying the confidence threshold $\delta$. Arguments (Arg) and facts are considered at the sentence level.

### 3.4.3 Experiments

In this section, we investigate how our dataset might contribute to building an expert agent for NLP paper comprehension. In particular, we are interested to know how well the agent's responses to P's messages are similar to DE's messages in the ArgSciChat dataset. Since the dataset also contains facts from a paper that are used to generate a response, we study to what degree a conversational agent retrieves facts similar to those in the dataset.

We follow QASPER to setup this experiment. We use LED [12] to respond to a P's message which is taken as *query*. We consider four settings where the agent is given (i) the input query ($Q$); (ii) the query and the scientific paper ($Q$, $P$); (iii) the query, the scientific paper and the dialogue history ($Q$, $P$, $H$), i.e. the sequence of all previous dialogue turns; (iv) the query and the supporting facts ($Q$, $F$). For fact selection, we only consider input combinations (ii) and (iii) as the scientific paper content is required.

We employ a 5-fold repeated cross-validation evaluation routine to account for model variance issues. We set the number of repetitions to three as in [51]. Each fold divides ArgSciChat into train (80%), validation (10%) and test (10%) splits. We perform splits at the scientific paper level to limit dialogue partners and paper content biases.

Given the presence of multi-sentence messages, we consider two dataset versions: *Original* (Orig) and *Sentence-split* (Split). Figure 3.9 describes the task formulation in both datasets. We consider the Split version as it (i) represents a relaxation of the response generation task; (ii) is the upper bound of a pipeline-based approach where $\tilde{S}_{i,DE}$ replaces $S_{i,DE}$ as input; (iii) is compliant with our action set as action labels are provided at sentence-level (Section 3.4.2). In particular, each input set in the Split dataset represents an individual sample. At inference time, sentence-level predictions are merged for comparison with the Orig dataset. Table 3.16 reports ArgSciChat statistics for Orig and Split dataset versions.

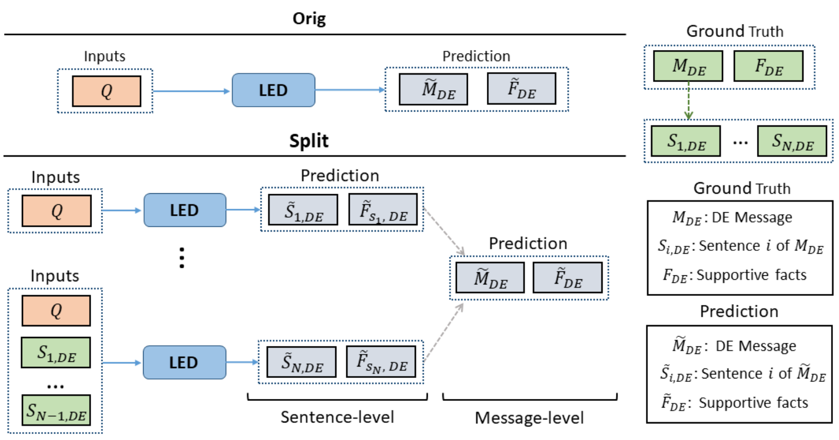


Figure 3.9: Problem setup for Orig and Split dataset versions of ArgSciChat. Additional inputs are omitted for clarity.

Table 3.16: Samples amount for Orig and Split (in brackets) dataset versions of ArgSciChat.

| Split | Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5 |
|---|---|---|---|---|---|
| Train | 199 (338) | 200 (371) | 183 (321) | 212 (376) | 207 (376) |
| Validation | 27 (63) | 14 (32) | 39 (74) | 27 (57) | 22 (40) |
| Test | 23 (53) | 35 (51) | 27 (59) | 10 (21) | 20 (38) |

Following [51], we report token-level F1-score, i.e. *Message-F1*, for dialogue generation, and sentence-level F1-score, i.e. *Fact-F1*, for fact selection. Additionally, we apply early stopping training regularization on each task reference metric with patience set to 20 epochs. We set 50 as the maximum number of epochs. We evaluate LED models for fact selection with direct supervision or leveraging argument annotations. Then, we report LED performance for the response generation task.

### Fact Selection

As in [51], we consider the fact selection supervision loss, $\mathscr{L}_{fact}$. In particular, the LED model is jointly trained to generate DE messages and select corresponding facts. We consider Tf-Idf and SentenceBert (S-BERT) models as baselines for comparison. Additionally, we compute a human-lower bound as in [51] with three NLP experts. Table 3.17 reports model performance on the fact selection task. We observe that ($Q$, $P$) is the best performing input combination on both data splits. Furthermore, we notice that models reach higher performance on the Split dataset. This behaviour might be motivated due to facts being associated with specific message sentences. In particular, the LED model with ($Q$, $P$) input combination is the best performing model with 14.24 Fact-F1 on the test set.

Regarding LED models trained on the Split dataset, we distinguish between *Best* and *Union* fact selection merging strategies. Compared to the Orig dataset, sentence-level model predictions have to be merged into a single prediction. E[51], each model prediction is a binary mask of size equal to the number of fact candidates. More precisely, a candidate predicted as 1 corresponds to a supporting fact. In the first scenario, we consider sentence-level predictions and determine the best performing mask as a message-level merging criterion. Conversely, sentence-level fact masks are summed up together in the second scenario. Table 3.18 reports LED model performance when considering these two merging strategies.

Based on the results of the argumentative analysis (Section 3.4.2), we replace $\mathscr{L}_{facts}$ with an argument detection objective, i.e. $\mathscr{L}_{arg}$. Facts are then retrieved by selecting the arguments that have the highest similarity with the query. We experiment with $\delta \in \{0.70, 0.75\}$ for argument annotation. As shown in Table 3.17, LED models with the argument-based fact selection show comparable or improved performance up to $\approx 1-2$ Fact-F1 points on the validation split. The only exception is represented by (*Q, P*) on the Split dataset. Such results are particularly valuable as no direct fact selection supervision is provided while less than 58% of paper sentences are considered on average.

On the other hand, S-BERT and Tf-Idf baselines show reversed behaviours when considering argument-based fact selection. This result is further accentuated when we limit the search space of these baselines to the argumentative graph. In particular, we consider facts from the available dialogue history as the initial set of candidates for fact selection. Then, we iteratively include arguments related to the current set of candidates via the argumentative graph. Thus, the efficacy of arguments is tightly dependent on how text similarity is defined.

We observe that the graph-based Tf-Idf baseline greatly improves over its argumentative counterpart and shows comparable performance with its standard counterpart. Such results are particularly valuable as the graph-based baselines search space uses $\approx 15-40\%$ of paper sentences and $\approx 26-59\%$ of argumentative sentences. Table 3.19 reports graph baseline performance with an increasing number of iterations. In particular, each iteration corresponds to adding new arguments as potential candidates by considering the neighbourhood of each current candidate fact.

In the first iteration $i = 0$, the initial set of candidates for fact selection is the set of supporting facts in the dialogue history. If the set of candidates is empty, we consider the whole set of arguments for fact selection like their non-graph argumentative counterparts. For the next iteration $i+1$, the set of candidates consists of the set of candidates of iteration $i$ plus the neighbourhood of each candidate according to the argumentative graph. Additionally, Table 3.20 reports argumentative graph statistics for graph argument-based fact selection baselines. In particular, we evaluate to what extent the argumentative graph is used for fact selection. We

Table 3.17: Experimental results and human lower-bound on supporting fact selection. We report the best sentence-level performance for models trained on the Split dataset. We report the best graph iteration performance for argumentative graph-based baselines.

| Model | Fact-F1 | | | | | |
|---|---|---|---|---|---|---|
| | $\mathscr{L}_{fact}$ | | $\mathscr{L}_{arg}$ | | | |
| | | | $\delta = 0.70$ | | $\delta = 0.70$ | |
| | Val | Test | Val | Test | Val | Test |
| *LED$_{Orig}$ (Q, P)* | 10.69 | 10.58 | 12.25 | 6.85 | 11.26 | 5.25 |
| *LED$_{Orig}$ (Q, P, H)* | 8.43 | 8.50 | 9.90 | 7.19 | 8.61 | 7.48 |
| *LED$_{Split}$ (Q, P)* | 17.91 | 14.24 | 10.93 | 6.72 | 11.24 | 10.24 |
| *LED$_{Split}$ (Q, P, H)* | 10.56 | 9.97 | 10.94 | 7.80 | 10.13 | 6.22 |
| Tf-Idf | 10.85 | 11.41 | 3.99 | 5.85 | 4.51 | 7.01 |
| S-BERT | 7.27 | 9.95 | 8.34 | 8.4 | 7.75 | 8.34 |
| Graph Tf-Idf | / | / | 9.42 | 8.19 | 9.07 | 9.60 |
| Graph S-BERT | / | / | 3.80 | 5.99 | 4.06 | 7.72 |
| Human (LB) | 51.26 | | | | | |

also compute the ratio of candidates amounts to the number of paper sentences and the number of argumentative sentences.

We observe that the argumentative graph is exploited in $\approx 8-27\%$ of examples on average. Moreover, the search space for graph-based fact selection baselines is relatively tiny compared to the number of paper sentences ($\approx 15-40\%$) and the number of argumentative sentences ($\approx 26-59\%$). Altogether, these results are particularly valuable as some models, such as Tf-Idf, manage to reach comparable performance to their counterpart with full knowledge, i.e. which has access to all paper sentences for fact selection, while using far less data. Additionally, performance improvement is also observed for their argumentative baselines. Therefore, the argumentative graph represents a valuable form of added knowledge as (i) arguments are often selected as supporting facts in dialogues and (ii) the semantic structure of the graph helps in attentively retrieving valuable information when the graph is navigated according to the dialogue history. These results suggest that the argumentative information of scientific papers can act as valuable knowledge for fact selection.

### Response Generation

Table 3.21 reports LED performance with different input combinations. We cannot entirely determine human performance due to multiple dialogue intents. In particular, the capability of mixing actions into messages is arbitrary. As expected, giving supporting facts as input significantly boosts model performance by $\approx 3-6$ Message-F1 points depending on the given input combination. In contrast, dialogue history seems to not bring any additional benefit to the task. This result is partly expected as collected dialogues are relatively short and centred

Table 3.18: Fact selection performance for LED models trained on the Split dataset according to Best and Union merging strategies.

| Input | Fact-F1 | | | | | |
|---|---|---|---|---|---|---|
| | $\mathcal{L}_{fact}$ | | $\mathcal{L}_{arg}$ | | | |
| | | | 0.70 | | 0.75 | |
| | Val | Test | Val | Test | Val | Test |
| *Union (Q, P)* | 13.94 | 12.37 | 8.33 | 5.61 | 8.59 | 8.36 |
| *Best (Q, P)* | 17.91 | 14.24 | 10.93 | 6.72 | 11.24 | 10.24 |
| *Union (Q, P, H)* | 8.42 | 8.55 | 8.17 | 6.36 | 7.66 | 5.39 |
| *Best (Q, P, H)* | 10.56 | 9.97 | 10.94 | 7.80 | 10.13 | 6.22 |

Table 3.19: Baseline performance for graph argument-based fact selection.

| Input | Fact-F1 | | | | | |
|---|---|---|---|---|---|---|
| | $\mathcal{L}_{fact}$ | | $\mathcal{L}_{arg}$ | | | |
| | | | 0.70 | | 0.75 | |
| | Val | Test | Val | Test | Val | Test |
| Tf-Idf | 10.85 | 11.41 | 3.99 | 5.85 | 4.51 | 7.01 |
| Graph Tf-Idf ($i = 0$) | / | / | 7.97 | 9.86 | 8.41 | 10.73 |
| Graph Tf-Idf ($i = 1$) | / | / | 8.27 | 8.55 | 8.27 | 9.60 |
| Graph Tf-Idf ($i = 2$) | / | / | 9.42 | 8.19 | 9.07 | 9.60 |
| S-BERT | 7.27 | 9.95 | 8.34 | 8.4 | 7.75 | 8.34 |
| Graph S-BERT ($i = 0$) | / | / | 3.80 | 5.99 | 3.41 | 5.99 |
| Graph S-BERT ($i = 1$) | / | / | 3.00 | 7.72 | 3.25 | 6.70 |
| Graph S-BERT ($i = 2$) | / | / | 3.29 | 7.04 | 4.06 | 7.72 |

Table 3.20: Graph statistics for graph argument-based fact selection baselines. We report the ratio of samples with at least one candidate extracted from the argumentative graph that is not a fact from dialogue history (Graph Usage Ratio). Additionally, we report the ratio between the number of candidates and (i) the number of paper sentences (Paper Candidates Ratio) and (ii) the number of argumentative sentences (Argumentative Candidates Ratio).

| Model | Graph Usage Ratio (%) | | | | Paper Candidates Ratio (%) | | | | Argumentative Candidates Ratio (%) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\delta = 0.70$ | | $\delta = 0.75$ | | $\delta = 0.70$ | | $\delta = 0.75$ | | $\delta = 0.70$ | | $\delta = 0.75$ | |
| | Val | Test | Val | Test | Val | Test | Val | Test | Val | Test | Val | Test |
| Graph Tf-Idf ($i = 0$) | 0.0 | 0.0 | 0.0 | 0.0 | 20.81 | 27.74 | 20.30 | 26.57 | 35.74 | 42.36 | 36.95 | 45.42 |
| Graph S-BERT ($i = 0$) | 0.0 | 0.0 | 0.0 | 0.0 | | | | | | | | |
| Graph Tf-Idf ($i = 1$) | 10.43 | 24.04 | 8.70 | 16.35 | 16.05 | 29.08 | 15.05 | 25.06 | 26.72 | 43.73 | 26.89 | 42.47 |
| Graph S-BERT ($i = 1$) | 12.17 | 24.04 | 9.57 | 22.12 | | | | | | | | |
| Graph Tf-Idf ($i = 2$) | 15.65 | 25.0 | 13.04 | 17.31 | 18.78 | 40.74 | 16.62 | 34.49 | 31.32 | 63.27 | 29.75 | 59.18 |
| Graph S-BERT ($i = 2$) | 19.13 | 26.92 | 15.65 | 24.04 | | | | | | | | |

on different parts of the scientific paper (Section 3.4.2). Thus, the dialogue history may not contain relevant information to the generation task. The overall low achieved Turn-F1 reflects the task's difficulty, leaving room for improvements.

Table 3.21: Performance on DE message generation with different input combinations.

| Model | Message-F1 | | | |
|---|---|---|---|---|
| | Orig | | Split | |
| | Val | Test | Val | Test |
| *LED (Q)* | 16.54 | 13.72 | 45.59 | 40.19 |
| *LED (Q, P)* | 17.32 | 14.93 | 43.89 | 36.11 |
| *LED (Q, P, H)* | 17.26 | 13.96 | 44.59 | 37.19 |
| *LED (Q, F)* | 20.92 | 19.54 | 46.14 | 37.15 |

Table 3.22: Sentence-level and message-level performance on DE message generation with different input combinations on the Split dataset.

| Model | Message-F1 | | | |
|---|---|---|---|---|
| | Split (Sentence) | | Split (Message) | |
| | Val | Test | Val | Test |
| *LED (Q)* | 11.19 | 9.63 | 45.59 | 40.19 |
| *LED (Q, P)* | 11.82 | 9.91 | 43.89 | 36.11 |
| *LED (Q, P, H)* | 11.77 | 10.37 | 44.59 | 37.19 |
| *LED (Q, F)* | 13.60 | 10.07 | 46.14 | 37.15 |

Regarding the Split dataset, the reduced complexity leads to significant performance improvement compared to the Orig dataset. In particular, the LED model gains up to $\approx 18-29$ Message-F1 points. Interestingly, the ($Q$) input combination achieves one of the highest performances on both evaluation splits. Nonetheless, additional evaluation metrics are required as the Message-F1 score does not account for word ordering [51]. For instance, Table 3.22 reports sentence-level and message-level LED model performance on the Split dataset. We notice that sentence-level Message-F1 for the ($Q$) and ($Q$, $F$) input combinations is 11.19 and 13.60, respectively. We generally observe an incremental sentence-level performance trend when considering more informative input combinations. Nonetheless, such a trend is not visible when considering merged message-level predictions as the Message-F1 metric does not consider word orderings. Action annotations could also be employed in the Split dataset to instruct the generative model. We leave this as future work.

### 3.4.4 Discussion

Experimental results on dialogue generation and fact selection show that scientific dialogues are a challenging domain for conversational agents. We have experimentally evaluated the contribution of argumentative graphs for fact selection. In particular, in some scenarios, the conversational agent solely relying on argumentative information reaches similar performance with its counterpart that has direct supervision on supporting facts.

However, the application of the argumentative graph is hindered by the lack of adequate data. First, we relied on the DrInventor corpus to train argumentative models. However, this corpus contains scientific papers on computer graphics and not on NLP. Second, we would prefer manually annotated argumentative graphs rather than relying on models for automatic detection. Indeed, a pipeline approach for argument detection hides prediction errors in each pipeline step. Altogether, these errors exacerbate the adoption of models for the automatic extraction of an argumentative graph. The capability of leveraging additional annotation layers like the argumentative one opens to a wide variety of research directions. We briefly outline some research directions worth exploring: (i) digesting long scientific documents with sampling-based strategies and (ii) argument-guided dialogue generation.

### Handling Long Scientific Documents with Sampling

Recent work has proposed efficient models that scale up to large documents [12]. However, their applicability to long documents is still hindered by several imperfections, such as important resource requirements, slow computational times, and a maximum document length. We formulate the task of document reading as a sampling task [158]. In particular, we introduce an information extraction agent that samples content from the available scientific paper. The task of information retrieval is an iterative process: (i) the agent attempts to select a few sampling candidates; (ii) the agent receives feedback based on the associated evaluation criterion and (iii) the agent updates its internal sampling policy based on the feedback.

The given sampling granularity defines candidates for sampling. For instance, as in ArgSciChat, each paper sentence could act as a potential candidate. Additionally, hierarchical approaches to reduce the search space progressively can be considered [84]. As for the evaluation criterion, we could consider the performance of the conversational agent regarding response generation as in an end-to-end architecture. The described sampling-based dialogue generation approach views the problem of handling long textual documents as a search problem. The main advantage of this method is that the complexity of the model is mainly attributed to its sampling component. Nonetheless, the problem of handling an ample search space remains.

To this end, we plan to adopt the argumentative graph to restrict the sampling space while not excluding potential valid candidates. In particular, at each sampling step, the sampling agent can decide to use the topology of the argumentative graph to select new candidates (location-based sampling) or exploit current information to look for non-local candidates according to a similarity metric (content-based sampling).

**Argument-guided Dialogue Generation**

We can also employ the information of the argumentative graph for dialogue generation. Structured representations like the argumentative graph represent a valuable resource for conversational agents: (i) allow the integration of multiple and heterogeneous sources, such as online blogs, dialogue history and other documents; (ii) act as a verification tool to explain the decision making process of the agent; (iii) define a general tool for information extraction that is not limited to a query-driven approach. The dialogue history represents one relevant information source. We can categorize the environment on which the conversational agent operates: (i) the static component, i.e. the scientific paper and the corresponding argumentative graph; (ii) the dynamic component, the dialogue history. As the dialogue goes on, the dialogue turns accumulate, defining an incremental dialogue history. From this perspective, we extend the dialogue generation task by conditioning the generation on both the static and dynamic components. More precisely, the conversational agent already uses both the scientific paper and the dialogue history to respond.

Nonetheless, as it increases in size, we seek to bridge connections between the argumentative graph and the dialogue history itself. In particular, we define a new dynamic component, denoted as the *dialogue graph*. The dialogue graph comprises dialogue messages as nodes and their inferred relations as edges. Additionally, nodes in the dialogue graph are also connected to nodes in the argumentative graph or, more in general, to the structured representation of the scientific paper. By doing so, a structured representation of both the dialogue and related employed paper content is summarized into a single structure.

The dialogue graph has two main advantages regarding dialogue generation: (i) it provides a backup explanation of the generation process since we can inspect the sub-graph used to respond; (ii) it provides a high-level input source that can boost the generation process. The main requirement of such research direction is defining a model capable of inferring relations between dialogue messages to define the dialogue graph eventually. A concrete example of the described approach is represented in Figure 3.10.

## 3.5 Considerations on Knowledge Extraction

This chapter has explored two concrete scenarios regarding structured knowledge extraction (KE). In the first, structured text representations, i.e. sub-trees, aim at capturing high-level textual features. More precisely, we have declined the knowledge extraction task to automatically retrieving sub-trees from a given input constituency tree. These sub-trees represent a structured feature type that is employed for the task of argumentative sentence detection. In the second

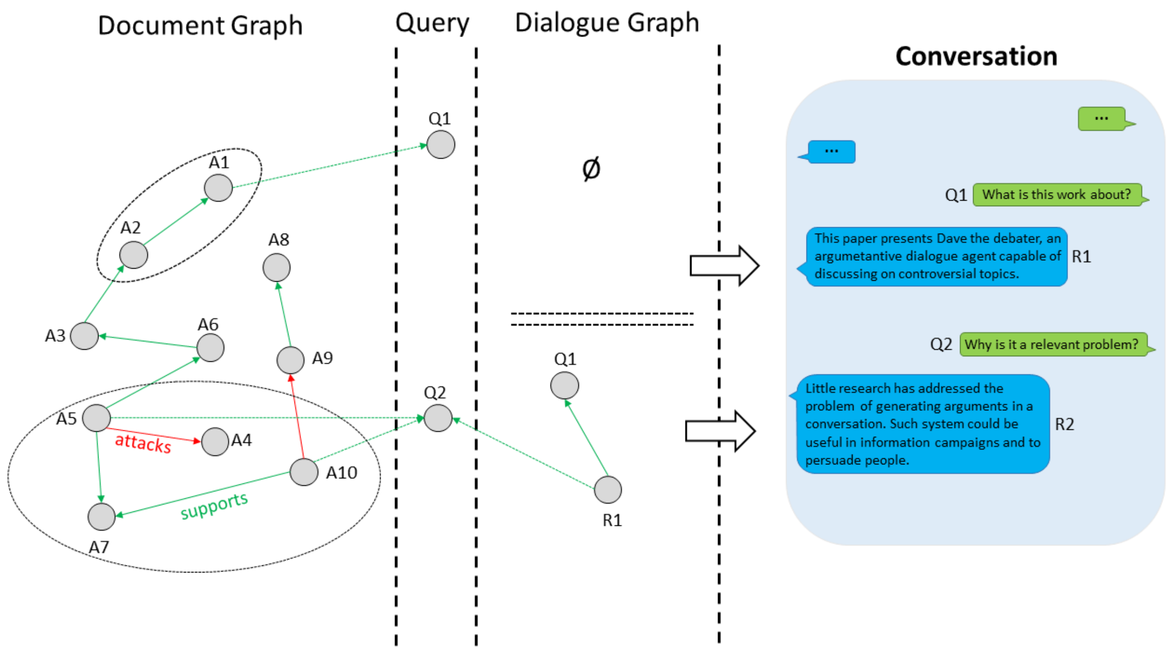

Document Graph
Query
Dialogue Graph
Conversation
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
attacks
supports
Q1
Q2
∅
Q1
R1
...
...
Q1 What is this work about?
This paper presents Dave the debater, an argumetantive dialogue agent capable of discussing on controversial topics. R1
Q2 Why is it a relevant problem?
Little research has addressed the problem of generating arguments in a conversation. Such system could be useful in information campaigns and to persuade people. R2

Figure 3.10: Argumentative dialogue generation. Using argument relations, the agent can integrate document information with current dynamic dialogue content to respond appropriately to the user's query. In particular, argument relations broaden the search space, thus, allowing the agent to access additional content that query-based retrieval methods lack.

scenario, automatically extracted arguments from scientific papers guide a conversational agent in two different tasks: (i) dialogue generation and (ii) supporting fact selection.

These structured knowledge representations can represent a valuable source of information. In the case of TC-GNNs (Section 3.3), extracted sub-trees lead to improved performance over several argument mining datasets. Additionally, the extracted sub-trees, albeit achieved via soft constraints, can be quickly retrieved for inspection. In particular, we can evaluate whether different argument classes are characterized by different structured textual patterns. This approach falls under the frame of argument evaluation. Here, the focus is on determining the underlying distinguishing properties of arguments and their components [52, 96].

Similarly, the extracted argumentative graph for dialogues on scientific papers aids the conversational agent in reducing its search space for supporting fact selection. Despite the limited search space, the conversational agents solely rely on the argumentative graph to achieve comparable performance with their counterparts with direct supervision. In addition to task-specific contributions, the argumentative graph can be potentially employed for different purposes. For instance, we highlight the integration of heterogeneous sources, such as online blogs, scientific documents and dialogue history.

To sum up, in both experimental scenarios, a KE process can be a valuable component for the definition of high-level model reasoning. Additionally, structured text representations can provide a unified input representation, as shown in ArgSciChat (Section 3.4). Nonetheless, the significant advantages of a KE process also carry along some strong assumptions. First, an ad-hoc model for KE is required. Such a model is meant to be pre-trained on a provided data set to accomplish its task. For instance, argumentative models for ArgSciChat had to be trained on the DrInventor corpus [79, 141]. In many cases, such preliminary requirement becomes a strong assumption that may not be guaranteed. For instance, we have shown that no adequate dataset with argument annotations for NLP exists in the literature. The lack of sufficient data may lead to poor results and additional challenges concerning KE.

Furthermore, there might be cases in which the desired structured knowledge is domain- or task-specific. Thus, existing tools for KE like tree parsers [167] or knowledge graphs do not provide the required information to address the task. For instance, specific knowledge graphs that expose domain-specific entities and connect them are usually required in the legal domain. Nonetheless, defining such resources is prohibitively expensive in the majority of the scenarios. Such an annotation process can represent a significant bottleneck depending on the given domain.

In conclusion, KE is an effective supplement when addressing a particular task. However, we have to evaluate such a process's cost and efficacy properly. In the context of no available structured knowledge, we explore unstructured knowledge integration (UKI). The main advantage of this approach is that no particular expertise regarding a given structured representation is needed. In other terms, we can consider unstructured knowledge representations from different expertise levels independently of each other. Nonetheless, integrating such unstructured knowledge becomes particularly challenging since no specific semantic interpretation is given. In contrast, structured representations like knowledge graphs provide a semantic frame for the integration and evaluation of known knowledge. We then focus on the definition of adequate unstructured knowledge integration processes.

# Chapter 4

# Unstructured Knowledge Integration

In some scenarios, we cannot guarantee the availability of adequate structured knowledge. Generally speaking, extracting and formulating well-organized knowledge can be quite challenging, time-consuming or not feasible. This chapter presents some experimental scenarios where structured knowledge is not available and challenging to define. To this end, we formulate these scenarios from the point of view of unstructured knowledge integration (UKI). Our objective is to overcome restrictions concerning structured knowledge definition while maintaining some possible derived benefits like some level of model interpretability and an increased generalization capability from limited data. Among the many challenges regarding UKI, dealing with the natural language formulation of knowledge is one of the core dynamics to consider. In particular, the absence of a specific knowledge interpretation semantic frame creates the necessity of defining a good UKI process.

As a preliminary study, we investigate the unstructured knowledge integration mechanism of comparison. More precisely, unstructured knowledge is defined as a collection of relevant descriptors for a particular class in a classification setting. Such descriptors can be class rationales or contextual data in highly correlated information concerning text inputs. We choose memory-augmented neural networks (MANNs) [252, 276] as the base model architecture for defining a UKI process as a natural way to extend any deep learning model with an external knowledge set. The flexibility of such architecture allows decoupling the unstructured knowledge integration process with its application. By doing so, we can progressively consider selective extensions. For instance, integrated knowledge can shift from being static to being iteratively updated throughout the learning process while keeping the integration semantic fixed [282]. We explore simple MANN architectures mainly focused on defining an unstructured knowledge mapping mechanism that addresses task-specific requirements. Additional UKI subprocesses like knowledge validation are partially covered due to their challenging satisfiability. Despite these limitations, we remark the difficulty of defining an unstructured knowledge mapping mechanism when facing issues like large-sized knowledge, partial knowledge coverage and direct knowledge supervision. Although some knowledge validation properties are not quantitatively measured and verified, integrating unstructured knowledge into simple neural architectures via the MANN architecture is sufficient to provide notable benefits like improved model performance.

## 4.1 Knowledge Representation

In this chapter, we consider two arbitrary textual knowledge types (KTs): (i) class rationales concerning the positive class of a binary classification task; (ii) contextual data that is highly correlated with the text inputs belonging to the positive class of a binary classification task. As a preliminary case study, we consider a classification problem in both scenarios as it imposes fewer requirements for knowledge integration and represents a more manageable task than generation [91]. The first KT we explore is in the legal domain. In particular, we consider the automatic detection of unfair clauses in legal documents. In such a scenario, we use expertise explanations, denoted as *legal rationales*, to motivate the classification of an unfair clause. Thus, legal rationales are the unstructured knowledge for unfair clause detection.

Intuitively, the integration of legal rationales requires a knowledge integration mechanism that allows some verification process. The classification model learns how to use unstructured knowledge to classify unfair clauses better. The interaction between the model and the unstructured knowledge should be made explicit and interpretable to properly verify if the model has learned to associate text inputs to legal rationales. Similar behaviour is desired in our second experimental scenario. We explore the task of argument component detection. In particular, argument components are the classes for the classification task. The unstructured knowledge is represented by text data belonging to a given component class. Such knowledge is employed as contextual data to ease the task of classification. More precisely, we consider two argumentative components based on Walton's argument model [268]: the *claim*, i.e. an opinion or subjective proposition, and the *evidence* or *premise*, i.e. a fact that supports the definition of the claim.

We then formulate the task of claim detection by using text examples labelled as evidence as our unstructured textual knowledge. Evidence and claims are paired by their argumentative support relation. Thus, it is possible to explicitly associate each evidence text example with its corresponding claim text example in the dataset. Unlike the legal domain, the unstructured knowledge comprises non-synthetic text examples that might ease the detection of claims based on the correlation of each (evidence, input example) pair. In both scenarios, unstructured knowledge is a valuable indicator to aid a model in addressing a classification task. However, legal rationales are also the desired output of the model itself. The model should be capable of jointly classifying and motivating its decision process. On the other hand, such a tight requirement is not enforced in argument component detection.

In the latter scenario, the preferred behaviour is when the model can select the correct evidence for the detected claim. Nonetheless, in the domain of Argument Mining (AM), a given text can be labelled as a particular component depending on contextual information [35]. Thus, provided argument relations are tied to the provided context. For instance, the same text can appear as a claim and a evidence throughout the dataset. For this reason, the employed knowledge does not impose an additional model behaviour, e.g. motivating its decision process, but rather provides valuable information to ease the decision process itself and remove potential ambiguities.

## 4.2 Challenges

Many challenges characterize the integration of unstructured textual knowledge. This section describes the major challenges that we face in the preliminary experiments listed in this chapter. First, we consider the problem of dealing with knowledge of poor quality. Second, we describe the issue of modelling direct knowledge supervision. Third, we discuss the case when the knowledge only covers a subset of input data. In particular, we consider how we should model the unstructured knowledge integration (UKI) processes. Lastly, we evaluate how incorrect mappings between text inputs and the knowledge should affect the unstructured knowledge mapping mechanism. More precisely, we reason to what extent such incorrect mappings can condition model learning depending on the given knowledge mapping mechanism.

### 4.2.1 Poor Knowledge Quality

Unstructured textual knowledge hides several challenges regarding its natural language formulation. In this chapter, we present the challenge of integrating knowledge of poor quality. In particular, quality is measured in UKI properties like integration robustness and efficiency (Section 2.4). For instance, in Section 4.3, we address the classification task of unfair clause detection in the legal domain with class rationales. However, the formulation of such rationales is not particularly compliant with the given input data to classify. For example, some class rationales are too abstract or contain multiple reasoning concepts. Thus, some class rationales should be split or reformulated better.

Having knowledge of poor quality is undoubtedly a possibility that we can encounter when considering UKI. Therefore, we require the definition of adequate solutions. These solutions can involve a dynamic knowledge representation. For instance, if a team of experts defined the original class rationales, we can consider instructing them to refine the knowledge. Alternatively, we can also consider a dynamic knowledge update step applied at the unstructured knowledge integration sub-process. In this case, we use the data-driven approach to instruct a model and update the knowledge.

### 4.2.2 Supervised Knowledge Mapping

Depending on the given problem setting and the employed knowledge type (KT), direct supervision about input to knowledge mappings might be available. Such information is particularly valuable as it allows quantitatively measuring *correctness*. Additionally, direct supervision opens the path to more intuitive explainable systems. In particular, direction supervision allows a critical evaluation of potential language biases while assessing fundamental properties, such as input stability. These properties are mainly required in settings subject to hard constraints, such as safety. In this scenario, model predictions must be sustained by confident motivations.

Nonetheless, correctly encoding direct supervision into the knowledge mapping process may represent a challenge by itself. As direct supervision is strongly coupled with how the UKI process is defined, we require direct supervision to be aligned with the unstructured knowledge mapping mechanism. For instance, in the context of unfair clause detection (Section 4.3), a text to classify can be associated with multiple class rationales. However, it is sufficient to select just one correct class rationale based on how the task is formulated. In this context, direct supervision should consider the equivalence of class rationales associated with a given text input.

### 4.2.3 Partial Knowledge Mapping

Another relevant challenge concerning UKI is the partial coverage of knowledge for input data. In this scenario, the knowledge is only applicable to a specific input data subset. Thus, the unstructured knowledge mapping process should not alter the decision-making process of a model when dealing with input data that is not covered by the knowledge. For instance, in the context of unfair clause detection (Section 4.3), the class rationales only cover the set of unfair clauses in the dataset. In particular, unfair clauses are just one minor portion of the dataset. Therefore, it is particularly crucial to define an unstructured knowledge mapping mechanism capable of attentively making use of knowledge when required. Additionally, the partial coverage of knowledge should be compatible with other dynamics like direct knowledge supervision.

### 4.2.4 Dealing with Incorrect Knowledge Mapping

One subtle dynamic that concerns the unstructured knowledge mapping process is described by using incorrect knowledge for given text input. In this scenario, other tightly related dynamics must not be altered when the model incorrectly uses the knowledge. For instance, in Section 4.5 we explore unstructured sampling to account for large knowledge sizes. In particular, sampling strategies extract knowledge portions, i.e. knowledge elements, to define a new knowledge of nominal size. In this context, sampling strategies are explored based on how well the knowledge can benefit the decision-making process of a model. That is, we jointly learn to address the task and retrieve the proper knowledge subset for a given input.

Consequently, the unstructured knowledge mapping mechanism should be robust to sampling irrelevant knowledge elements for given text input. Furthermore, the unstructured knowledge mapping mechanism should learn how to correct these irrelevant mappings similarly to how the model receives feedback regarding its decision-making process.

## 4.3 Unfair Clause Detection with Legal Rationales

We consider the task of unfair clause detection in consumer contracts. The task is formulated as a classification problem where a sentence be labeled as unfair or not. Despite a clause can range from one to multiple sentences, we explore the simplified problem where each sentence is treated individually. Thus, no contextual information is considered for the classification. In this domain, classification models are tight to strong requirements. For instance, the decision process of the classification model should be easily evaluable by legal experts. This is a required process for experts to adopt the proposed classification model.

As a solution, we consider experts' explanations, denoted as *legal rationales* regarding unfairness as a form of unstructured textual knowledge to be integrated into the classification models [136]. A legal rationale describes for what reasons a clause has been labelled as unfair. We focus on the possible integration of a knowledge base of legal rationales. That is, the unstructured knowledge to be integrated is represented by legal rationales. Legal rationales fall within the KT of class rationales (Section 2.6.4). This corresponds to the idea that human lawyers use legal rationales to provide explanations for intuitions of unfairness. Additionally, lawyers use legal rationales to guide such intuitions, pointing to general features relevant to unfairness, that could be shared by other similar clauses. Via an adeguate unstructured knowledge integration mechanism, legal rationales can be employed by the classification models to: (i) aid the classification process by leveraging domain information; (ii) providing a motivation of the decision making process of a model.

We adopt Memory-Augmented Neural Networks (MANNs) [252, 276] as a way to encode legal rationales into the classification model to achieve above benefits. Indeed, MANNs define a model architecture with an external memory component on which information can be stored and read. In their basic formulations [252], MANNs also define a simple knowledge mapping mechanism based on semantic similarity. Such property permits to condition the classification problem on the unstructured knowledge, thus satisfying (i). Additionally, the same unstructured knowledge mapping mechanism offers ways to evaluate to what extent the knowledge has been used, thus satisfying (ii). We initially introduce a new dataset for unfair clause detection with legal rationales annotations. We then describe our method for unstructured knowledge integration of legal rationales. Lastly, we report experimental results and discuss the case study from the point of view of UKI. More details about this work are reported in [136].

### 4.3.1 Data

Our starting point is the dataset produced by [154], consisting of 50 relevant online consumer contracts, i.e., Terms of Service (ToS) of online platforms, analyzed by legal experts and marked in XML. We selected the new contracts among those offered by some of the major players in terms of global relevance, number of users, and time the service was established. We focused on eight categories of clauses, which most often are unlawful or unfair, i.e., clauses establishing: (1) jurisdiction in a state different from the consumer's (J); (2) choice of a law other than the consumer's (LAW); (3) limitation of provider's liability (LTD); (4) provider's right to unilaterally terminate the contract and/or access to the service (TER); (5) provider's right to unilaterally modify the contract/service (CH); (6) arbitration on disputes arising from the contact (A); (7) provider's right to unilaterally remove consumer content from the service, including in-app purchases (CR); (8) acceptance of contract by the mere use of the service, even when the consumer has not read the contract or explicitly agreed to it (USE) [154].

All the ToS we analyzed are standard terms available on the provider's website for review by potential and current consumers. Indeed, as it will be noted below, many providers assert that rather than being an obligation on the service provider to notify their users regarding changes to the ToS and even to the service, consumers are required to review the ToS from time to time, visiting the website on a regular basis to check for any changes. As reported by [159] and as our research indicates [176], such categories are widely used in ToS for online platforms.

For the purposes of this study we focused on *limitation of liability* (LTD). Clauses falling under this category stipulate that the duty to pay damages is limited or excluded for certain kinds of losses and under certain conditions. One reason for focusing on limitation of liability is that LTD is the category for which we have the largest number of problematic clauses in our dataset. More precisely, our corpus contains a total of 21,063 sentences, 674 of which contain a potentially or clearly unfair clause (note that the total number of sentences containing a potentially unfair clause, in any category, is 2,346).

In particular, clauses excluding liability for broad categories of losses or causes of them were marked as potentially unfair, including those containing blanket phrases like “to the fullest extent permissible by law”. Conversely, clauses meant to reduce, limit, or exclude the liability for physical injuries, intentional harm, or gross negligence were marked as clearly unfair [154, 176]. The second observation concerns the particular difficulty of detecting unfair LTD clauses. Our classifier has shown lower performance on such clauses in comparison to other categories [154]. Moreover, focusing on a single category of unfairness makes it easier to circumscribe a dedicated knowledge base for testing the MANNs. However, this does not affect the significance of our experiments, since unfair limitation of liability can be identified on the basis of several different rationales.

### Legal Rationales

An initial analysis enabled us to identify 21 legal rationales for (potentially) unfair limitation of liability, which map different questionable circumstances under which the ToS reduce or exclude liability for losses or injuries. For each rationale we defined a corresponding identifier `[ID]`. The rationales have been formulated by two independent legal experts, each adopting different approaches. The first approach was more synthetic, and produced a smaller number of broad grounds of unfair exclusion of liability (Table 4.1). The second approach was more analytical, and produced many explanations, each describing multiple kinds of unfairly excluded losses or damages (Table 4.2). The two lists of rationales were then merged and used together for running the experiments.

Table 4.1: Legal rationales for the legal qualification of unfairness (synthetic approach).

| **ID** | **Legal Rationale** |
|---|---|
| `blanket_phrase` | `The limitation of liability uses a blanket phrase to the fullest extent permissible by law, any indirect or incidental damages, liability arising out of or in connection with these Terms or similar.` |
| `srv_con_liab` | `Liability is excluded in cases related to availability, usability or legality of service, website and/or user’s content.` |
| `vir_malware` | `Liability is excluded for data loss, corruption or damage whether caused by viruses, trojan horses, malware or other malicious activity.` |
| `physical_harm` | `Liability is excluded also in cases of physical or personal injuries.` |

| | |
|---|---|
| third_party | Liability is excluded for the actions and/or services of third parties. |

Table 4.2: Legal rationales for the legal qualification of unfairness (analytical approach).

| ID | Legal Rationale |
|---|---|
| extent | since the clause states that to the fullest extent permissible by law the provider is not liable. |
| discontinuance | since the clause states that the provider is not liable for any technical problems, suspension, disruption, modification, discontinuance, limitation of services and features. |
| compharm | since the clause states that the provider is not liable for harm or damage to hardware and software, including viruses, worms, trojan horses, or any similar contamination or destructive program. |
| anydamage | since the clause states that the provider is not liable for any special, direct and/or indirect, punitive, incidental or consequential damage, including negligence, harm or failure. |
| amount | since the clause states that the compensation for liability or aggregate liability is limited to, or should not exceed, a certain amount. |
| thirdparty | since the clause states that the provider is not liable for any action taken from third parties or other people, including service and products, material and link posted by others. |
| security | since the clause states that the provider is not liable for any damage deriving from a security breach, including any unauthorised access. |
| disclosure | since the clause states that the provider is not liable for damages resulting from disclosure of data and personal information. |
| reputation | since the clause states that the provider is not liable for reputational and goodwill damages or loss. |
| anyloss | since the clause states that the provider is not liable for any loss resulting from the use of the service and or of the website, including lost profits, data, opportunity. |
| awareness | since the clause states that the provider is not liable whether or not he was, or should have been, aware about the possibility of any damage or loss. |
| contractfailure | since the clause states that the provider is not liable for any failure in performing contract and terms obligations, breach of agreement. |
| unilateral | since the clause states that the provider is not liable for any unilateral change or unilateral termination. |
| dataloss | since the clause states that the provider is not liable for any loss of data. |
| grossnegligence | since the clause states that the provider is not liable for gross negligence. |
| injury | since the clause states that the provider is not liable for personal injury and death. |

Each unfair limitation of liability clause in the training set has been indexed with on or more identifiers of rationales that apply to the specific clause. As an example consider the following clause taken from the terms of service of Badoo (last updated on 11 September 2018) and previously classified as potentially unfair:

```
To the fullest extent permitted by law, Badoo expressly excludes:all conditions,
representations, warranties and other terms which might otherwise be implied by statute,
common law or the law of equity; and any liability incurred by you arising from use of
Badoo, its services or these Terms, including without limitation for any claims, charges,
demands, damages, liabilities, losses or expenses of whatever nature and howsoever direct,
indirect, incidental, special, exemplary, punitive or consequential damages (however
arising including negligence), loss of use, loss of data, loss caused by a computer
or electronic virus, loss of income or profit, loss of or damage to property, wasted
management or office time, breach of contract or claims of third parties or other losses
of any kind or character, even if Badoo has been advised of the possibility of such damages
or losses, arising out of or in connection with the use of Badoo.
```

The clause above has been linked to the following identifiers one the analytical approach: `ID: extent, anydamage, compharm, anyloss, awareness, contractfailure, dataloss`. Conversely, on the synthetic approach, the clause has been associated to the following `ID: blanket_-phrase, vir_malware`. The link between rationales and clauses will be used in future experiments to instruct the system so that it can provide an explanation for the unfairness of particular clauses.

### 4.3.2 Method

Legal experts can detect unfair clauses by relying on multiple sources of knowledge, such as the applicable legal regulations, the relevant judicial cases, their trained common sense. They also use these sources also for generating rationales (explanations). While a system aimed at recognizing unfair clauses cannot be expected to reason like a lawyer, it should however be expected to provide results that match the assessments that a trained lawyer would give after carefully reading the document containing such clauses. In CLAUDETTE we have adopted a supervised machine learning approach, based on a training set of documents annotated by domain experts [154]. The system compares clauses classified as fair or unfair, and, based on such a comparison, it develops its implicit concept of unfairness. Such an approach has delivered very encouraging results, as noted above.

However, the legal knowledge used by the system is restricted to the annotations (category and unfairness level) provided by the experts, which does not directly point to the rationales behind the annotations. Conversely, we aim to directly exploit legal rationales for unfairness in a forward-looking way. This corresponds to the idea that human lawyers use rationales not only to provide explanations for intuitions of unfairness, but also to guide such intuitions, pointing to general features relevant to unfairness, that may be shared by other similar clauses.

### Knowledge Integration via MANNs

To provide a computable model for the forward-looking use of rationales we rely on a particular category of deep learning models, denoted as Memory-Augmented Neural Networks (MANNs) [92, 276]. MANNs are a type of a recurrent neural network (RNN) that introduces an external memory block as a support for reasoning. Given an input, the model checks whether the memory contains some slot that is related to that input (e.g., through a similarity measure). Subsequently, the memory content is extracted and coupled with the given input to accomplish the classification task.

We can distinguish between two phases: (i) the *memory addressing* sub-process, where the network computes some representation of the input to operate with the memory; (ii) the *reasoning* sub-process, where the new content is distilled for the resolution of the task. The inter-operability of the model with respect to the memory block is general: the enhanced representational capacity is equivalent to a Turing-complete machine, which can read and write information on an hypothetical infinitely large memory tape. MANNs have been widely used for complex tasks where reasoning about the context of the given inputs plays a key role: question answering [25, 26, 134, 252], sentiment analysis [257], reading comprehension [40, 109, 179], graph analysis and navigation [93, 94]. From a technical point of view, we explore a simple variant [252] of the general concept of MANNs, named end-to-end memory network, and define the task of unfair clause detection as a binary classification task [154], where a given input clause can be labelled as either fair or (potentially) unfair.

Since unfairness detection inherently requires a comparison action with an adequate knowledge base, covering syntactic, lexical and most importantly semantic 'patterns', we require the model to have access to the knowledge base while analyzing the input clause. The knowledge base stored in the memory consists of a fixed collection of possible rationales for unfairness. These rationales were provided by legal experts, based on their experience or on the case law, and linked by the same experts to the clauses they apply to. Some examples of such rationales are reported in Table 4.2).

Concerning knowledge integration, we consider provided unfairness prior information as it is, i.e. raw texts, since conveying such knowledge via structured formulations, such as knowledge graphs, stand as a challenging task itself. For instance, consider the `awareness` explanation in Table 4.2. Compared to other simpler examples, such as `physical_harm`, that intuitively introduce knowledge by highlighting relevant words or phrases, the type of information provided by the `awareness` rationale is quite challenging to correctly distill. In particular, it requires high-level understanding which cannot easily be formulated as a single logic rule or by means of entity relations.

**How Integrated Knowledge is employed**

When analyzing an input clause, the system accesses the knowledge base to retrieve the rationales that best match such input. The links between statements and rationales established by the system provide a simple criterion for qualitative analysis of the model. In particular, if clause-rationale links were given for some samples, it would be possible to compare such references made by the experts with those made by the MANN. Considering notable tasks where MANNs have been employed, namely question answering and reading comprehension, the problem of unfair clause detection locates nearly halfway between them. Typically, in simple question answering, a single memory slot containing necessary information about the answer is demanded. Moreover, contents stored in the memory blocks are usually independent of each other and do not represent fragmented sequential information.

Conversely, in reading comprehension, document sentences referenced by the input question are stored in individual memory slots. In this scenario, preserving sequential knowledge is usually an important factor for correctly solving the task. In our case study, memory definition follows the one of question answering, since explanations are independent of each other. However, depending on the given knowledge formulation, multiple slots may be consulted in order to correctly detect violations in clauses text, just as in reading comprehension. In broad terms, the method employed by the MANN to classify a given input clause is as follows. The MANN iteratively performs reading operations based on a similarity metric between each memory slot and the input clause. Subsequently, the MANN extracts memory content by combining all memory slots, and attributing to each slot a weight proportional to the similarity score. Next, the extracted content is added to the current input to build a representation that can possibly be distilled as a new input for another iteration (reasoning phase). In this way, past iterations are always taken into account during memory reading. Eventually, after the last memory iteration, the network operates on the distilled input to predict a fair/unfair label. Note that the same memory slot may be read multiple times in order to properly exploit its content, since a distribution of weights is applied to the whole memory block. Figure 4.1 illustrates the architecture.

### 4.3.3 Experiments

Our experiments use the proposed dataset of consumer contracts, focusing solely on LTD clauses. We employ a 10-fold cross-validation as both an evaluation and a calibration method for our models of interest. In particular, the whole corpus is first split into 10 subsets, named folds. Each fold is then used, in turn, as the test set, whereas the union of the other folds is further split into a training set and a validation set, exploited for hyper-parameter tuning.

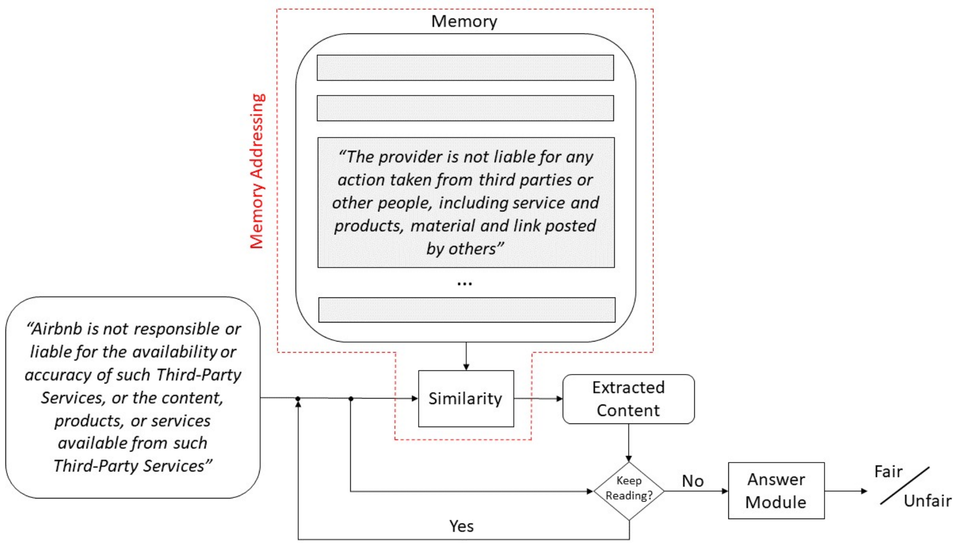

Memory
Memory Addressing
"The provider is not liable for any action taken from third parties or other people, including service and products, material and link posted by others"
...
"Airbnb is not responsible or liable for the availability or accuracy of such Third-Party Services, or the content, products, or services available from such Third-Party Services"
Similarity
Extracted Content
Keep Reading?
No
Yes
Answer Module
Fair
Unfair

Figure 4.1: Architecture of the proposed model for unfairness detection. Each given input clause to be classified is firstly compared with the memory block via a similarity metric. By doing so, pertinent content is extracted from the memory and coupled with the input clause. Subsequently, based on the chosen fixed amount of read iterations, the model either repeats the procedure described so far with the newly modified input, or uses content gathered so far to produce a prediction via a dedicated answer module.

Concerning calibration, we consider two simple baselines that were also tested in some of our previous work [154] about unfairness detection in consumer contracts: (i) a network comprised of stacked recurrent neural network layers, a variant of RNNs referred to as Long Short-Term Memory network (LSTM), used extensively in the deep learning community; and (ii) a network defined as a stack of convolutional neural networks (CNNs). Moreover, we consider the current state-of-the-art solution for this task [154], featuring at its core a Support Vector Machine (SVM). Among all these models, the MANN is the only one that leverages an external knowledge base. The only input of LSTM, CNN, and SVM is the clause to be classified.

The memory network we propose follows the architecture described in [252] with minimal differences. With respect to the system illustrated in Section 4.3.2, the model performs six iterations, i.e. hops, over the memory before producing an answer. Qualitative analysis of this behaviour is reported below. Just like in [252], the similarity operation between the input clause and each content stored in memory is implemented as a simple dot product between the two sentence-embedding vectors, numerical representations of the corresponding texts. Far more complex implementations have been adopted in the literature [94, 106, 134, 202, 282], and we will consider them in future extensions.

For the present study we decided to adopt the simplest similarity operation, because of the exploratory nature of our investigation. Once extracted, the distilled memory vector, defined as the weighted sum of read contents, is summed with the current input and used as input for the next iteration, as shown in Figure 4.1. As a last stage, a stack of fully connected layers is used to predict the classification label, given the latest memory-enhanced input.

To account for performance variations due to different initial configurations, we address model stability by repeating the cross-validation routine a sufficient amount of times. In our experimental setting we fix the number of repetitions to 10. In this way, it is possible to gather insight about performance variance and select at the same time the best-performing results. Input sensitivity is a crucial factor for detection systems, especially when the task is centred on infrequent or hard-to-detect anomalies. Furthermore, models were early stopped based on validation loss scores. Hyper-parameters calibration was accomplished via the same evaluation method, but without repetitions.

Table 4.3 reports the results of the proposed experimental setting. In particular, for all the models we report precision, recall, and $F_1$ scores, macro-averaged over the ten folds.[1] From the collected results, it is evident that baseline models that exclusively leverage the input-clause content fail to correctly classify the majority of legal violations in consumer contracts. On the other hand, the MANN model shows a strong improvement in performance with respect to the baselines. This indeed corroborates the rationale that background knowledge is a crucial element for this task. The proposed model exploiting an external memory shows even slightly better results than the state-of-the-art SVM.

Differently from other described architectures, modelling explicit comparison in memory networks presents the advantage of directly visualizing the interaction level of the model with respect to its memory blocks. This opens up the possibility of understanding *what* the model believes to be useful for the detection task. As an example, Figure 4.2 shows the overall memory usage over all folds.

It is clear that the model does not exploit all the memory slots equally. The most used memory is the one stating *"The limitation of liability uses a blanket phrase like to the fullest extent permissible by law, any indirect or incidental damages, liability arising out of or in connection with these terms, or similar"*, which is one of the most general explanations, also providing several examples. Overall, the eight most used memories account for over 45% of the cases labeled by our experts, which is very interesting, since we point out that no information about which explanations were linked to which clauses was given during training. The latter

[1]For all the neural models, we have exploited multi-start by training ten different networks for each fold, and selecting the best network for each fold according to the validation performance.

Table 4.3: Results on 10-fold cross-validation on ToS-100. Performance measures are macro-averaged.

| Model | Precision | Recall | $F_1$ |
|---|---|---|---|
| Baseline LSTM | 37.55 | **88.93** | 51.51 |
| Baseline CNN | 43.43 | 87.90 | 56.27 |
| Memory Network | **68.36** | 84.31 | **64.33** |
| SVM | 52.52 | 81.57 | 63.7 |

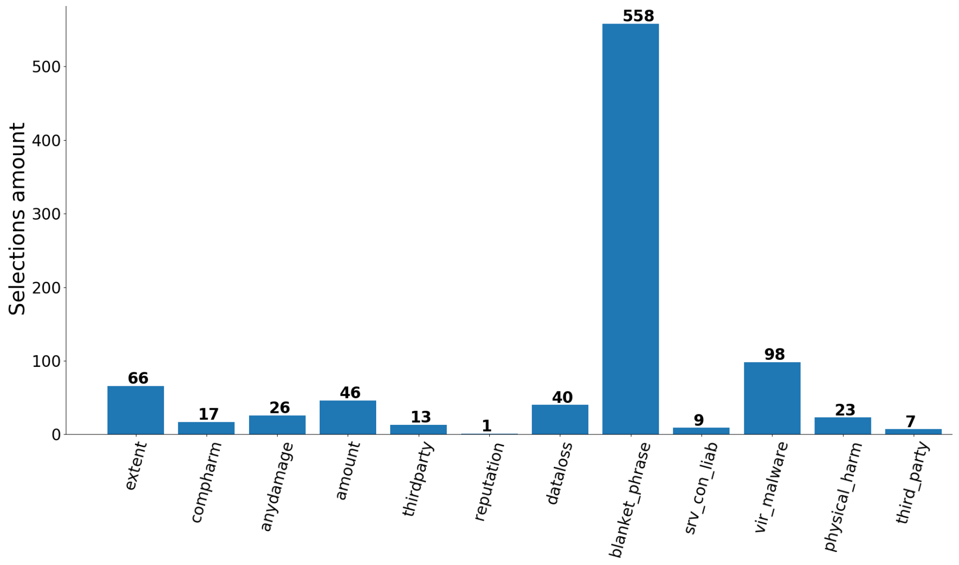


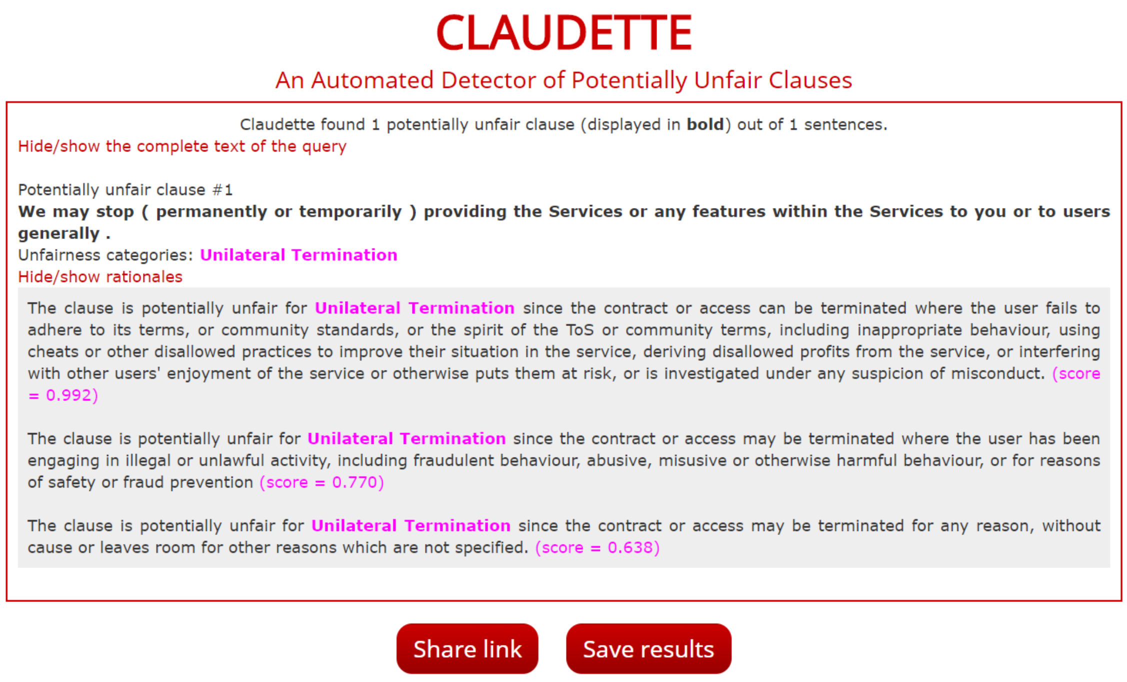
Figure 4.2: Cumulative memory distribution across all test folds. Slots never selected are not shown.

would be called a *strong* supervision for memory networks, and we plan to use it in future work.

### 4.3.4 A New CLAUDETTE Tool

We make use of the aforementioned MANN-based methodology to provide an extended version of the CLAUDETTE tool [154]. Figure 4.3 provides a concrete example for unfair clause detection where legal rationales, and their model confidence scores, are provided in addition. In particular, the tool offers the user the possibility to enter some text to analyse [150]; the input text is then separated into sentences, and each of them is classified as either unfair or not.

In the first case, the system also predicts the unfairness category. For each detected unfair sentence presented in the results web page, CLAUDETTE thus reports the unfairness category and, if any, also the list of legal rationales that were employed by the underlying MANN model during classification, each with a corresponding confidence score. In this way the user is not



Claudette found 1 potentially unfair clause (displayed in **bold**) out of 1 sentences.
Hide/show the complete text of the query

Potentially unfair clause #1
**We may stop ( permanently or temporarily ) providing the Services or any features within the Services to you or to users generally .**
Unfairness categories: **Unilateral Termination**
Hide/show rationales

The clause is potentially unfair for **Unilateral Termination** since the contract or access can be terminated where the user fails to adhere to its terms, or community standards, or the spirit of the ToS or community terms, including inappropriate behaviour, using cheats or other disallowed practices to improve their situation in the service, deriving disallowed profits from the service, or interfering with other users' enjoyment of the service or otherwise puts them at risk, or is investigated under any suspicion of misconduct. (score = 0.992)

The clause is potentially unfair for **Unilateral Termination** since the contract or access may be terminated where the user has been engaging in illegal or unlawful activity, including fraudulent behaviour, abusive, misusive or otherwise harmful behaviour, or for reasons of safety or fraud prevention (score = 0.770)

The clause is potentially unfair for **Unilateral Termination** since the contract or access may be terminated for any reason, without cause or leaves room for other reasons which are not specified. (score = 0.638)

Figure 4.3: Example of classification performed by the CLAUDETTE tool. Unfair sentences are highlighted in bold and tagged with predicted unfairness label, i.e., category. Additionally, if the memory has been used during classification, the list of exploited legal rationales along with model confidence score (ranging from 0 to 1) is reported.

only informed about the unfairness categories and reasons for unfairness, but also is given an indicator on how relevant these reasons are for the input text.

### 4.3.5 Discussion

We have investigated the integration of legal rationales with a focus on limitation of liability. As a preliminary study, the integration of unstructured knowledge has been formulated via MANNs for deep learning models. In particular, we explored a simple unstructured knowledge mapping mechanism via a neural attention mechanism [77]. The unstructured knowledge integration mechanism of entailment is then viewed as a simple embedding-based similarity operator with no asymmetry constraint [250].

As a preliminary study, we only focused on measuring unstructured knowledge integration from the point of view of model performance. Nonetheless, the knowledge type (KT) of legal rationales is class rationales. Thus, further studies regarding the unstructured knowledge validation process have to be carried out. For instance, we can quantitatively measure *correctness* if direct knowledge supervision is provided. We assume that *consistency* is guaranteed from the point of view of legal rationales definition.

However, their natural language formulation may represent a notable obstacle for their integration. Lastly, *coherence* is hardly measurable when considering neural attention models and the KT of class rationales. As a preliminary approach, we could rely on attention values to identify linguistic patterns that denote the entailment between the input clause and the selected legal rationales. Nonetheless, more advanced techniques that belong to other research areas like eXplainable Artificial Intelligence (XAI) [117] should be considered.

This study was motivated by two main goals. The short-term goal was aimed at verifying whether the use of a knowledge base of legal rationales can improve the system performance in unfairness detection, while improving the reliability of the classification task. The experimental results show that a MANN model using a relatively small set of legal rationales without direct supervision can outperform other neural baselines. Additionally, the employed simple MANN architecture is able to slightly improve over the state-of-the-art SVM.

Moreover, memory-enhanced architectures inherently allow qualitative in-depth analyses of the model's behaviour, facilitating task-related investigations concerning relevant issues, such as trustworthiness and stability. Nonetheless, further steps are required, such as the annotation of memory targets for each input clause, to be exploited both to train the model towards task objectives, and to directly assess behaviour comparison.

The experimental preliminary case study raises up to major challenges. In particular, these challenges mainly concern unstructured knowledge integration in terms of memory usage. More precisely, contrary to traditional settings in which unstructured information is exploited, such as goal-oriented dialogues or question answering, a correct integration of available knowledge is a challenge itself. This is due to potential information incompleteness or due to its inherent high-level and abstract formulation.

#### Optional Memory Usage

Enhancing binary classification by introducing complex legal rationales concerning unfairness naturally opens up several questions concerning memory usage. In particular, albeit expertise knowledge is a useful support tool, there could be cases in which such information is not required at all. For instance, the employed legal rationales only refer to unfair clauses. Nonetheless, their integration via a softmax-based attention mechanism enforces their usage for each input clause.

As a result, an inappropriate knowledge integration mechanism may even hinder the classification due to irrelevant introduced unstructured knowledge that may obfuscate crucial clause information. In other terms, contrary to traditional dialogue-oriented task, available knowledge base may be optional for certain situations. This stand as a necessary gimmick due to the impossibility to properly define an unstructured knowledge capable of covering all possible scenarios.

### Partial Strong Supervision

Providing direct knowledge supervision about which legal rationales should be associated with each input unfair clause is highly desirable. Traditional MANNs scenarios [25, 134, 179, 257, 282] have direct supervision defined for each input example. In contrast, in our case study the integration of unstructured knowledge only affects unfair input examples.

As a result, traditional implementations concerning knowledge supervision may not come as effective as seen in standard settings. Moreover, if knowledge has partial coverage over input data, the integration of direct knowledge supervision is particularly challenging. Thus, ad-hoc solutions have to be considered.

Interestingly, the described scenario can also be beneficial from the point of view of knowledge evaluation. For instance, consider a legal rationale that is frequently associated with unfair clauses. Nonetheless, the model hardly uses that legal rationale. Therefore, we can hypothesize that the natural language formulation of the legal rationale is quite challenging for its integration. Consequently, a modification of the legal rationale could be considered.

## 4.4 Unfair Clause Detection with Partial Strong Supervision

This section extends the work presented in Section 4.3 regarding unfair clause detection with MANNs from several perspectives. The present study relies on a significantly extended dataset with several unfairness categories and related rationales. Additionally, we evaluate multiple MANN configurations, limited to single-hop reasoning regarding task-related assumptions. Furthermore, we explore the benefits of partial direct knowledge supervision, denoted as *strong partial supervision*, from both the classification performance and the model explainability perspectives.

Regarding UKI, we use direct knowledge supervision to measure *correctness* quantitatively. Additionally, we analyze how unstructured knowledge is employed by reporting several memory-oriented statistics. These measures do not explicitly guarantee *coherence* but rather are an additional support tool for evaluating how legal rationales are integrated and used. For instance, we can evaluate the quality of unstructured knowledge from its natural language formulation. Additionally, insights concerning the adopted unstructured knowledge mapping mechanism emerge as well. We can determine the efficacy of the mapping mechanism when input data and unstructured knowledge are considered fixed. Consequently, we can consider extensions regarding the unstructured knowledge mapping mechanism if it cannot address knowledge with partial coverage and partial strong supervision.

We briefly outline the extended dataset with new categories of unfairness and corresponding legal rationales. We then describe the method for unstructured knowledge integration with MANNs. In particular, the MANN models are formulated to support partial knowledge coverage and partial strong supervision. Lastly, we report experimental results and further discussions. More details about the presented work are reported in [223].

### 4.4.1 Data

We employ the same dataset of online consumer contracts, i.e. Terms of Service (ToS), introduced in [136]. As a contributing point, the dataset was further extended by introducing legal rationales for the following categories: Arbitration (A), Unilateral changes (CH), Unilateral termination (TER) and Content removal (CR). Additionally, the sets of legal rationales for the Limitation of liability (LTD) category were merged into a single set and further refined due to overlapping expressions. The distribution of the different categories across the 100 documents is reported in Table 4.4. We shall notice the high frequency of some of the chosen categories within the dataset. Arbitration clauses are the most uncommon and are found in 43 documents only. All other categories appear in at least 83 out of 100 documents. Limitation of liability and unilateral termination together represent more than half of all potentially unfair clauses.

We expected that detecting unfair clauses under these categories would be especially challenging. The state-of-the-art classifier shows lower performance on such clauses than other categories [154]. Additionally, it also turns out that we could match each of these potentially unfair clause categories with several potentially unfair practices/legal rationales. The resulting one–to–many mapping of clauses to legal rationales will be detailed in the following sections. We can use the link between rationales and clauses to instruct the system to explain the unfairness of particular clauses. We limit the listing of legal rationales to the Arbitration (A) unfairness category. The complete legal rationales listing is reported in Appendix C.

#### Arbitration

The arbitration clause can be considered a kind of forum selection clause since it requires or allows the parties to resolve their disputes through an arbitration process before the case could go to court. However, such a clause may or may not specify that arbitration should occur within a specific jurisdiction. While clauses defining arbitration as fully optional has been marked as clearly fair, those stipulating that the arbitration should (1) take place in a state other than the state of consumer's residence and (2) be based not on law but arbiter's discretion were marked

Table 4.4: ToS-100 corpus statistics. For each category of clause unfairness, we report the overall number of clauses, the number of documents they appear in and the average sentence length (i.e. word count).

| Type of clause | # clauses | # documents | average length (# words) |
|---|---|---|---|
| Limitation of liability | 629 | 98 | 47.04 |
| Content removal | 216 | 83 | 39.81 |
| Unilateral termination | 420 | 93 | 36.04 |
| Unilateral changes | 344 | 97 | 26.03 |
| Arbitration | 106 | 43 | 47.55 |

as clearly unfair. In every other case, the arbitration clause was considered potentially unfair. Under this category, we identified eight legal rationales, as reported in Table 4.5. As examples, consider the following clauses taken from the Airbnb (retrieved in November 2019) and Grindr (recovered in July 2018) terms of service:

```
By accepting these Terms of Service, you agree to be bound by this arbitration clause and
class action waiver.

Any Covered Dispute Matter must be asserted individually in binding arbitration
administered by the American Arbitration Association (AAA) by its Consumer Arbitration
Rules (including utilizing desk, phone or video conference proceedings where appropriate
and permitted to mitigate costs of travel).
```

The first clause above, previously classified as potentially unfair, has been linked to the `consent_tos` identifier since it states that the agreement to the Terms of Service is specifically treated as consent to the arbitration clause. Conversely, the second clause was previously classified as clearly unfair, and it has been linked to the following IDs: `arb_obligatory`, `extralegal_rules` since it states that all disputes must be resolved through arbitration, and the consumer is mandatorily subject to rules on dispute resolution not covered by the law, i.e. Consumer Arbitration Rules by the American Arbitration Association (AAA).

Table 4.5: Legal rationales for the Arbitration category.

| **ID** | **Legal Rationale** |
|---|---|
| `arb_obligatory` | `All disputes must be resolved through arbitration, instead of a court of law, and the rights and obligations of the party will be decided by an arbitrator instead of a judge or jury.` |
| `exceptions_apply` | `Arbitration is mandatory though the clause contains exceptions where arbitration is not mandatory or does not apply under certain circumstances; this includes pursuing certain claims in a small claims court.` |
| `extralegal_rules` | `The consumer is mandatorily subject to rules on dispute resolution not covered by law; this includes any rules on arbitration coined by an arbitral body, chamber, association or other type of organization.` |
| `outside_domicile` | `The arbitration is to take place in country different than the consumer's domicile.` |
| `opt_out` | `The consumer must first opt out for the arbitration not to be obligatory.` |
| `unless_-prohibited` | `The arbitration is mandatory unless prohibited by applicable law.` |
| `soft_redirect` | `Disputes which are unresolved informally, through a small claims court or otherwise, may be submitted for arbitration, also on option of one of the parties.` |
| `consent_tos` | `The user is bound by the arbitration clause on grounds of accepting the ToS.` |

### 4.4.2 Method

We formulate the task of potentially unfair clause detection as a standard classification problem, yet enhanced by introducing external knowledge containing possible explanations for the labelled unfairness types. Specifically, we extend the setup described by [136], in which statements to be classified are explicitly compared with the given knowledge. In particular, following the technical formulation of MANNs, we consider input clauses as a query to the memory, which contains a collection of legal rationales.

When a clause $q$ has to be classified, the system attempts to retrieve from memory $M$ those rationales (possibly more than one) that best match such clause. We can compute this match by exploiting a similarity metric $s(q,m_i)$ between the input clause and each memory slot $m_i$. Then, the MANN extracts content from memory $M$, by computing a weighted combination of all the memory slots, where the weight $w_i$ of the $i$-th slot is proportional to the similarity metric: $c = \sum_{i=1}^{|M|} w_i \cdot m_i$. It is worthwhile noticing that, differently from [136], here we compute $w_i$ by applying a sigmoid function $\sigma$ to the output of the similarity function: $w_i = \sigma(s(q,m_i))$. Compared to softmax, a sigmoid activation allows us to view the external memory as an optional source of information, promoting the idea that rationales may not be required for all clauses and enabling the activation of multiple memory slots.

Content $c$ is then combined with clause $q$ to build the final representation $\tilde{q}$ to be used to classify the clause. This query update consists of a simple concatenation of the two vectors in our experimental setting.[2] The MANN finally considers $\tilde{q}$ to predict whether the input clause is potentially unfair. The whole architecture of the system is depicted in Figure 4.4a.

### 4.4.3 Weak and Strong Partial Supervision

We consider two different kinds of supervisions for the use of memory, usually named *weak* and *strong* supervision [252]. Under weak supervision, we feed the MANN with the whole collection of rationales (the KB), without providing the information, during training, of which set of rationales should be used for which clause. Under strong supervision, instead, we provide the MANN with the explanation(s) used for every potentially unfair clause.

Strong supervision of legal rationales encoded in the memory is implemented as a max-margin loss at the extraction level. We can view it as suggesting a higher preference for memory elements, i.e., legal rationales, labelled by experts as the true motivation of given clause unfairness. This added penalty term gives a high cost to examples (unfair clauses) that are not assigned to the corresponding ground-truth explanation(s) in the memory:[3]

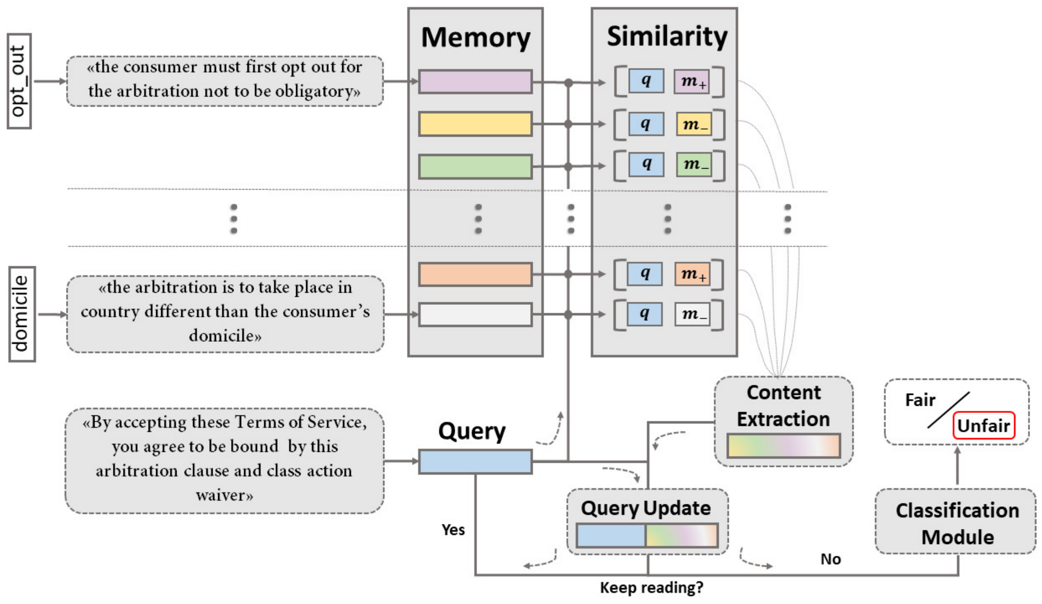
$$\mathscr{L}_{SS} = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{|M_+^n||M_-^n|}\sum_{m_+\in M_+^n}\sum_{m_-\in M_-^n}[\mathscr{L}(m_+,m_-)] \tag{4.1}$$

$$\mathscr{L}(m_+,m_-) = \max\Big(0,\gamma-\sigma(s(q^n,m_+))+\sigma(s(q^n,m_-))\Big) \tag{4.2}$$

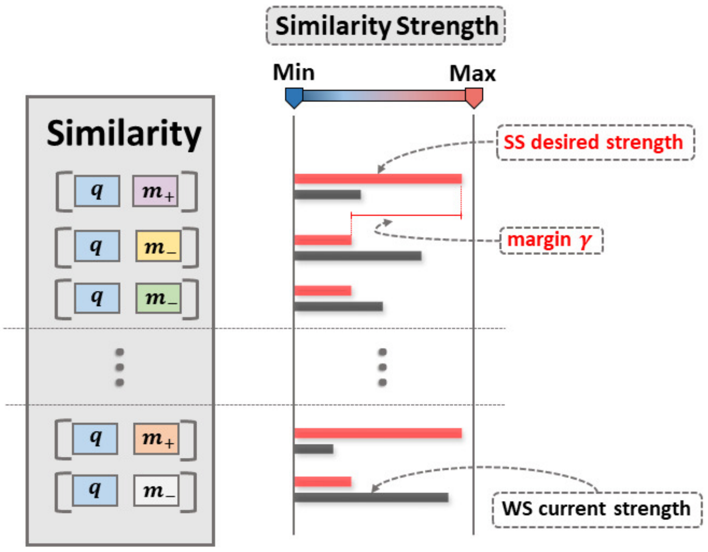
where $M_+^n$ is the set of target explanations for a given example $q^n$, $M_-^n$ is the set of non-target explanations for $n$, $(m_+,m_-)$ is a target/non-target explanation pair for a given sample $n$, and $\gamma$ is the margin hyper-parameter. Intuitively, this loss function pushes the MANN to compute scores for the larger target explanations than those for non-target explanations for at least a margin $\gamma$. By controlling the intensity of this preference via a sort of regularization coefficient, we can trade-off between classification performance and model output interpretability (see Figure 4.4b). In fact, in our scenario, properly motivating the proposed model predictions is as crucial as evaluating its performance by standard classification metrics. As already mentioned,

[2]The general MANN architecture would also allow for multiple iterations of memory reading operations. We leave to future research a deep investigation of this possibility.

[3]It is worthwhile noticing that each clause in the training set could be associated with multiple justifications. We consider the memory to be correctly used if at least one of the supervision rationales is selected.


Memory
Similarity
opt_out
«the consumer must first opt out for the arbitration not to be obligatory»
domicile
«the arbitration is to take place in country different than the consumer's domicile»
«By accepting these Terms of Service, you agree to be bound by this arbitration clause and class action waiver»
Query
Content Extraction
Query Update
Fair
Unfair
Classification Module
Yes
No
Keep reading?
q
m+
m−

(a)



(b)

Figure 4.4: (a) MANN general architecture for unfair clause detection. Given a clause $q$ to classify and a set of legal rationales, i.e., memory $M$, the model first evaluates the similarity between $q$ and each memory slot $m_i$. Subsequently, proportionally to their similarity with $q$, memory components $m_i$ are aggregated into a single vector representation. Intuitively, if the similarity metric is non-zero only for a single $m_i$, the aggregated vector is a weighted representation of $m_i$. Lastly, such aggregated vector is used to update $q$ to enrich input information for the final classification step. (b) Weak and strong supervision. We can guide the parametric similarity step by specifying which memory content $m_i$ should have high similarity with $q$. This process is formally known as strong supervision (SS) in the MANN literature. In particular, the model is instructed to shift from its current similarity behaviour (black bars) to the one desired by SS (red bars).

introducing a KB comprised of expertise rationales allows us to better interpret the model output by considering its linkage to the KB.

### 4.4.4 Experiments

For each unfairness category introduced in Section ,4.4.1 we employed sets of expert justifications, each of a different size. In particular, we consider each category as a standalone binary classification task, where model evaluation is defined as a standard 10-fold cross-validation.

As it is well-known that standard MANN architectures are affected by high variance [276], three networks are trained with different random seed initialization for each fold in the cross-validation. We pick the best according to the performance on the validation set, as customary in many applications [14].

As for the deep architecture, we employ a simple variant of the traditional end-to-end MANN [252], with the following modifications, already introduced in Section 4.4.2:

(i) Attention operation via sigmoid activation function instead of softmax to enable the selection of multiple (possibly none) memory slots;

(ii) The number of reading iterations over memory limited to just one, which also simplifies training;

(iii) Query update via concatenation instead of embedding sum;

(iv) Clause-justification similarity operation via a two-layer MLP, to further consider strong supervision as an additional penalty term.

The last two architectural choices follow a preliminary experimental evaluation and make the model more expressive. In particular, we have defined a hyper-parameter calibration step on the validation set via Bayesian optimization (with the `hyperopt` package). We tested different query update variants: sum, concatenation, computation via an MLP or an RNN. For the similarity operation, we have tested the following configurations: dot product, scaled dot-product, from 1 up to 3 dense layers with units belonging to the set $[64, 128, 256, 512]$.

For each test fold in the cross-validation, the remaining data are split between 80% for training and 20% for validation. Training is then regularized by early stopping on validation $F_1$, with patience equal to 40 epochs.

As for performance comparison, we consider the current state-of-the-art solution implemented in the CLAUDETTE system [154], based on support vector machines (SVM). Additionally, we consider a set of deep neural networks that do not leverage any external information: (i) a simple set of stacked convolutional neural networks (CNNs), on the top of which a 2-layer

Table 4.6: Classification performance of the employed models according to macro-F1 computed on 10-fold cross-validation for unfair examples. For MANN, WS and SS stay for weak and strong supervision, respectively.[4]

| | **Categories** | | | | |
|---|---|---|---|---|---|
| **Model** | A | CH | CR | LTD | TER |
| SVM | 0.350 | **0.673** | 0.538 | 0.636 | 0.636 |
| CNN | 0.361 | 0.654 | 0.584 | 0.627 | 0.612 |
| LSTM | 0.326 | 0.639 | 0.498 | 0.589 | 0.589 |
| MANN (WS) | 0.503 | 0.670 | 0.596 | 0.649 | 0.664 |
| MANN (SS) | **0.526** | 0.665 | **0.606** | **0.659** | **0.666** |

MLP classifier is added for binary classification; (ii) a set of stacked recurrent neural networks along with a 2-layer MLP for classification.

We set the hyper-parameters of each model, including MANNs, by selecting the architecture with the best performance on the validation set. In particular, for MANNs, we chose a word embedding size of 256, an $L2$-regularization weight of $10^{-3}$, a dropout equal to 0.7, an MLP with a hidden layer of 32 neurons for the similarity score between clause and memory, and 64 neurons for the final classification layer. For strong supervision, we set the margin $\gamma = 0.5$ to ensure that uncorrelated legal explanations are irrelevant for classification, while we tuned the penalty coefficient separately for each unfairness category.

We present two different tables for results. In Table ,4.6 we report the standard classification metrics, namely the macro-average $F_1$ score computed on the cross-validation procedure, evaluated on the five unfairness categories of interest. Table 4.7 instead reports several statistics about memory usage and related classification performance when considering weak and strong supervision. In particular, we compute the following metrics:

**memory usage (U):** the percentage of input examples (clauses) for which memory is used;

**coverage (C):** the percentage of correct memory slot selection over all unfair examples;

**coverage precision (CP):** the percentage of correct memory slot selection over examples where memory is used.

[4]The reported performance is not directly comparable with the work of [136] for two different reasons: (1) the corpus is different, as it consists of a more extensive (and more heterogeneous) collection of 100 contracts; (2) the task here is a binary classification for each separate category, whereas [136] addressed the (more straightforward) problem of detecting potentially unfair clauses (of any category).

Table 4.7: Memory statistics concerning model predictions on unfair examples only. Memory interaction is evaluated on the test set only. Several metrics are reported to assess the extent to which legal rationales are of use and exclude ill-based performance. In particular, the following metrics are considered: memory usage (U), correct memory usage over unfair examples (C) and over examples for which memory is used (CP), along with a more fine-grained ranking version (CP@1-3), correct classification when memory is employed (MP) and, lastly, the average memory usage per sample (APM). Models are compared with baselines that always select the most (@1) or second-most (@2) frequent legal explanation.

| **Model** | **U** | **C** | **CP** | **CP@1** | **CP@3** | **MP** | **APM** | **MRR** |
|---|---|---|---|---|---|---|---|---|
| Arbitration (A) | | | | | | | | |
| Baseline@1 | 1.0 | 0.698 | 0.698 | 0.698 | 0.698 | 0.698 | 0.125 | / |
| Baseline@2 | 1.0 | 0.358 | 0.358 | 0.358 | 0.358 | 0.358 | 0.125 | / |
| MANN (WS) | 0.292 | 0.283 | 0.968 | 0.548 | 0.839 | 0.968 | 0.839 | 0.673 |
| MANN (SS) | 0.547 | 0.500 | 0.914 | 0.741 | 0.897 | 0.862 | 0.500 | 0.876 |
| Unilateral changes (CH) | | | | | | | | |
| Baseline@1 | 1.0 | 0.837 | 0.837 | 0.837 | 0.837 | 0.837 | 0.143 | / |
| Baseline@2 | 1.0 | 0.212 | 0.212 | 0.212 | 0.212 | 0.212 | 0.143 | / |
| MANN (WS) | 0.526 | 0.445 | 0.845 | 0.210 | 0.757 | 0.459 | 0.752 | 0.417 |
| MANN (SS) | 0.872 | 0.805 | 0.913 | 0.850 | 0.877 | 0.693 | 0.454 | 0.896 |
| Content removal (CR) | | | | | | | | |
| Baseline@1 | 1.0 | 0.546 | 0.546 | 0.546 | 0.546 | 0.546 | 0.059 | / |
| Baseline@2 | 1.0 | 0.435 | 0.435 | 0.546 | 0.546 | 0.435 | 0.059 | / |
| MANN (WS) | 0.074 | 0.051 | 0.688 | 0.250 | 0.563 | 0.688 | 0.482 | 0.271 |
| MANN (SS) | 0.148 | 0.093 | 0.625 | 0.375 | 0.500 | 0.625 | 0.292 | 0.452 |
| Limitation of liability (LTD) | | | | | | | | |
| Baseline@1 | 1.0 | 0.676 | 0.676 | 0.676 | 0.676 | 0.676 | 0.056 | / |
| Baseline@2 | 1.0 | 0.423 | 0.423 | 0.423 | 0.423 | 0.423 | 0.056 | / |
| MANN (WS) | 0.176 | 0.144 | 0.818 | 0.264 | 0.564 | 0.818 | 0.581 | 0.351 |
| MANN (SS) | 0.174 | 0.145 | 0.835 | 0.321 | 0.624 | 0.835 | 0.586 | 0.414 |
| Unilateral termination (TER) | | | | | | | | |
| Baseline@1 | 1.0 | 0.388 | 0.388 | 0.388 | 0.388 | 0.388 | 0.036 | / |
| Baseline@2 | 1.0 | 0.229 | 0.229 | 0.229 | 0.229 | 0.229 | 0.036 | / |
| MANN (WS) | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| MANN (SS) | 0.998 | 0.631 | 0.632 | 0.387 | 0.492 | 0.499 | 0.136 | 0.531 |

These metrics ascertain whether knowledge integration is effective and to what extent. Moreover, we report a classification-oriented score concerning prediction accuracy with non-zero memory usage, namely memory precision (MP).

Table 4.8: Qualitative evaluation of strong supervision concerning two examples of different unfairness categories. The leftmost column shows the category. The rightmost column reports the target set (**Targets**) associated with given clauses and the legal rationales selected by the model during the inference of two MANNs trained with strong (**SS**) and weak supervision (**WS**). In these examples, the contribution of the strong supervision leads to more content-aware and selective classification, providing at the same time a qualitative tool for model interpretability.

| **Cat.** | **Clause** | **Targets/SS/WS** |
|---|---|---|
| A | You and Airbnb mutually agree that any dispute, claim or controversy arising out of or relating to these terms or the breach, termination, enforcement or interpolation thereof, or to the use of the Airbnb platform, the host services, or the collective content (collectively, "disputes") will be settled by binding arbitration (the "arbitration agreement" | **Targets**: `exceptions_apply`<br>**SS**: `exceptions_apply`<br>**WS**: ∣ |
| CH | [...], additionally, there may be times when we need to remove or change features or functionality of the service or stop providing a service or access to third-party apps and services altogether | **Targets**: `anyreason`<br>**SS**: `anyreason`<br>**WS**: `anyreason`, `nowarning`, `justponed`, `imbalance`, `consresp`, `againsterms`, `lawchange`, `update` |

However, more detailed information about memory selection is required to exclude ill-behaved scenarios, such as where the model uses all available memory, and avoid rushed positive conclusions about memory usage even though the model mostly prefers non-target justifications. To this end, we also report metric CP when only the memory slot with the highest score—that is, the most preferred explanation—is considered (CP@1) as well as the three slots with the highest scores are considered (CP@3). Finally, we report the average percentage of used memory the average number of memory elements selected concerning total memory size (APM).

The results in Table 4.6 show that even a naive combination of a simple MANN architecture and raw knowledge representation is sufficient to show increased performance over traditional knowledge-agnostic models, such as the proposed neural baselines and the current state-of-the-art SVM solution. However, solely relying on classification performance is a non-exhaustive evaluation criterion to assess whether a model is better. Weak supervision is already sufficient to slightly enhance performance, whereas strong supervision still adds a margin of improvement. More specifically, more challenging unfair detection scenarios, such as A and CR, show a considerable benefit of added knowledge that compensates for the high-class distribution unbalance of these settings.

However, different factors represent a challenger for unstructured knowledge integration. Notable examples are noisy knowledge representation, many possible explanations of unfairness, and increased available data. Hence, the performance improvement over the other interest categories appears to be only moderate. In particular, LTD and TER categories present the highest explanation variation, increasing the complexity of knowledge formulation for a correct performance boost.

As for the accuracy in providing explanations, which is the main point of adopting a MANN architecture, Table 4.7 confirms our intuition that strong supervision offers a crucial element in correctly linking potentially unfair clauses to their correct rationales. Table 4.7 clearly shows that the CP metric is very high when strong supervision is applied, ranging from 0.63 (CR and TER) up to 0.8 and 0.9 (A, CH, LTD). The top-scoring rationale is often correct for A and CH (above 0.7 and 0.8, respectively). For all the categories, the correct explanation is among the top-3, scoring in over 50% of the cases (around 0.9 for A and CH). The APM metric also shows that the MANN is quite selective in choosing the memory: in fact, the percentage of used explanations never exceeds 50% of the memory size, on average.

Lastly, to further illustrate the benefits introduced by strong supervision, Table 4.8 shows two examples of proper and selective memory usage that lead to correct unfairness detection. In particular, not only is the model encouraged to use target legal rationales for the given example, but it also learns to use such added knowledge attentively. On the other hand, a model without such direct supervision may fail to filter out irrelevant information or select relevant information for classification. Supplementary material regarding the contribution of partial strong supervision is reported in Appendix D.

### 4.4.5 Discussion

This second case study concerning unfair clause detection confirms that integrating unstructured textual knowledge into data-driven methodologies is challenging. This is particularly evident in legal analysis where providing an interpretable and transparent unstructured knowledge integration process is desired. In particular, the integrated knowledge is significant, broad and noisy. We extensively analyzed simple formulations for unstructured knowledge integration and mapping processes. In particular, the provided analysis concerning the adoption of partial strong supervision sheds light on the difficulty of defining a successful UKI process.

Different analyses have to be carried out to evaluate the integration of unstructured knowledge properly. Here, we have only scratched the surface regarding the unstructured knowledge mapping mechanism. Additionally, more sophisticated mapping mechanisms, model architectures and, more importantly, knowledge validation criteria have to be explored. An additional issue is scalability. Being able to rely on several sources of explanation may provide high flexibility, easing cross-domain adaption. However, incorporating large sets of legal rationales requires approximation methods at the memory addressing level [36, 189], depending on the given model architecture.

Other areas of legal analytics of far higher complexity have yet to be explored. Notable examples are legal argument analysis [153] and privacy policies [45]. In particular, in the latter scenario, the high structural complexity of legal documents demands richer task formulations than just sentence-level information. For instance, intra-document contextual information is crucial to assess the correctness of a legal statement. The integration of legal rationales for unfair clause detection remains a challenging task where tight constraints like model interpretability are particularly important. We briefly outline in the below sections some research directions worth exploring.

### Dynamic Knowledge

The conditioning of knowledge on the data-driven learning process is not new [114] and has also been explored in the context of MANNs [282]. However, any modification process regarding a knowledge type (KT) should be compliant with the selected knowledge integration and mapping mechanisms. For instance, the newly added unstructured knowledge modification process should maintain the validity of the provided knowledge. Thus, the proposed data-driven unstructured knowledge modification process should be somewhat constrained to not alter the knowledge itself in a significant way.

Nonetheless, automatically determining violations in the modification process is non-trivial. For instance, in the case of legal rationales, we should consult legal experts periodically to evaluate the nature of the unstructured knowledge modification process. However, such a requirement may not be feasible at all. Despite the problem of defining a formal unstructured knowledge validation process, a controlled unstructured knowledge modification process could significantly aid model learning. Indeed, in our experimental scenario of unfair clause detection, some legal rationales are of poor quality and, thus, hardly utilized throughout learning.

### Zero- And Few-Shot Learning with Legal Rationales

Formulating the task of unfair clause detection as a zero- or few-shot learning task is of particular importance for two main reasons: (i) it facilitates extending the task to new sets of legal rationales and new unfairness categories; (ii) it explicitly enforces the process of unstructured knowledge mapping throughout learning. Regarding the first point, we aim at evaluating classification models on unseen unfairness categories with their corresponding set of legal rationales. Such capability eases defining new legal rationales and automatically modifying existing ones. By doing so, no additional training phases specialized for each unfairness category may be required. Instead, a unique model is preferred to adapt to a new set of legal rationales.

Lastly, relying on legal rationales for classification enforces the model to use the provided unstructured knowledge explicitly. This process also allows a critical approach to evaluating the quality of legal rationales based on model performance. Nonetheless, a different KT for this task should be considered. In particular, adequate class descriptors for each unfairness category are preferred. Consequently, we should discuss the difficulties of determining these class descriptors from available legal rationales.

## 4.5 Sampling from Unstructured Knowledge with Transformers

This section explores two challenging research directions to assess the contribution of unstructured knowledge integration in aiding a model in addressing a particular task. More precisely, we consider classification scenarios with (i) limited training data and with (ii) large knowledge bases. As for classification tasks, we explore the task of argument sentence detection [225] and pick the task of unfair clause detection [136, 223], once again. We then formulate the experimental setup for text classification with limited training data by considering a smaller version of two datasets, one for argument sentence detection and the other for unfair clause detection. In the context of argument sentence detection, such a choice is also aligned with the size of the knowledge to be integrated as it scales with the size of the data.

We consider the same setup of [223] for unfair clause detection regarding unstructured knowledge integration. That is, legal rationales define unstructured knowledge. On the other hand, unstructured knowledge for argument sentence detection comprises textual examples that are highly correlated with the ones to classify. We view these textual examples to be integrated as a contextual data knowledge type (KT) (Section 2.6.6). To account for the extensive knowledge size, we explore knowledge sampling strategies that can adapt to the underlying data-driven learning approach. We condition how knowledge is retrieved for a particular input example based on its usefulness for the task. Additionally, we explore advanced language understanding models like BERT [58] within the framework of MANNs to address the task of text classification with UKI.

We initially describe our methodology concerning knowledge sampling and introduce a memory-augmented version of DistilBERT [228] and BERT [58], denoted as MemDistilBERT and MemBERT, respectively. Lastly, we report experimental results and discussions. More details about this work are reported in [224].

### 4.5.1 Method

We propose a method for combining transformers with natural language explanations to enhance interpretability without creating a knowledge acquisition bottleneck. This approach could be described as *learning from textual descriptions*, and it resembles certain human learning processes. Importantly, we achieve this result without hampering classification performance.

#### Knowledge as an External Memory

Dealing with knowledge expressed in natural language requires ad-hoc methodologies that can efficiently use knowledge while maintaining desired properties like model interpretability. Past research on deep learning models tackled the problem of handling external content by introducing external memory blocks the model could interact with in a differentiable way [251]. Such *memory-augmented neural models* enabled complex reasoning tasks like reading comprehension and question answering and opened a new path in the context of meta-learning [291].

Therefore, they seem an ideal candidate architecture for the integration of natural language explanations. However, the main purpose of our extension of transformers is not to improve classification performance, but to generate explanations in the form of grounding to elements of a domain knowledge. Accordingly, the knowledge stored in our memory is not required, strictly speaking, to correctly address the task (as, for example, in question answering). It is instead an auxiliary collection of pieces of information that can be consulted to perform reasoning and to make the neural model interpretable.

#### Memory-Augmented Transformers

The main building block of our architecture is a transformer-based model. In our experiments we considered BERT [58] and DistilBERT [228], a distilled version of BERT that achieves competitive performance while limiting the overall computational burden. However, our approach is general and is not restricted to these two architectures. Memory-augmented neural network architectures [92, 276] (MANNs) extend neural networks with an external memory block that supports reasoning. We denote our memory-augmented transformer models as *memBERT* and *memDistilBERT*. Memories are loaded with relevant background information. Our architecture (see Figure 4.5) follows the typical structure of MANNs:

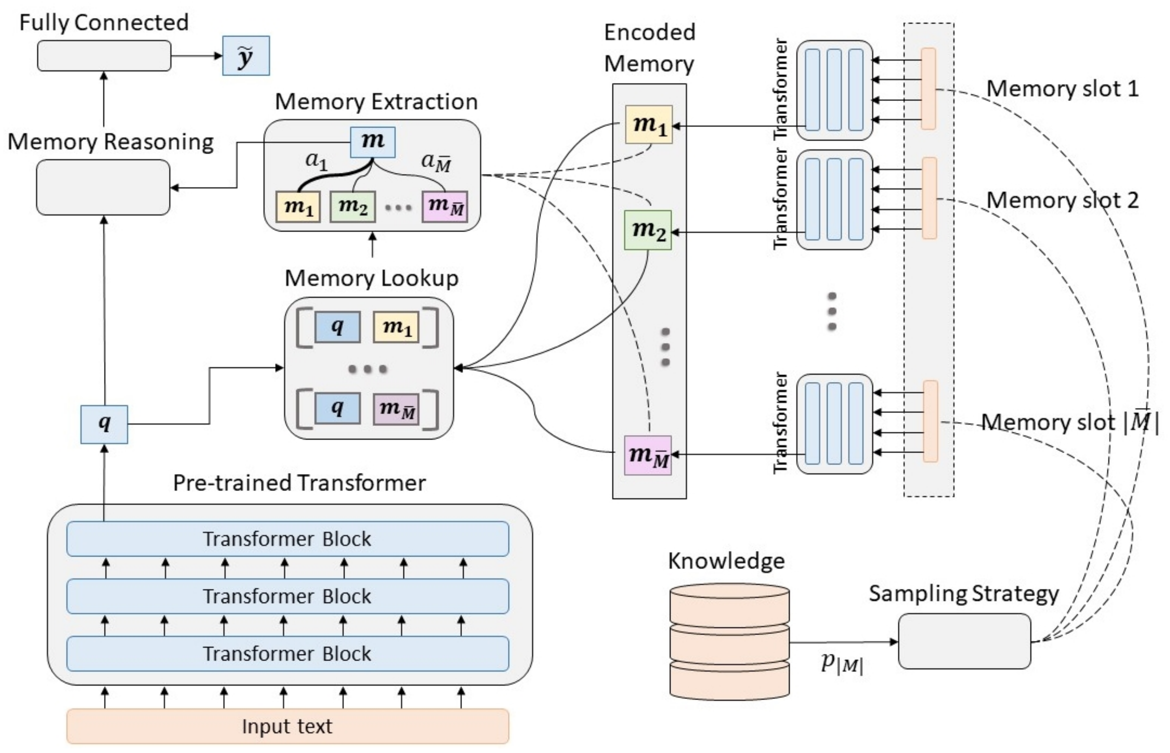


Figure 4.5: Memory-augmented transformer architecture. $M$ denotes the whole knowledge, $\bar{M}$, the part of the knowledge loaded in the memory after a possible sampling step. $|\cdot|$ denotes cardinality.

(i) a transformer layer encodes the input texts, including memories, producing a set of sentence embeddings;

(ii) a memory-lookup layer compares the memory content with the input, producing a set of similarity scores;

(iii) a memory extraction layer converts such similarity scores into attention scores $\{a_1, \ldots, a_M\}$ and uses them to compute a single memory summary embedding vector via a weighted sum of the memory slots;

(iv) a memory reasoning layer uses the memory summary embedding vector to update the input embedding vector. For instance, a simple form of input update consists in summing input and memory summary embedding vectors together;

(v) a fully-connected layer uses the updated vector as an input for the classification task.

Being $N$ the number of training examples and $C$ the set of classes, all models are trained to minimize standard cross-entropy loss:

$$\mathscr{L}_{CE} = -\sum_{i=1}^{N}\sum_{c=1}^{C} p(y_i = c)\log p_\theta(y_i = c) \tag{4.3}$$

In this setting there is no information about which is the correct memory slot to be applied to a given example. In the literature, this is known as Weak Supervision (WS).

### Guiding Memory Interaction with Strong Supervision

The concept of WS is important, since not always single memory slots can be directly associated with individual training examples (e.g., this might be costly with regard to the time required from an expert, but also intrinsically challenging). This is a very general setting. Yet, if background knowledge comes with the capability of naturally linking each example with the associated memory content, it is then possible to guide the model in the training phase by only focusing on specific target memory slots. This procedure is formally known as Strong Supervision (SS) and it has been widely explored in memory networks since their introduction [251]. SS can assume different formulations.

In our experimental scenarios, it is sufficient to enforce a preference for just one of the (possibly multiple) memory slots associated with a given input example. Following [223] we exploit SS in the form of a max-margin loss. We define target memory slots as those that have been linked to the input example during the annotation phase. We introduce a penalty term that enforces target memory slots to have a higher similarity score with respect to the input $x$ than to the remaining slots, up to a $\gamma$ margin:

$$\mathscr{L}_{SS} = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{|M_+^n||M_-^n|}\sum_{m_+\in M_+^n}\sum_{m_-\in M_-^n}[\mathscr{L}(m_+,m_-)] \tag{4.4}$$

$$\mathscr{L}(m_+,m_-) = \max\Big(0, \gamma - \sigma(s(x,m_+)) + \sigma(s(x,m_-))\Big) \tag{4.5}$$

where for the $n$-th example $M_+^n$ is the set of target memory slots for a given input example $x$, $M_-^n$ is the set of non-target memory slots for the $n$-th input example, $(m_+,m_-)$ is a target/non-target memory slot pair for $n$, $\sigma$ is the activation function and $s(\cdot,\cdot)$ is the similarity function. The hyper-parameter $\gamma$ controls the intensity of the SS constraints, while trading-off classification performance and model interpretability: large values of $\gamma$ would basically turn the preference ranking induced by the hinge loss into a classification loss. Overall, the loss function for memBERT and memDistilBERT with SS is:

$$\mathscr{L} = \mathscr{L}_{CE} + \mathscr{L}_{SS} \tag{4.6}$$

### Efficiently Handling Large Memories with Sampling

In the architecture described so far, efficiently handling large memories is particularly challenging, for two main reasons. First, as the size of the memory grows, the number of irrelevant (if not noisy) memory slots for a given input example inevitably grows as well. Second, the complexity of the memory-related layers (steps (ii)–(iv) in Section 4.5.1) also increases with the size of the memory. To this end, approximate solutions or sampling-based methods have been explored, trading-off performance with efficiency [36]. Typical solutions mainly exploit simple comparison strategies, like dot product similarity, that are easily executed in parallel and approximated.

In our case, we aim to define a more complex memory lookup phase. Therefore, we allow neural-based similarities and strive to look for an efficient sampling strategy to achieve scalability. Memory sampling brings about both benefits and, possibly, issues. Reducing the overall memory size enables large knowledge integration when dealing with complex deep neural models like transformers. Moreover, sampling eases the knowledge integration process from the point of view of noise compensation, i.e., filtering out irrelevant memory content. In a similar fashion to curriculum learning [15], sampling reduces complexity by considering a smaller memory, which, in turn, induces a limitation of spurious lookup operations. This is also reflected at training time, where in each batch the model sees a different memory, thus avoiding to always focus on specific slots, which could possibly head to over-fitting.

Nonetheless, sampling inherently introduces variance, therefore compatibility issues with other strategies like SS might arise as well. To compensate for both problems, we define smart sampling strategies that prioritize memory content according to the observed input examples. To this end, similarly to prioritized experience replay (PER) [231], introduced in reinforcement learning, we introduce priority-based sampling strategies that progressively take into account the importance of the added information with respect to both input examples and task objectives. In particular, to efficiently deal with large memories, we operate at batch level by sampling a shared knowledge base. Then, depending on the given strategy, priority weights $p_{m_i}$ associated to each memory slot $m_i$ are updated and available for the next batch step. More precisely, we adopt the same priority definition of PER, where the temporal difference (TD) error is replaced by a custom importance assignment function:

$$p_{m_i} = (w_{m_i} + \varepsilon)^{\alpha} \tag{4.7}$$

where $w_{m_i}$ is the importance weight of the $i$-th memory slot $m_i$, $\varepsilon$ is a sufficiently small real value that avoids zero-based exponentiation and $\alpha$ controls the priority degree. Intuitively, $\alpha$ equals to 0 produces all priorities equal to 1, thus defining a uniform sampling strategy. Lastly,

**Algorithm 1:** General Training Sampling Procedure

```
    Input: Data $D = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$,
           Memory priority weights distribution $p_{|M|}$,
           Model weights $\theta$
 1  Initialize model weights $\theta = \theta^0$
 2  Initialize memory priority weights $p_{|M|} = p^0_{|M|}$
 3  repeat
 4      Sample a minibatch $B = (\mathbf{X}, \mathbf{Y}) \subset D$
 5      Sample memory $\bar{M}$ using $p^{k-1}_{|M|}$
 6      Compute model loss $\mathscr{L}(x|\bar{M})$ on minibatch $B$
 7      Update model parameters $\theta^k$ using any optimizer
 8      Compute memory importance weights $w^k_{|M|}$
 9      Update memory priority weights $p^k_{|M|} = (w_{m_i} + \varepsilon)^\alpha$
10      Normalize $p^k_{|M|}$ to get corresponding distribution
11  until stopping criteria
12  Save memory priority weights $p_{|M|}$
    Output: Trained model weights $\theta$, trained sampler priority weights $p_{|M|}$
```

**Algorithm 2:** General Inference Sampling Procedure

```
    Input: Data $D = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$,
           Trained memory priority weights distribution $p_{|M|}$
 1  repeat
 2      Get next inference minibatch, $B = (\mathbf{X}) \subset D$
 3      Sample memory $\bar{M}$ using $p_{|M|}$
 4      Save model predictions $\tilde{\mathbf{Y}}$
 5  until $B \in \varnothing$
    Output: Model predictions and sampled memory $\{(\tilde{\mathbf{y}}_n, \bar{\mathbf{M}}_n)\}_{n=1}^N$
```

sampling is applied both at training and at inference time, when learnt priority weights are employed to sample the memory. Algorithm 1 describes the training sampling procedure, while Algorithm 2 describes the inference sampling procedure. Introduced strategy variants differ on how priority weights are updated at each batch step.

**Uniform Sampling** As a baseline strategy we consider uniform memory sampling. At each batch, each memory slot has the same probability of being selected. Priorities are kept fixed during training. Formally, being $|M|$ the memory size, we define the probability of sampling each memory slot $m_i$ as $p_{m_i} = \frac{1}{|M|}$.

**Priority-based Sampling** Intuitively, uniformly sampling from memory immediately relieves us from the problem of handling a large knowledge base, but it does not take into account the importance of each slot. In order to do that, at training time, each individual input example should be able to modify the underlying sampling distribution by giving more priority to those memory slots that have been reputed useful. We explore two different formulations of priority that ground the notion of usefulness to a specific architectural property of the model. More precisely, we consider the memory attention layer as the major indicator of importance of each memory slot with respect to the given input example, considering two variants: (i) the first one relies exclusively on the attention value of each memory slot, whereas (ii) the second formulation also takes into account the contribution of each memory slot to the overall objective of training (i.e., the classification loss).

**Attention-Based Priority-based Sampling** In this formulation, the priority associated to each memory slot during sampling is based only on the corresponding attention value computed by the memory lookup layer, given an input example. In particular, working with batches inherently requires to aggregate each memory slot importance at batch level. To this end, we consider a *masked average* based aggregation, which considers the mean attention value of each memory slot for a certain class of input examples. This strategy accounts for noise introduced by input examples for which no target memory slot exists. For instance, in the case of a binary classification task where memory contains valuable information for the positive class only, we should limit sampling priority update to positive examples, to avoid spurious memory selections. Formally, the attention-based priority sampling defines each memory slot importance as its summary attention value in the batch.

$$w_{m_i} = \frac{\sum_j^{|B|} a_{j,i}}{\sum_j^{|B|} \mathbb{1}_{y_j \in \mathbf{Y}_+}} \tag{4.8}$$

where $a_i$ is the attention value attributed to memory slot $i$, $|B|$ is the batch size and $\mathbf{Y}_+$ is the set of examples in the batch belonging to the positive class. Note that this strategy is purely similarity-based and it does not take into account the contribution of each memory slot to the task-specific objective. In other terms, it operates under the assumption that each knowledge descriptive concept contained in the memory should have high semantic and syntactic similarity with its associated input examples. Therefore, we argue that, in cases where input examples have high similarity with unrelated memory slots, the proposed approach might be not appropriate, as it would aim to perform the task at hand by combining the input with irrelevant information.

**Loss Gain Priority Sampling** In order to render priority sampling aware of the usefulness of each memory slot to the task itself, we consider an additional variant that directly takes into account the classification loss. In particular, we consider the loss difference between the model architecture and the one without the memory layer. By doing so, we are able to estimate the informative gain introduced by the external knowledge with respect to the task. The larger is the difference, the larger is the impact of the memory. To associate a specific priority value to each memory slot $m_i$, we weigh the computed difference by the corresponding attention value $m_i$. We consider the same reduction operation strategy adopted for attention-based priority sampling. Formally, this loss gain priority sampling can be formulated as follows:

$$w_{m_i} = \frac{\sum_j^{|B|} a_{j,i} \exp^{(\mathscr{L}_{CE}(x_j) - \mathscr{L}_{CE}(x_j|\bar{M}))}}{\sum_j^{|B|} \mathbb{1}_{y_j \in \mathbf{Y}_+}} \tag{4.9}$$

where $a_i$ is the attention value attributed to memory slot $i$, $\bar{M}$ is the sampled memory, $\mathscr{L}_{CE}(x)$ and $\mathscr{L}_{CE}(x|\bar{M})$ are the cross-entropy loss obtained without any interaction with the memory, or with the introduction of the sampled memory $\bar{M}$, respectively. The exponential guarantees a positive priority while awarding positive differences with high values and down-weighting negative differences.

### 4.5.2 Experimental Setup

To test our approach, we consider two distinct classification scenarios. In the first, we extend the work in [223] focusing on the legal domain for unfair clause detection in online ToS, by employing explanations given by legal experts as domain knowledge. In the second scenario, we explore the argument mining case study of claim detection by considering evidence as the source of knowledge, following the assumption that class correlation might ease the classification task.

These scenarios motivated some architectural choices regarding the usage of memory. Typically, when memory slots contain connected segments of text, steps $(ii) - (iv)$ described in Section 4.5.1 are repeated multiple times to capture sequential information. However, since our memory slots contain independent fragments, we run these steps just once. In the MANN jargon, this means setting the number of hops to 1.

Both our case studies have input examples associated with multiple target memory slots, as well as examples associated with none. For example, a negative result (a clause that is not unfair) has no associated justification. For these reasons, we adopt sigmoid-based attention scores instead of softmax-based ones.

**Knowledge as Class Descriptions: Unfairness Detection**

Online Terms of Service (ToS) often contain clauses that are potentially unfair to the consumer. Unfair clause detection in online ToS is a binary classification task where each clause is labeled as either fair or unfair [154]. Multiple unfairness categories result in multiple binary classification tasks. Recent proposals [223] address this problem by using legal expertise in the form of *rationales*. Because rationales define, in a way, the unfairness category, they can be used as explanations of the unfairness predictions. In our setting, the input text is a clause to be classified, and the shared memory contains all the possible legal rationales.

**Data** The corpus originally published in [223], named ToS-100, contains 100 documents. It defines legal rationales for five unfairness categories following a multi-label perspective (i.e., a clause can be unfair according to more than a single category). These categories are:

**A** arbitration on disputed arising from the contract;

**CH** the provider's right to unilaterally modify the contract and/or the service;

**CR** the provider's right to unilaterally remove consumer content from the service, including in-app purchases;

**LTD** liability exclusions and limitations;

**TER** the provider's right to unilaterally terminate the contract.

To assess the ability of our architecture to cope with limited training data, we randomly extracted from ToS-100 30 documents, while ensuring that each document contained at least one example per unfairness category. We name this corpus ToS-30. Table 4.9 reports some statistics on ToS-30, showing the number and percentage of unfair clauses per category and the amount of corresponding legal rationales defined by experts. More technical details concerning the definition of both the unfairness categories and the rationales can be found in [223].

**Setting** As in [223], we consider each unfairness category individually. For a given unfairness category, we consider as positive examples those who have been labeled as being unfair for the given category, while the remaining sentences are viewed as not unfair.[5] We consider the following neural baselines: (i) 2-layer CNN network; (ii) 1-layer bi-LSTM network; (iii) MANN baseline defined in [223], which computes the sentence embedding as an average

[5]Note that the corpus labeling procedure distinguishes between potentially unfair and clearly unfair clauses, but following [223] we merged these categories together.

Table 4.9: Corpus statistics for ToS-30. For each category of clause unfairness, we report the number and percentage of unfair clauses, and the number of rationales.

| Type of clause | # unfair clauses | % unfair clauses | # rationales |
|---|---|---|---|
| Arbitration (A) | 45 | 0.8 | 8 |
| Unilateral change (CH) | 89 | 1.7 | 7 |
| Content removal (CR) | 58 | 1.1 | 17 |
| Limitation of liability (LTD) | 161 | 3.0 | 18 |
| Unilateral termination (TER) | 121 | 2.3 | 28 |

over word embeddings. Neural baselines use 512-dimensional embeddings, except for MANN which, following [223], uses 256-dimensional embeddings. All the embeddings are learnt from scratch.[6] We use a single-layer MLP-based memory-lookup implementation with 32 units. The reasoning layer is a concatenation of input and memory summary embedding vectors. Additionally, models are further regularized with $L^2$ penalty with $10^{-5}$ weight, dropout rate in the $[0.5, 0.7]$ range and early-stopping based on the validation F1-score, with a patience equal to 10 for BERT models[7] and to 50 for other neural baselines. For optimization we use Adam, with $10^{-3}$ learning rate. For evaluation, we employ a 10-fold cross-validation procedure with multi-start with 3 repetitions, choosing the best one according to the validation F1-score.

### Knowledge as Supporting Facts: Claim Detection

The second classification scenario is claim detection, which is an argument mining task [156]. We formulate the problem as a binary classification task where each input sentence can be either identified as containing a claim or not. Here, background knowledge is expressed by the set of labelled evidence that can potentially support a claim. Thus, we aim to investigate whether knowing the supporting evidence improves claim detection.

**Data** We consider a portion of the IBM2015 argumentative corpus, built in the context of the Debater project [217, 241]. The dataset consists of a collection of Wikipedia pages, grouped into topics. The annotation procedure carried out by IBM is context-dependent, which means that claims and evidence are annotated with the assumption that the topic (context) is given. In our scenario, we selected the four topics with the largest amount of claims and evidence.

[6]We also investigated the use of pre-trained word embeddings like GloVe [? ], but we did not observe a performance improvement.

[7]We initially trained for 3-4 epochs as a standard practice, but we immediately saw that more epochs were required to achieve significant performance.

Table 4.10: Topics extracted from the IBM2015 corpus, with associated number of evidence and claims. For claims, we also report the percentage with respect to the overall corpus size, which corresponds to the frequency of the positive class.

| Topics | No. Evidence | No. Claim | Claim Ratio |
|---|---|---|---|
| 1 | 130 | 113 | 4.2% |
| 2 | 239 | 201 | 4.0% |
| 3 | 578 | 288 | 3.8% |
| 4 | 642 | 374 | 3.5% |

Subsequently, we defined the set of evidence as the model memory. Due to how argumentative texts were annotated, there are cases in which a single sentence may contain both a claim and evidence or, more seldom, an evidence spans through multiple sentences and incorporates a claim. Indeed, these cases hinder the quality of selected data, but due to the low amount of such spurious sentences, we do not consider further pre-processing steps and use the corpus as is. Table 4.10 reports details about the selected subsets of topics, focusing on their argumentative content. We remark that evidence in the IBM2015 dataset are typically statements extracted from studies or facts established by experts: it is thus a reasonable assumption to have a list of such items available to support claim detection.

**Setting** We frame claim detection as a sentence-level binary classification task, where each sentence can either be identified as containing a claim or not. To evaluate the benefits of added knowledge, we consider incremental subsets of topics, spanning from 1 up to 4. However, differently from the legal domain case study, the memory content dramatically increases as the number of topics gets larger, from 130 with a single topic up to 642 with four topics. Additionally, the number of claims associated with each evidence is very low (range 1-5). Thus, a correct usage of memory and of strong supervision is extremely challenging. Moreover, the increased memory size prohibits the use of all the given knowledge with transformer-based architectures, due to hardware limitations. Hence, the need to adopt a sampling strategy. Given the low amount of samples, models are evaluated via a *k*-fold cross-validation procedure (we chose *k*=4). All hyper-parameters and training configurations are identical to those described for the ToS-30 setting, except for the MANN baseline, which now uses 50-dimensional embeddings.

## 4.5.3 Memory Analysis

Evaluation of memory usage cannot be reduced to task-specific metrics only, like classification's F1. Of course one may hope for increased performance; however, the addition of knowledge should be evaluated by also taking into account other possible benefits, for example in terms of interpretability. Directly measuring interpretability would require human studies, which we do not have. However, if we assume that memories contain human-interpretable, natural language justifications, a reasonable proxy could be given by suitable memory usage metrics.

As an initial step, we consider several memory-related metrics already introduced in [223]. These metrics evaluate memory usage under several perspectives, covering typical information retrieval analysis concerning top-K memory ranking, as well as purely memory-oriented statistics like target coverage. Such metrics will also be employed to analyze the behavior of sampling in domains where the full memory size is prohibitively large. We only select a limited amount of these metrics by picking the most informative ones:

- **Memory Usage (U)**: percentage of examples for which memory is used.
- **Coverage (C)**: percentage of positive examples for which (at least one) memory slot selection is correct.
- **Coverage Precision (CP)**: percentage of examples using memory, for which (at least one) memory slot selection is correct.
- **Precision at K (P@K)**: percentage of positive examples for which at least one correct target memory is retrieved in the first K positions.
- **Mean Reciprocal Rank (MRR)**: average of the reciprocal memory ranks, by considering the highest-ranking correctly retrieved target memory.

One problem is how to define an appropriate activation threshold $\delta$, i.e., the minimum value to consider a memory slot has been used by the model. While in some cases, like ToS, a $\delta = 0.5$ threshold could be meaningful enough, in other cases, like for the IBM2015 corpus, we preferred to consider in our analysis different vales for $\delta$.

## 4.5.4 Results

In this section we discuss results based on standard classification metrics, memory usage metrics, and an error analysis.

#### Unfairness Detection

Table 4.11summarizes the results obtained on the ToS-30 corpus, by reporting the macro-F1 averaged on 10-fold cross-validation. A first, clear result is that all BERT models maintain a significant performance gap over the other neural baselines.

A first, clear result is that all BERT models maintain a large performance gap over the other neural baselines. The introduced legal rationales significantly aid memory-augmented models in categories for which there are few positive examples like A and CR. On the other hand, on category CH we can observe that memory-based approaches do not show an improvement over their counterparts: MANNs perform worse than LSTMs, and MemDistilBERT performs worse than DistilBERT. This may be due to the nature of the natural language rationales given for the CH category, which are probably less informative than those given for the other categories[8]. The introduction of SS regularization always leads to better performance for MemDistilBERT models with full knowledge. SS regularization is especially beneficial in categories with larger memory size like CR, LTD and TER. Sampling-based models manage to reach performance comparable with their full knowledge counterparts. However, memory sampling attenuates the added contribution of SS regularization, leading to mixed results for different strategies.

Table 4.12 reports memory-related metrics on unfair clause classification. The results clearly show that SS brings a crucial contribution in identifying the correct explanations: a correct memory slot is ranked in the top-3 positions in over 55-80% of the cases in all categories. This behaviour is reflected also in the MRR score which shows a clear advantage of SS over WS.

### Claim Detection

Table 4.13 reports our experimental results. Like we did with unfair clause detection, we evaluate knowledge-agnostic neural baselines as well as MANN models. First of all, we note how MANNs achieved significant improvements over CNNs and LSTMs, suggesting that the introduced knowledge is indeed beneficial to the task itself. Similarly, MemDistilBERT achieves higher F1-score with respect to standard DistilBERT, whereas MemBERT is comparable with BERT. Interestingly, even the uniform sampling strategy manages to outperform other neural baselines, suggesting that sampling also compensates the noise introduced by memories, thus increasing model robustness. This behaviour is even more evident with MANNs, where the combination of SS and priority sampling largely outperforms the version with full knowledge. Interestingly, for 1 and 2 topics MANNs nearly match DistilBERT and BERT, in spite of a drastically lower number of parameters. These results support our intuition that controlled and smart knowledge sampling can be particularly beneficial when limited training data is available.

We observed that with MemDistilBERT the priority sampling strategies produce quasi uniform distributions. We think this should be ascribed to the sparsity of relations between

[8]Legal rationales for CH category are very similar to each others, with minor distinctions that might be hard to discriminate without supervised guidance.

Table 4.11: Classification macro-F1 computed on 10-fold cross-validation for unfair examples on the ToS-30 dataset. Sampling employs a 5-slot reduced size memory. We compare uniform (U), priority attention only (P-Att) and priority loss gain (P-LG) strategies. Priority sampling always considers negative example filtering (F) for correct priority update. Best results are in bold, second-best results are underlined.

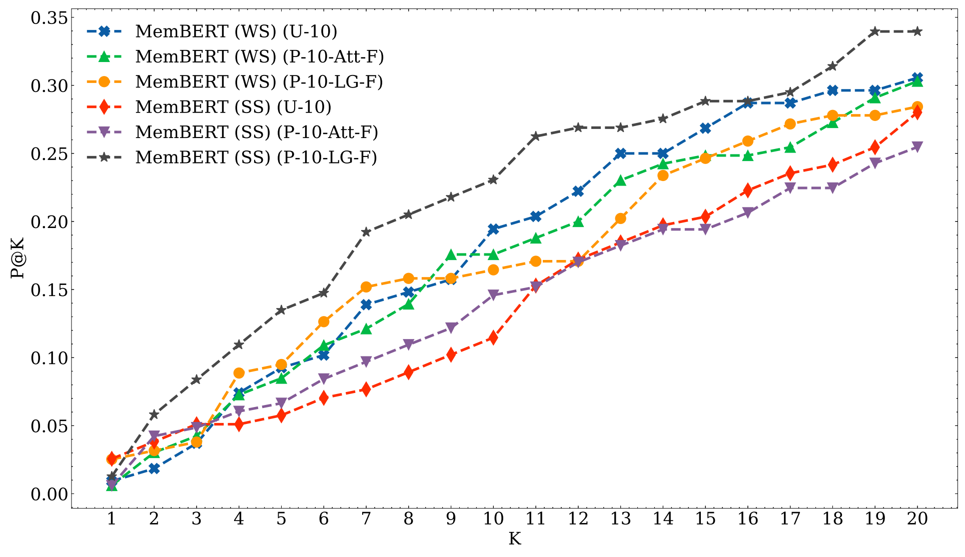
| | A | CH | CR | LTD | TER |
|---|---|---|---|---|---|
| **No Knowledge** | | | | | |
| CNN | 0.339 | 0.506 | 0.403 | 0.628 | 0.583 |
| LSTM | 0.302 | 0.573 | 0.363 | 0.602 | 0.508 |
| DistilBERT | 0.447 | **0.635** | 0.620 | 0.670 | <u>0.748</u> |
| BERT | 0.442 | 0.547 | 0.653 | 0.653 | 0.724 |
| **Full Knowledge** | | | | | |
| MANN (WS) | 0.483 | 0.506 | 0.387 | 0.635 | 0.602 |
| MANN (SS) | 0.465 | 0.516 | 0.414 | 0.605 | 0.660 |
| MemDistilBERT (WS) | 0.494 | 0.565 | 0.639 | 0.664 | 0.705 |
| MemDistilBERT (SS) | 0.504 | <u>0.609</u> | **0.670** | **0.686** | 0.737 |
| MemBERT (WS) | 0.477 | 0.581 | 0.617 | 0.650 | 0.723 |
| MemBERT (SS) | **0.524** | 0.541 | 0.616 | 0.630 | 0.740 |
| **Sampling** | | | | | |
| MemDistilBERT (WS) (U-5) | <u>0.514</u> | 0.556 | 0.609 | <u>0.678</u> | 0.702 |
| MemDistilBERT (WS) (P-5-Att-F) | 0.491 | 0.559 | 0.601 | 0.643 | 0.703 |
| MemDistilBERT (WS) (P-5-LG-F) | 0.475 | 0.574 | <u>0.660</u> | <u>0.678</u> | 0.716 |
| MemDistilBERT (SS) (U-5) | 0.503 | 0.580 | 0.617 | 0.652 | 0.702 |
| MemDistilBERT (SS) (P-5-Att-F) | 0.448 | 0.599 | 0.635 | 0.661 | 0.708 |
| MemDistilBERT (SS) (P-5-LG-F) | 0.490 | 0.536 | 0.625 | 0.656 | 0.706 |
| MemBERT (WS) (U-5) | 0.514 | 0.590 | 0.563 | 0.677 | 0.723 |
| MemBERT (WS) (P-5-Att-F) | 0.477 | 0.597 | 0.655 | 0.659 | 0.710 |
| MemBERT (WS) (P-5-LG-F) | 0.501 | 0.574 | 0.625 | 0.636 | **0.753** |
| MemBERT (SS) (U-5) | 0.463 | 0.593 | 0.599 | <u>0.678</u> | 0.719 |
| MemBERT (SS) (P-5-Att-F) | 0.459 | 0.601 | 0.645 | 0.649 | 0.747 |
| MemBERT (SS) (P-5-LG-F) | 0.480 | 0.580 | 0.637 | 0.666 | 0.723 |

memories and examples in this dataset. We thus decided to only run experiments with uniform distribution for BERT.

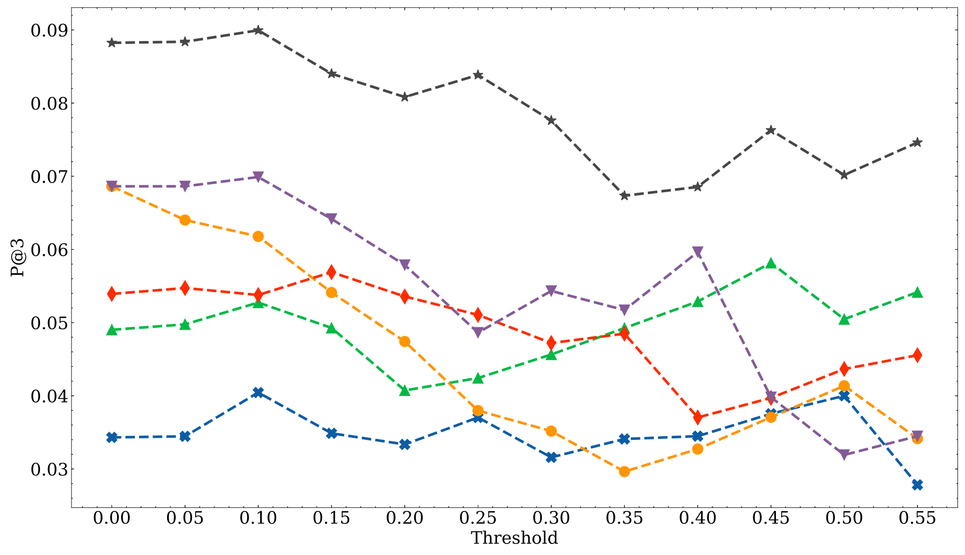
Differently from unfairness detection, the large memory size hinders selective memory lookup operations. Thus, metrics are computed over incremental values of activation threshold

Table 4.12: Memory-related metrics concerning predictions on ToS-30 unfair examples only. Activation threshold $\delta$ is set to 0.5. We shall remark that columns C to P@3 are not directly comparable among models due to their different memory usage. Comparison should be based on the MRR column only. We highlight in bold the MRR results of models with SS regularization.

| **Model** | U | C | CP | P@1 | P@3 | **MRR** |
|---|---|---|---|---|---|---|
| Arbitration (A) | | | | | | |
| MANN (WS) | 0.311 | 0.289 | 0.929 | 0.571 | 1.000 | 0.761 |
| MANN (SS) | 0.689 | 0.644 | 0.935 | 0.903 | 0.968 | **0.861** |
| MemDistilBERT (WS) | 0.489 | 0.400 | 0.818 | 0.273 | 0.545 | 0.478 |
| MemDistilBERT (SS) | 0.956 | 0.911 | 0.953 | 0.767 | 0.837 | **0.864** |
| MemBERT (WS) | 0.756 | 0.667 | 0.882 | 0.147 | 0.588 | 0.420 |
| MemBERT (SS) | 1.000 | 0.933 | 0.933 | 0.778 | 0.844 | **0.862** |
| Unilateral Change (CH) | | | | | | |
| MANN (WS) | 0.169 | 0.090 | 0.533 | 0.000 | 0.067 | 0.299 |
| MANN (SS) | 0.854 | 0.730 | 0.855 | 0.855 | 0.961 | **0.883** |
| MemDistilBERT (WS) | 0.404 | 0.382 | 0.944 | 0.250 | 0.750 | 0.522 |
| MemDistilBERT (SS) | 1.000 | 0.955 | 0.955 | 0.809 | 0.888 | **0.886** |
| MemBERT (WS) | 0.801 | 0.719 | 0.889 | 0.222 | 0.597 | 0.505 |
| MemBERT (SS) | 1.000 | 0.865 | 0.865 | 0.809 | 0.921 | **0.889** |
| Content Removal (CR) | | | | | | |
| MANN (WS) | 0.017 | 0.000 | 0.000 | 0.000 | 0.000 | 0.335 |
| MANN (SS) | 0.672 | 0.414 | 0.615 | 0.282 | 0.872 | **0.612** |
| MemDistilBERT (WS) | 0.328 | 0.328 | 1.000 | 0.316 | 0.632 | 0.478 |
| MemDistilBERT (SS) | 1.000 | 0.948 | 0.948 | 0.431 | 0.879 | **0.681** |
| MemBERT (WS) | 0.430 | 0.362 | 0.84 | 0.2 | 0.32 | 0.328 |
| MemBERT (SS) | 1.000 | 0.914 | 0.914 | 0.466 | 0.879 | **0.686** |
| Limitation of Liability (LTD) | | | | | | |
| MANN (WS) | 0.037 | 0.025 | 0.667 | 0.33 | 0.833 | 0.504 |
| MANN (SS) | 0.814 | 0.534 | 0.656 | 0.313 | 0.573 | **0.501** |
| MemDistilBERT (WS) | 0.497 | 0.416 | 0.838 | 0.100 | 0.275 | 0.328 |
| MemDistilBERT (SS) | 1.000 | 0.919 | 0.919 | 0.224 | 0.565 | **0.474** |
| MemBERT (WS) | 0.478 | 0.366 | 0.766 | 0.156 | 0.351 | 0.303 |
| MemBERT (SS) | 1.000 | 0.919 | 0.919 | 0.304 | 0.609 | **0.500** |
| Unilateral Termination (TER) | | | | | | |
| MANN (WS) | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.499 |
| MANN (SS) | 1.000 | 0.471 | 0.471 | 0.438 | 0.537 | **0.536** |
| MemDistilBERT (WS) | 0.223 | 0.198 | 0.889 | 0.074 | 0.074 | 0.193 |
| MemDistilBERT (SS) | 1.000 | 0.851 | 0.851 | 0.438 | 0.579 | **0.567** |
| MemBERT (WS) | 0.215 | 0.198 | 0.923 | 0.423 | 0.538 | 0.275 |
| MemBERT (SS) | 1.000 | 0.876 | 0.876 | 0.380 | 0.636 | **0.568** |



(a)



(b)

Figure 4.6: Memory analysis on IBM2015, 1-Topic. Left: P@K for increasing $K$ values and $\delta = 0.25$. Right: P@3 for increasing $\delta$ values. Metrics for sampling-based models are averaged across three distinct inferences on the test set.

$\delta$. In particular, we mainly focus on the P@K metric, since it is a valuable indicator of the model retrieval capabilities. Figures 4.6a reports values of P@K on the 1-Topic case, along incremental value of $K$, with fixed $\delta = 0.25$, whereas Figure 4.6b reports P@3 with incremental $\delta$ values. In both cases, we observe a slight advantage in using loss gain priority sampling coupled with SS (P-10-LG-F).

Table 4.13: Classification performance for dataset IBM2015. For each topics subset, we report the achieved macro F1-score. WS and SS stay for weak and strong supervision, respectively. Memory sampling is tested with a 10-slot reduced size memory. We report the uniform strategy (U), the priority attention only strategy (P-Att) and the priority loss gain strategy (P-LG). Priority sampling always considers negative example filtering (F) for correct priority update. Best results are in bold, second best results are underlined instead.

| | 1 Topic | 2 Topics | 3 Topics | 4 Topics |
|---|---|---|---|---|
| **No Knowledge** | | | | |
| CNN | 0.196 | 0.283 | 0.287 | 0.268 |
| LSTM | 0.194 | 0.344 | 0.278 | 0.272 |
| DistilBERT | 0.317 | 0.431 | 0.405 | <u>0.451</u> |
| BERT | 0.246 | 0.442 | 0.431 | 0.450 |
| **Full Knowledge** | | | | |
| MANN (WS) | 0.252 | 0.380 | 0.325 | 0.336 |
| MANN (SS) | 0.205 | 0.392 | 0.317 | 0.281 |
| **Sampling** | | | | |
| MANN (WS) (U-10) | 0.269 | 0.406 | 0.331 | 0.355 |
| MANN (WS) (P-10-Att-F) | 0.251 | 0.402 | 0.322 | 0.358 |
| MANN (WS) (P-10-LG-F) | 0.259 | 0.408 | 0.332 | 0.340 |
| MANN (SS) (U-10) | 0.297 | 0.400 | 0.319 | 0.352 |
| MANN (SS) (P-10-Att-F) | 0.264 | 0.423 | 0.332 | 0.348 |
| MANN (SS) (P-10-LG-F) | 0.302 | 0.424 | 0.344 | 0.354 |
| MemDistilBERT (WS) (U-10) | 0.311 | <u>0.457</u> | **0.454** | **0.453** |
| MemDistilBERT (WS) (P-10-Att-F) | 0.275 | 0.449 | 0.422 | 0.434 |
| MemDistilBERT (WS) (P-10-LG-F) | 0.305 | 0.449 | 0.428 | 0.428 |
| MemDistilBERT (SS) (U-10) | <u>0.341</u> | 0.442 | <u>0.444</u> | 0.436 |
| MemDistilBERT (SS) (P-10-Att-F) | **0.354** | 0.424 | 0.421 | 0.423 |
| MemDistilBERT (SS) (P-10-LG-F) | 0.290 | **0.459** | 0.411 | 0.444 |
| MemBERT (WS) (U-10) | 0.278 | 0.430 | 0.427 | 0.428 |
| MemBERT (SS) (U-10) | 0.253 | 0.421 | 0.436 | 0.432 |

### 4.5.5 Error Analysis and Discussion

As observed, in unfairness detection, SS regularization is highly effective in selecting the appropriate legal rationales when detecting an unfair clause (see Table 4.12). This is especially true with categories with a larger memory like CR, LTD and TER. Compared to its weak supervision (WS) counterpart, the SS regularization helps in filtering out irrelevant legal rationales. For instance, consider the following clause, taken from the Linden Lab contract, that is unfair according to the limitation of liability (LTD) category:

*In such event, you will not be entitled to compensation for such suspension or termination, and you acknowledge that Linden Lab will have no liability to you in connection with such suspension or termination.*

BERT fails in identifying the clause as being unfair, whereas memBERT (WS) and memBERT (SS) correctly detect the unfairness. Nonetheless, memBERT (WS) almost uses half of the memory slots (17 out of 28). In contrast, memBERT (SS) selects only four of them and shows a preference for the target memory slot, which corresponds to the following rationale:

*Since the clause states that the provider is not liable for any technical problems, failure, suspension, disruption, modification, discontinuance, unavailability of service, any unilateral change, unilateral termination, unilateral limitation including limits on certain features and services or restriction to access to parts or all of the Service without notice.*

We notice that the impact of SS regularization is more significant when the memory is limited in size and each memory slot is associated with several input samples. This is not the case when addressing claim detection where hundreds of memory slots are considered. In contrast, we observe performance improvement when the amount of training data is limited. We also notice a similar behaviour in unfairness detection in categories with fewer unfair clauses like A and CR.

The introduction of smart sampling strategies allows to scale to larger memories. However, the introduced priority-based strategies lack a proper conditioning on the input, but rather learn a dataset-level importance of each memory slot. This works well in unfairness detection, where there are few memory slots and they are associated with many samples; not so well in claim detection, where there are many memory slots, each associated with one or very few samples. In our experiments, this led priority-based strategies to learn a quasi-uniform priority distribution. This suggests as a possible direction for improvement the adoption of input-conditioned sampling strategies compatible with SS regularization. For instance, a parametric model could be trained to embed input texts near their target memory slots into a latent space [190].

We explored unstructured knowledge integration with BERT models. In particular, we proposed a memory-augmented version of DistilBERT that follows the traditional MANN architecture. We considered two challenging experimental settings: (i) class descriptions for unfair clause detection in the legal domain and (ii) evidence as support for claim detection in the context of argumentation mining. Despite a relatively straightforward formulation, promising results have been achieved concerning classification performance and interpretability. Most notably, when strong supervision is applied, memory-augmented models learn to select target memory slots with high precision correctly and, in most cases, reach higher performance. Memory sampling allows to scale up to large knowledge bases. In particular, priority-based sampling strategies reach more consistent performance from the viewpoint of both classification and interpretability.

Nonetheless, correctly handling text examples not covered by the integrated knowledge is still an open problem. To this end, more sophisticated memory interaction mechanisms have to be investigated to improve the performance of the approach. Generally speaking, devising adequate unstructured knowledge mapping mechanisms is particularly challenging. This is also evident when considering the contextual data KT for argument sentence detection. In particular, contextual data is relatively more straightforward to integrate than legal rationales since it doesn't involve specific integration requirements. However, correctly mapping contextual data in domains like Argument Mining (AM) is not immediate. Unwanted input to knowledge mappings can occur and have to be addressed in multiple stages of UKI. For instance, unstructured knowledge might be reformulated as a solution or explore more sophisticated integration mechanisms.

We briefly outline some research directions concerning unstructured knowledge integration from this perspective. In particular, the introduced memory sampling strategies present two major drawbacks: (i) the priority update process is disruptive as it considers each sample batch's contribution. Thus, there may be cases in which a specific memory slot usage oscillates between frequent use and no usage at all. Such oscillations are then reflected when updating the memory slot priority value; (ii) the sampling process is not directly affected by the given input samples. That is, priority values consider the cumulative behaviour of memory usage and are not specific to each input sample. More precisely, the memory for a specific batch is sampled according to the current priority values and not on the given inputs. Such behaviour is particularly evident at inference time, where priority values are fixed, and no explicit conditioning on the given input is applied to regulate the sampling process.

### Cumulative Memory Priority Update

We can attenuate the disruptive behaviour regarding priority updates by introducing cumulative updates. Intuitively, these updates are accumulated across batches rather than updating priority values at each batch. Priority updates are then applied periodically every $B$ batches. By doing so, batch-wise priority values jumps are averaged together into a single update. By controlling the $B$ hyper-parameter, it is possible to shift towards more stable priority updates.

### Input Conditioned Sampling

To condition the sampling process on the given input sample, we propose to replace the sampling strategies with a parametric model $f_\phi$. This model is responsible for outputting the memory priority values for sampling based on the given input sample. In this way, it is possible to associate input-specific memory priority values via $f_\phi$. When considering multiple inputs simultaneously, e.g. at batch-level during training time, we can consider averaging priority values across inputs to determine a unified set of memory priority values. Subsequently, memory slots are sampled according to obtained priority values. The added parametric model $f_\phi$ can be trained in a meta-learning fashion [190], where the training phase alternates between classification model updates and $f_\phi$ parameter updates. Furthermore, $f_\phi$ can be trained to imitate a particular sampling strategy.

## 4.6 Considerations on Unstructured Knowledge Integration

In this chapter, we have introduced two primary preliminary studies about unstructured knowledge integration in NLP. In the first, we have presented the problem of unfair clause detection in legal documents. Here, knowledge is defined as legal explanations regarding the unfair class in a binary classification task. Legal experts have acquired these legal explanations as motivation concerning the decision process behind the labelling of a clause as unfair. Such motivations are crucial in the legal domain for adopting automatic detection models as their decision process must be accompanied by adequate and interpretable motivations.

In the second study, we considered the task of claim detection and formulated it as a binary classification task. In this scenario, the knowledge is comprised of the set of examples labelled as evidence. We then explore the task of argument detection by employing the set of evidence examples as an indicator of the presence of a claim. Indeed, claim, and evidence are strongly correlated classes as an evidence often supports a claim [268]. Despite the challenges related to UKI, we have shown that a straightforward approach via MANNs, based on semantic similarity, can achieve satisfying results on both presented case studies.

First, in the context of unfair clause detection, the MANN-based classification model is enriched to output the employed legal rationales supporting its reasoning process. Thus, the legal rationales act as a support device to ground the model classification predictions on interpretable information. In other words, we can evaluate the classification performance by also looking at the employed set of legal rationales. Therefore, the model is judged not only on its capability of providing the correct answer, but we also introduce an additional reference, i.e. the knowledge to which we can relate model predictions. Second, regarding claim detection, we have experimentally verified how correlated class information, such as the set of evidence examples, can act as a valuable task-specific indicator. This property is particularly evident when considering low-resource scenarios, i.e. when available training data is limited. In this case, introducing examples belonging to the same data distribution that highly correlate with specific input examples are sufficient to boost model performance significantly. Third, we have shown that MANNs are a simple and straightforward model architecture for UKI that is compliant with multiple knowledge types (KTs). More precisely, legal rationales and evidence examples have been integrated as the memory component of the MANN-based model without any architecture modification.

Generally speaking, UKI is a challenging task. The described preliminary studies only explore a few KTs and are subject to strong assumptions and limitations. First of all, knowledge interpretability is only superficially explored. In both our case studies, we consider soft attention weights as the major indicator for explaining the decision process of the classification models. The unstructured knowledge mapping mechanism is cast down to a basic semantic similarity operation in the embedding space. We explored UKI with deep learning models by leveraging the MANN architecture. In particular, MANNs are naturally compliant with knowledge integration since we can use the memory component to store the knowledge. Nonetheless, other neural architectures could be used either in place or in combination with MANNs for UKI.

Although recent work supports the analysis of attention weights as an explanation tool [77], it is necessary to develop methods that impose a robust knowledge verification process that is interpretable by experts. For instance, causal reasoning [286] and probabilistic logic programming (PLP) [54, 166] could be employed to entail the association between the input text and the selected set of legal rationales. In this context, structured knowledge extraction (KE) could be considered in combination with the research mentioned above areas to satisfy critical requirements of UKI like *coherence* and *correctness*. In Chapter 5, we propose Neural-Symbolic Learning and Reasoning (NeSy) [17, 293] as a valuable methodology to integrate unstructured knowledge while guaranteeing some of the derived, hard requirements.

# Chapter 5

# Open Challenges and Future Research Directions

In Chapter 3, we described a few experimental scenarios regarding structured knowledge extraction (KE). In particular, we have concluded the chapter by underlining the importance of appropriate KE representations. Such representations are abstract formulations of a corresponding text robust to specific natural language issues (Section 2.7). Additionally, structured knowledge also defines a semantic frame that allows reasoning on the abstract representation. Notable examples are knowledge graphs and soft logic rules. In Chapter 4 we have scratched the surface of UKI by presenting some preliminary studies regarding text classification. Our final considerations show that even simple approaches that do not guarantee crucial unstructured knowledge integration properties can provide notable benefits to a classification model.

Regarding UKI, we have described KE as a sub-process that can be adopted to satisfy specific hard requirements like *coherence* and model transparency. In this chapter, we further bridge the connection between KE and UKI. We initially provide a few examples of how the methods introduced in Chapter 3 could address the tasks presented in Chapter 4 within the umbrella of UKI. In particular, we describe the benefits of adopting structured knowledge representations to define UKI sub-processes like unstructured knowledge mapping. We remark using KE sub-processes that are compliant with the given knowledge type (KT) in terms of knowledge integration. For instance, the extracted structured knowledge should not violate the unstructured knowledge integration mechanism of the KT.

We provide a unified vision of all possible KE representations by viewing them and their corresponding semantic frame as a general symbolic method for knowledge integration. Under this perspective, we consider the framework of Neural-Symbolic Learning and Reasoning (NeSy) [17, 293] as a valuable tool for encoding KE as part of a UKI process to guarantee specific demanding requirements of UKI. We conclude by providing concrete examples where KE formulated as NeSy is encoded as a UKI sub-process.

## 5.1 KE as a Sub-process of UKI

We resume the experimental scenarios presented in Chapter 3 regarding structured knowledge extraction. In particular, we provide a few examples of their application to define more sophisticated UKI sub-processes. In the first example, we discuss the adoption of TC-GNNs to define a more complex unstructured knowledge mapping mechanism that relies on the extracted sub-tree features. In particular, both the semantic and syntactic dimensions of input clauses and legal rationales are considered for knowledge mapping.

In the second example, we consider the argumentative graph that connects evidence and claims for claim detection. In this scenario, we identify two potential benefits regarding using the argumentative graph. First, the argumentative graph can limit the search space for unstructured knowledge mapping. In particular, the graph topology is exploited to regulate knowledge sampling. Second, the argumentative graph could also condition the decision-making process of a model. More precisely, model predictions should not violate the argumentative relations defined in the graph.

#### Sub-Tree Features for Unfair Clause Detection

We propose TC-GNNs to automatically extract structured features from the text for unfair clause detection. TC-GNNs are used to define a more sophisticated unstructured knowledge mapping mechanism of legal rationales. Sub-tree features are automatically extracted from legal rationales and input clauses to classify. Consequently, the mapping mechanism is built on top of the aforementioned structured knowledge.

The extracted sub-trees define a more sophisticated unstructured knowledge mapping mechanism by evaluating the similarity of each sub-tree pair in the embedding space. Formally, we consider the input class to classify as $c$, and the provided set of class rationales as knowledge $\{r_1, r_2, \dots, r_M\}$, where $M$ is the size of the knowledge. Additionally, we consider a TC-GNN model $g_\phi$ with $\phi$ as the set of learnable parameters. For each input $c_j$ to classify, the unstructured knowledge mapping mechanism is defined as:

$$g_\phi(c_j) = \{st_1^{c_j}, \dots, st_K^{c_j}\} \tag{5.1}$$

$$g_\phi(r_i) = \{st_1^{r_i}, \dots, st_K^{r_i}\} \quad \forall i = 1, \dots, M \tag{5.2}$$

$$s(c_j, r_i) = h\left(g_\phi(c_j), g_\phi(r_i)\right) \quad \forall i = 1, \dots, M \tag{5.3}$$

where $st_k^{c_j}$ is the sub-tree node cluster $k$ for clause $c_j$, $K$ is the number of sub-tree clusters that $g_\phi$ extracts, $s(a,b)$ is the similarity score between two input texts $a$ and $b$, $h(a,b)$ is a matching function [149] that computes the similarity between each pair of sub-trees belonging to $a$ and $b$.

The described approach defines a more informative unstructured knowledge mapping mechanism based on structured knowledge. Note that the mapping is still applied in the embedding space. Alternatively, we can greedily extract hard sub-trees via a rounding threshold at the graph pooling layer. Subsequently, we can consider more transparent criteria regarding knowledge integration that rely on the extracted structured information. For instance, we might identify specific structured patterns in given legal rationales to regulate the unstructured knowledge mapping mechanism when facing an input clause $c_j$.

### Argumentative Graph for Argument Sentence Detection

Another example of integrating KE as a sub-process of UKI regards the task of claim detection described in Section 4.5. In this scenario, evidence define correlated information relevant to detecting claims. We can follow a similar approach to the one described for argumentative scientific dialogues (Section 3.4). In particular, we intend to build an argumentative graph of all claims and evidence. By doing so, we can exploit relations between claims and evidence to ensure inference consistency and ease the process of detection.

Formally, we consider the input text to classify $x$ and the extracted argumentative graph concerning the knowledge $G_k$ for knowledge $k$. A model $f_\theta$ is trained to classify $x$ as either being a claim or a non-argumentative sentence by comparing $x$ with $G_k$ as follows:

$$f_\theta(x)_{G_k} = u\left(\{f_\theta(x), m_\phi(x, G_k)\}\right) \tag{5.4}$$

where $u$ is a function that fuses inputs according to a given criterion. A few examples are summation and concatenation. $m_\phi$ is a matching function that computes the similarity between the input $x$ and each node in $G_k$.

The described decision process is as with the one reported in Section 4.5. In particular, the knowledge $G_k$ is employed to provide additional information $m_\phi(x, G_k)$ to the model $f_\theta$ to address the task of claim detection better. In Section 4.5.2, $m_\phi(x, G_k)$ is defined via an embedding-based similarity metric where $G_k$ has no connections, i.e. all evidence are independent of each other, and evidence are retrieved via sampling.

In this scenario, we can exploit the $G_k$ in several ways. For instance, we can limit the search space of evidence by leveraging the graph structure. In particular, sampling strategies can consider the graph topology to define a hierarchical sampling approach. Additionally, sampling priorities regarding specific evidence could be propagated to connected nodes in the argumentative graph. By doing so, the graph topology can compensate for potentially missed input to knowledge mappings due to low matching scores. In a reversed perspective, we can regulate model predictions to satisfy the relations reported in the argumentative graph. For instance, if $x$ is predicted as being a claim by using evidence $e_i \in G_k$ and $e_i$ is connected to $e_2$, then $x$ should also be supported by $e_2$. Note that we require an additional relation prediction model to regulate the confidence of the classification model $f_\theta$.

## 5.2 KE as a Symbolic Representation

A wide variety of abstract text representations fall within the frame of structured knowledge. Notable examples are soft logic rules, knowledge graphs and constituency parse trees. In Chapter 3, we discussed how KE could be viewed as a sub-process of UKI. In particular, structured knowledge representations help address the challenges of satisfying UKI properties like *coherence*. Furthermore, these properties depend on the given knowledge type (KT).

In this section, we provide a straightforward interpretation of KE to unify all possible abstract representations and corresponding semantic frames under a single scope. We can view the output of a KE process as two distinct factors: (i) a translation process from unstructured knowledge to structured knowledge and (ii) a semantic frame upon which reasoning can occur. Altogether, these two factors denote a *symbolic representation* of given text input. In particular, each KE process instance follows this general behaviour. The natural language representation of an input text is translated to a higher-level text representation. The translation process is realized in compliance with a semantic specific to the given KE process instance.

For example, we can employ soft logic rules to describe a general behaviour to guide the learning of a model [111, 114]. The underlying dynamic behind the adoption of soft logic rules is translating such behaviour into a specific representation. Rules define the latter according to a given framework, e.g. differentiable mathematical functions. Such an abstract representation also comes with a way to interpret them. Simply put, we can evaluate soft logic rules as functions.

As we can use structured representations interchangeably (Section 1.2), the same behaviour can be expressed via a knowledge graph, for instance. Such different but equivalent representation also comes with a corresponding semantic frame. Consequently, we view a KE sub-process for UKI as a means of defining a symbolic representation of a given unstructured textual knowledge. The defined symbolic representation conveys both the two distinguishing factors of KE. In particular, the first factor mainly addresses the integration requirement of *robustness* as it filters out spurious text information (Section 2.4.4). Furthermore, the second factor provides a method for guaranteeing properties like *transparency* and *coherency* (Section 2.3.3).

## 5.3 Neural-Symbolic Reasoning for UKI

We have declared KE as a sub-process of UKI that defines a symbolic representation of unstructured knowledge. Such symbolic representation is the backbone of an unstructured knowledge integration process to guarantee properties specific to the given KT. The integration of symbolic representations into deep learning models and, more generally, into sub-symbolic approaches dates back to the initial stages of AI (Chapter 1). Recently, an emerging hot topic in deep learning is represented by Neural-Symbolic Learning and Reasoning (NeSy) [17, 293] approaches that aim to combine symbolic representations with neural networks. We hereby identify NeSy as a general framework to address the integration of a KE sub-process into UKI. First, we briefly overview the motivations behind the introduction of NeSy. Then, we discuss the benefits how applying NeSy as part of a UKI process.

### 5.3.1 Neural-Symbolic Learning: A Brief Overview

A major research trend in Artificial Intelligence (AI) with theoretical foundations in sciences like psychology, philosophy and cognitive science is represented by Neural-Symbolic Learning and Reasoning (NeSy) [17, 211, 293]. Purely logic and symbolic based frameworks incorporate the notion of high-level reasoning such as classical, temporal and deductive logic [17]. The major advantage of such systems is their transparency and generalization capabilities relying on the employed symbols and their representation [82].

Nonetheless, symbolic systems are brittle to data fluctuations, i.e. noise, due to their poor flexibility at processing unstructured data. Conversely, purely data-driven approaches like neural networks complement symbolic systems as they are highly efficient at learning from a large amount of data. Their major downside is the lack of exposing a transparent decision-making process, and fragile generalization capabilities generally characterize them [293]. NeSy systems aim to combine inductive learning and deductive reasoning altogether to conciliate the two methodologies to acquire the advantages of both and address their respective shortcomings. As reported in [17], the objective of NeSy can also be formulated as *"providing a coherent, unifying view for logic and connectionism, to contribute to the modelling and understanding of cognition and, thereby, behaviour, and to produce better computational tools for integrated machine learning and reasoning"*.

### Combining Learning and Reasoning

The integration of the two worlds of NeSy (i.e. the symbolic and the sub-symbolic ones) assumes different forms. Notable examples are the integration of logic as part of the architecture of a neural network [70, 207, 208, 220, 262], using empirical evaluation criteria [121], ad-hoc neural architectures that expose reasoning capabilities [92, 94, 276], and symbolic-oriented regularizations [61, 111, 283].

A recent survey regarding NeSy applications [293] divides existing literature on how reasoning and learning are combined. In particular, the following taxonomy is introduced: (i) *heavy-reasoning and light-learning* and (ii) *heavy-learning and light-reasoning*. First, neural networks are the bridging step between unstructured input data and symbolic representation. On top of that, symbolic reasoning is applied [27, 68, 70, 119, 166]. Differently, in *heavy-learning and light-reasoning*, symbolic methods guide the training phase of a neural network and can be generally viewed as part of the input, denoted as the *knowledge*, that conditions its decision-making process [63, 283].

### Reasoning Frameworks

According to [293], the reasoning is generally defined via formal languages and knowledge graphs. In the first, first-order logic [69] and propositional logic are the mainly adopted representations. Depending on the given representation, logic reasoning is integrated into numerical representations via adequate frameworks. Notable examples are t-norm fuzzy logic [170], Logic Tensor Networks (LTN) [242] and Markov Logic Networks [216].

### Limitations and Challenges

Strong assumptions and limitations characterize current contributions that fall within the scope of NeSy. Indeed, the integration of symbolic and sub-symbolic paradigms into a unified perspective that can benefit both worlds is still in its infancy [17]. The set of limitations spans three main dimensions [293]: (i) computational inference capabilities; (ii) dynamic knowledge definition; and (iii) symbolic representation.

First, we can hardly employ NeSy solutions in real-case scenarios due to the combinatorial complexity of symbolic reasoning. For instance, Markov Logic Networks [216] have exponential complexity regarding the number of possible logic rules instantiations. To this end, solutions with approximate inference methods have to be considered [8, 125]. Second, in many cases, the symbolic knowledge is mainly hand-crafted by domain experts and considered static throughout the integration process. Few examples concerning Inductive Logic Programming (ILP) explore the automatic constructions of new rules but are affected by strong limitations [70, 169, 171, 288]. Third, NeSy systems hardly adapt to variations of the symbolic representation and are tied to its informative content. In particular, the mapping between unstructured data and its corresponding symbolic representation stands a research direction worth exploring.

Additionally, other theoretical limitations are pointed out in [17]. Amongst the many, modelling human behaviour is of particular interest as it represents a significant challenge in a logic-oriented representation. In particular, human reasoning is defined as non-deterministic and subjective, thus, not always following an optimal reasoning path that is logically sound. Factors attributable to the emotional sphere of human behaviour, like personal bias, have to be modelled. We can integrate probability into the symbolic paradigm to account for such uncertainties [108]. However, systems built on such consideration are still subdued to acquire large amounts of data to address possible uncertainties with statistical relevance.

### 5.3.2 NeSy for UKI

We hereby discuss how NeSy fits within the scope of UKI. According to the founding motivation behind the introduction of NeSy, UKI in NLP is another branch of sub-symbolic tasks characterized by two distinct inputs: (i) an unstructured task-specific set of data and (ii) another relevant set of textual data that in UKI is denoted as the knowledge. In the optic of UKI, we identify two distinct roles of NeSy that reflect the taxonomy described in Section 5.3.1. The first role involves translating unstructured textual knowledge into a symbolic representation to address unstructured knowledge validation. Secondly, the defined symbolic representation can regulate the data-driven learning approach of a model while considering the knowledge to be integrated.

The first role mainly reflects the idea behind *heavy-reasoning light-learning* when considering properties like *coherence* and *correctness*. For example, text constraints require a symbolic representation to evaluate their degree of satisfiability quantitatively. Additionally, the reasoning process must be transparent to qualitatively assess the proper enforcement of those constraints within the model.

On the other hand, we can apply solutions within the *light-reasoning and heavy-learning* category to UKI for the unstructured knowledge integration process. We may also define tight dependencies between symbolic representation and evaluation criteria regarding the task and the intended symbolic integration of the knowledge. Depending on the given KT, the above distinction may collapse into a single process when the unstructured knowledge integration process is also the primary evaluation criterion for guiding the learning of a model. For instance, forestalling one of the presented future research directions, the task of text classification with just class descriptors views unstructured knowledge integration as the learning signal to instruct the model on the task (Section 5.4.1). To better outline the above discussion concerning NeSy for UKI, we resume the two experimental settings introduced in Section 5.1. In particular, we briefly describe how we could adopt a NeSy framework to improve the sub-processes of UKI.

The application of TC-GNNs for unfair clause detection can be re-arranged into a NeSy formulation. We could denote the dependencies regarding unstructured knowledge integration concerning clause detection via probabilistic logic programming (PLP) and strict rules. For instance, we can define a simple unstructured knowledge mapping mechanism in DeepProbLog [166] as reported in Figure 5.1. In particular, we define consider a classification model that outputs the probability $p_u$ of a clause $c_j$ being potentially unfair. Furthermore, we consider a TC-GNN model that computes the similarity score $p_{r_i}$ between $c_j$ and each legal rationale $r_i$ (Eq. 5.3). A clause is then labelled as unfair if the classifier labels it as potentially unfair and there's a match with at least one legal rationale. Additionally, a clause can also be unfair without any knowledge match with some uncertainty to account for partial knowledge coverage and its natural language formulation.

Regarding claim detection, argument relations (between evidence and claims) can condition the classification process of a model via a NeSy formulation. In particular, the graph topology concerning argument relations should not be violated when an input text $x$ is classified as a claim and $x$ is connected to one or more evidence. For instance, we could define a dependency between claim $x$, its connected evidence $e_1$ and another evidence $e_2$ that is connected to $e_1$ as follows:

$$supports(E2,X) \leftarrow claim(X), supports(E2,E1), supports(E1,X)$$

A similar approach has been proposed for argument mining for relation prediction [78].

## 5.4 Application Examples

In this section, we provide a few examples of challenging UKI tasks that we view as future research directions. We provide a brief description of each problem and discuss how introducing a KE sub-process based on NeSy is a valuable solution to address task-specific challenges from the point of view of unstructured knowledge integration properties satisfaction.

```
% facts
p_u::pot_unfair(clause).

p_r1::match(clause,rationale1).
p_r2::match(clause,rationale2).
% ...
p_rM::match(clause,rationaleM).

% rules
unfair(C) :- pot_unfair(C), match(C,R).
0.3::unfair(C) :- pot_unfair(C).
```

Figure 5.1: An excerpt of a DeepProbLog program for defining an unstructured knowledge mapping mechanism based on TC-GNNs.

### 5.4.1 Text Classification with Class Descriptors

We consider a text classification problem where class label information is not provided. Instead, text examples are described via a set of textual class descriptors (Section 2.6.3). In other terms, class descriptors replace class label information and act as indicators for categorizing text examples into classes. We can view the problem as text clustering based on class descriptors. A similar approach based on class descriptors denoted as task descriptors has been recently proposed in the context of Zero-Shot Learning (ZSL) [275]. In this scenario, task descriptors are questions that define datasets of question-answer pairs. In particular, a task defines a specific generalization capability like paraphrasing and semantic flips. The chosen task then formulates questions.

As a concrete example, we consider the task of text classification with human annotation guidelines. Therefore, the task is viewed as imitating the process of a human annotator when dealing with the same input data. We assume that the given set of class descriptors covers a significant portion of input data. This assumption is reasonable as human guidelines are employed to annotate the same data. Furthermore, class descriptors provide explicit information regarding the distinctive properties of input data.

To showcase, we consider a single set of class descriptors related to a given class $\mathscr{A}$. In particular, the set of class descriptors contains $N$ distinct text descriptors: $\{D_1^{\mathscr{A}}, D_2^{\mathscr{A}}, \dots, D_N^{\mathscr{A}}\}$. Multiple class descriptors for the same class may be defined as, in general, multiple properties falling within the same class. Suppose that certain dependencies characterize these class descriptors. For instance, if $D_1^{\mathscr{A}}$ is considered, then $D_2^{\mathscr{A}}$ must be considered as well, thus defining a sort of metaclass descriptor (described as the union of $D_1^{\mathscr{A}}$ and $D_2^{\mathscr{A}}$). Additionally, certain class descriptors may be mutually-exclusive. In general, we can consider all kinds of dependencies. Most notably, in the context of multi-class classification, class descriptors referring to different classes are not compatible with each other. These domain-specific dynamics regarding class descriptors dependencies must be taken into account when addressing the task of text classification. In particular, model predictions should not violate the dependencies mentioned above.

We can formulate class descriptors dependencies in NeSy. For instance, Figure 5.2 shows an example formulation via Grounding Specific Markov Logic Networks (GS-MLN) [152]. In particular, we specify a hard constraint regarding $D_1^{\mathscr{A}}$ and $D_2^{\mathscr{A}}$ so that model predictions must follow the above behaviour regarding these two class descriptors. Additionally, we model uncertainty about other descriptors and define a hard constraint regarding another class descriptors set of size $M$: $D_1^{\mathscr{B}}, D_2^{\mathscr{B}}, \dots, D_M^{\mathbf{B}}$ for class $\mathscr{B}$.

We are particularly interested in this problem setup for two main reasons. First, text classification with class descriptors can be viewed as a relaxation of the standard classification formulation as explicit class properties are formulated in natural language. We can directly inspect these properties to assess the capabilities of a model and, in parallel, evaluate the employed set of class descriptors. For instance, in the above example, we can verify if the defined human guidelines are related to the input data. Second, a formulation based on class descriptors inherently defines a range of possible knowledge formulations. Considering the issues of unstructured textual knowledge like *semantic equivalence*, we can evaluate model performance when varying the formulation of text descriptors.

When adding the capability to generate or modify new knowledge, such a process reflects guidelines refinement in the above example. Lastly, the generation of new knowledge introduces further constraints that we can model via NeSy. Amongst many, newly generated class descriptors might describe distinct class properties that existing descriptors cannot subsume. The generation task with a NeSy formulation is particularly new, and few contributions have been recently proposed [183].

```
w1(x)      Entail(x,$fx,da1) ^ ClassA(da1) ^ ClassA(da2) =>
                !Entail(x,$fx,da2)

w2(x)      Entail(x,$fx,da) ^ ClassA(da) ^ ClassB(db) =>
                !Entail(x,db)
```

Figure 5.2: An excerpt of a GS-MLN program for defining class descriptors dependencies.

### 5.4.2 Substitution-based Pattern Matching

We explore more advanced methods for textual patterns integration and mapping. As an example, we consider the task of unfair clause detection. In this task, we view the set of legal rationales as comparison terms for the classification of unfair clauses. In particular, unstructured knowledge mapping is defined as a simple embedding-based semantic similarity. Thus, the problem setup is similar to textual patterns. In this example, legal rationales are often seen as listing specific abstract properties or, conversely, are far too grounded on the given domain. In the latter, legal rationales can be treated via the same unstructured knowledge integration process of textual patterns.

As a solution, we discuss the definition of a more sophisticated unstructured knowledge mapping mechanism based on term substitution. More precisely, we view each legal rationale as a complex textual pattern. Due to their complexity and word length, we replace embedding-based semantic similarity with an iterative substitution mechanism. Based on a specific key, each legal rationale is viewed as a text where key related terms are treated as variables open to grounding. Examples of keys are part-of-speech (POS) tags. The grounding operation is from logic programming languages where variables are assigned to specific instance values. The information of the input text represents the pool of candidates for grounding.

Textual pattern matching is then based on the degree of the reconstruction of the original legal rationale. Such method differs from the embedding-based semantic similarity as the former defines a semantic of knowledge mapping aligned with the interpretation and role of textual patterns. Figure 5.3 provides a schema of the described approach.

In this context, we require an adequate semantic to correct the substitution-based pattern matching method that defines the knowledge mapping mechanism. That is, term substitutions have to be logically combined to define the concept of pattern matching. Simply put, we define a successful pattern matching if all term substitutions with a candidate input text lead to the original class rationale. We can formulate the union of term substitutions via logic rules under a NeSy formulation. In particular, key-based term substitution inherently defines the atomic unit of reasoning. For instance, if we adopt POS tags as substitution keys, we identify words as atomic units. If we consider the legal rationale reported as an example in Figure 5.3, we can identify three distinct groups for the NN substitution key (see Figure 5.4):

$$\begin{aligned} G_1 &= \{NN_1\} \\ G_2 &= \{NN_2, NN_3, NN_4, NN_5\} \\ G_3 &= \{NN_6, NN_7\} \end{aligned}$$

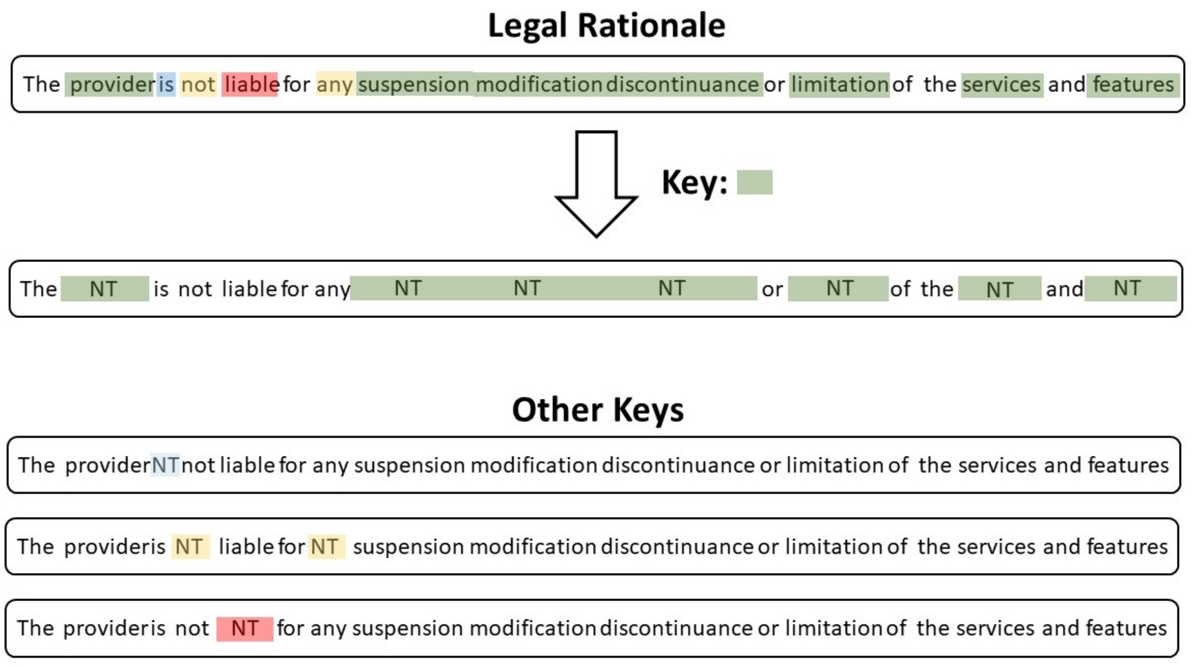

Legal Rationale
The provider is not liable for any suspension modification discontinuance or limitation of the services and features
Key:
The NT is not liable for any NT NT NT or NT of the NT and NT
Other Keys
The provider NT not liable for any suspension modification discontinuance or limitation of the services and features
The provideris NT liable for NT suspension modification discontinuance or limitation of the services and features
The provideris not NT for any suspension modification discontinuance or limitation of the services and features

Figure 5.3: Knowledge mapping mechanism via key-based term grounding. Given a substitution key, terms corresponding to that key in the legal rationale are variables for substitution. Candidates for substitution are taken from the given input clause to classify. In this example, we consider POS tags as keys for substitution. In particular, each colour denotes a POS tag. Terms corresponding to that POS tag are coloured accordingly.

these groups are defined based on the pattern matching semantic. For instance, it is sufficient that input text matches any candidate of $G_2$ to be aligned with the meaning expressed by the class rationale as $G_2$ describes equivalent actions. In GS-MLN language, we could formulate this behaviour as follows:

$$Ground(x, \$fx, nn) \wedge Group2(nn) \wedge Key(k, nn) \Rightarrow Sub(x, nn, k)$$

where *Ground* denotes the substitution predicate that operates with a feature set $\$fx$ of candidate word $x$. This approach assumes a preliminary notion of logic groups $\{G_1, G_2, G_3\}$. Such assumption is reasonable with a limited number of class rationales where a group of experts can manually identify the logic structure of each class rationale. Nonetheless, with large-sized knowledge (e.g. consider the case of knowledge generation), we require automatic representation methods to shift from textual rationales to their symbolic interpretation for pattern matching. For instance, we might adopt dependency trees to define symbolic rules like in the above Equation that rely on such intermediate representations.

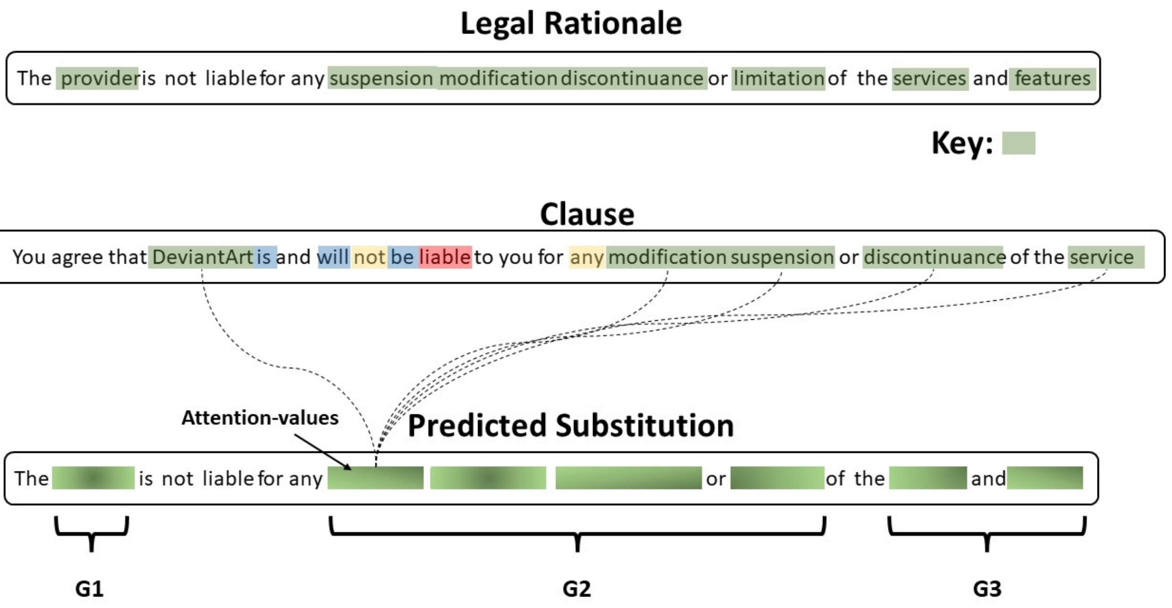

Legal Rationale
The provider is not liable for any suspension modification discontinuance or limitation of the services and features
Key:
Clause
You agree that DeviantArt is and will not be liable to you for any modification suspension or discontinuance of the service
Attention-values
Predicted Substitution
The is not liable for any or of the and
G1
G2
G3

Figure 5.4: An example of key-based term substitution. Given a substitution key, terms corresponding to that key in the input clause to classify are employed for reconstructing the original legal rationale. In this example, term substitution is done via attention-based neural mechanisms to define a differentiable operation compatible with data-driven model learning.

# Concluding Remarks

This thesis discussed the importance of knowledge integration for deep learning models. In particular, we remarked on the shortcomings of a purely data-driven approach and briefly introduced the research area of Informed Machine Learning (IML) [267] that leverages prior information (i.e. knowledge) as a solution.

In Chapter 1, we have outlined recent developments of IML in the context of Natural Language Processing (NLP) [44]. In particular, a wide variety of structured knowledge representations and applications have been presented over the years. Some notable examples we have reported are knowledge graphs, logic rules and universal dependencies. Our analysis found that structured knowledge is a valuable tool in many NLP domains, capable of addressing issues like linguistic biases and leading to higher models generalization capability. Nonetheless, in many scenarios, the acquisition process of structured knowledge is hardly realizable and may represent a critical bottleneck.

Within the scope of NLP, we have proposed to address the above situations by leveraging unstructured textual knowledge. We have motivated our objective based on the high availability of textual knowledge in a wide variety of domains. More precisely, we have identified the broad research area of Unstructured Knowledge Integration (UKI). We have defined UKI as the result of three main sub-processes. These processes regard the definition of an integration, a mapping and a validation mechanism that must be in concordance with the interpretation and role of knowledge.

From this perspective, we have provided an extensive listing of knowledge types, each defining its own set of properties to satisfy. We have discussed the major challenges of dealing with knowledge in natural language format and the advantages of integrating unstructured knowledge. Our analysis has also bridged connections with several well-known research areas that can be formulated as instances of UKI. Notable examples are reading comprehension, question answering, zero- and few-shot learning and knowledge-enhanced text generation.

Despite the potential of UKI, we have highlighted the need of defining advanced unstructured knowledge integration processes to guarantee the evaluation of specific UKI properties like *coherence*, *correctness* and model transparency. To this end, we have resumed the notion of structured knowledge extraction (KE) and identified it as a possible sub-process of UKI. In particular, we have reformulated the role of KE within the scope of UKI as the support device capable of defining abstract textual representations that are specific to the task of interest and the adopted knowledge type. To support this formulation, we have demonstrated in two experimental scenarios the benefits of KE. In parallel, we have further remarked on the issues regarding the adoption of appropriate KE processes.

In response to highlighted challenges regarding KE, we endeavoured to validate the importance of UKI by presenting a few challenging experimental scenarios where structured knowledge is not available. We have proposed memory-augmented neural networks (MANNs) [252, 276] as a convenient architecture to integrate unstructured knowledge into a deep learning model. This approach is by the taxonomy of IML regarding knowledge integration [267]. We have explored a few knowledge types under several preliminary assumptions regarding their integration. Nonetheless, despite the emerging domain-specific issues, we have shown that even simple neural-based UKI processes can improve generalization performance, model transparency, and efficient learning capabilities with limited training data.

To account for the above simplifying assumptions regarding UKI, we have discussed the general problem of guaranteeing UKI properties with challenging knowledge types like class rationales and text constraints. From this perspective, we have identified the need for a KE sub-process to define a semantic frame to satisfy these hard requirements. We have provided a unified vision of KE process instances by identifying them as a general symbolic representation. In this optic, we have envisioned adopting neuro-symbolic frameworks [17, 293] for KE as a sub-process of UKI with the above objectives.

While UKI has been introduced to compensate for certain shortcomings of structured knowledge integration, the general scope of UKI touches different research domains. It is not intended to be viewed as a distinct research field. For instance, we have discussed how structured knowledge can be considered part of UKI. Generally speaking, the definition of a complete UKI process (i.e. capable of satisfying all the knowledge integration properties) requires a suite of methodologies, including eXplainable AI (XAI) [5], KE, knowledge-compliant neural architectures [92, 94, 121, 276] and Natural Language Inference [250] to name a few.

One primary remark that we hope to convey regarding UKI is a change of perspective regarding knowledge integration. We have extensively analyzed and discussed the role and importance of knowledge within the realm of autonomous learning models. From this perspective, we would like to pinpoint one of the underlying purposes of UKI. It is undeniable that prior information represents a valuable supplement to a purely data-driven approach. In particular, we have remarked multiple times throughout this thesis the importance of a hybrid learning system that combines learning with reasoning [10, 17, 262].

However, we envision a knowledge-centric learning approach, where knowledge represents the accurate evaluation metric of an autonomous learning model. To be more precise, knowledge does not only play the role of an additional input to the system. Instead, we view knowledge as the desired output. In particular, knowledge defines a reference point for evaluation that assumes a primary role in conjunction with the specific task of interest. With a simple analogy, we could view knowledge as a system of equations describing the dynamics of an environment. Input data is then verified as being an instance of such a system. Compared to task evaluation, knowledge represents a valuable reference point in the assessment as it is directly interpretable and provides a richer informative content set. Nonetheless, it is undeniable that knowledge integration is still a highly challenging task, especially UKI.

We recognize that the work of this thesis has been limited to a few preliminary experimental scenarios with strong assumptions. However, we believe that the presented unified and broad view of structured and unstructured knowledge integration can be considered a practical step towards adopting more sophisticated and intelligent systems. We eagerly look forward to our future work in this direction.

# Appendix A

# GNN architecture calibration

We choose GCN [128] as the base GNN within our architecture. Nonetheless, we consider a few simple modifications to explore sufficiently complex model configurations. The GCN layer is mainly described via two operations: (i) aggregation via message passing and (ii) node encoding update. Our implementation follows a variant of Eq. 3.2:

$$H^{t+1} = RNN_{cell}(MLP(A, H^t; \theta^{t+1}), H^t) \tag{A.1}$$

multiple non-linearities might be applied in sequence instead of a single one. In the case of a single layer, we go back to Eq. 3.2. Lastly, a recurrent cell is applied to get the updated node encoding $H^{t+1}$, where the state is defined by the previous node encoding $H^t$. In the case of differentiable pooling, the GCN layer is enhanced with the DiffPool layer as in Eqs. 3.3, 3.4, 3.5. The GCN and DiffPool layers are viewed as a single block (see Figure 3.1) that we can stack multiple times. Also, in this case, we slightly modify the equations mentioned above by considering MLPs instead of single matrix multiplication as follows:

$$P = softmax(MLP(H)) \tag{A.2}$$

$$\tilde{H} = P^T MLP(H) \tag{A.3}$$

we transform current node encoding by applying a sequence of fully connected layers. In the case of determining $P$, the last layer is enforced to have $k$ neurons. For argument classification, we label GNN a GCN layer with a DiffPool layer with $k = 1$ for graph embedding. In contrast, we label GNN + DiffPool the GNN model with an intermediate GCN and DiffPool layers with arbitrary $k$. Lastly, the graph embedding is fed to a multi-layer perceptron (MLP) for classification. Regarding hyper-parameter tuning, we calibrate each part separately, in the following order: (1) GCN layers for the GNN model; (2) the number of layers and neurons in each layer for the final MLP; (3) the intermediate GCN and DiffPool blocks; (4) regularization hyperparameters $\alpha$ and $\delta$.

# Appendix B

# Data Collection Framework Implementation

We provide details about our implementation of the described data collection framework below. For each pipeline step, we define a time deadline to ease synchronizing participants along the data collection process.

**Participants Sign-up** We require participants to provide the following information: full name and email address. We also need participants to provide their Google Scholar profile regarding paper selection. If no Google Scholar profile is available, participants can still sign-up by manually submitting hyperlinks to the PDF files of papers about which they have enough expertise. The automatic paper retrieval step reports the top five participants' most recent papers listed in their Google Scholar profile. Participants are then asked to select two to participate in the study.

**Booking as DE** We consider `DE` dialogue sessions of one hour in length. Upon sign-up, summary information about selected papers and dialogue sessions is provided to participants. In particular, participants can add selected dialogue sessions to their calendar to ensure participation. Upon submission, summary information is provided to participants with the possibility to add chosen dialogue sessions to their calendar to ensure participation.

**Booking as P** Once the sign-up deadline has been met, our framework automatically notifies participants about the next phase. Additionally, a unique private authentication code is assigned and provided to each participant. The authentication code is used to identify participants during the study while ensuring anonymity. Regarding dialogue session booking as `P`, we split `DE` dialogue sessions into slots of 20 minutes duration. Participants are required to book four distinct dialogue session slots, each one regarding a different paper. Upon submission, summary information is provided to participants regarding their final schedule concerning both roles.

**Joining a Dialogue** Once the time deadline of the previous pipeline step has been met, our framework automatically notifies participants about their final schedule. By doing so, participants have exact information about which dialogue sessions they have to join. Our framework also supports an automatic notification system that ensures participants join each of their dialogue sessions on time.

**Why Synchronous Dialogues?** Given participants' required level of expertise and to ensure naturalness during dialogue collection, we opted for synchronous dialogue sessions. Indeed, such a choice introduces tight requirements about participants' availability and corresponding attributable effort. However, asynchronous dialogues can potentially lead to harsher drawbacks, such as an extended data collection period. Additionally, in an asynchronous dialogue, we can quickly overcome dialogue properties like the partial observability of `P` due to the absence of a time limit. Conversely, short synchronous dialogue sessions encourage both dialogue partners to maximize the exchanged information flow similar to a natural goal-oriented conversation.

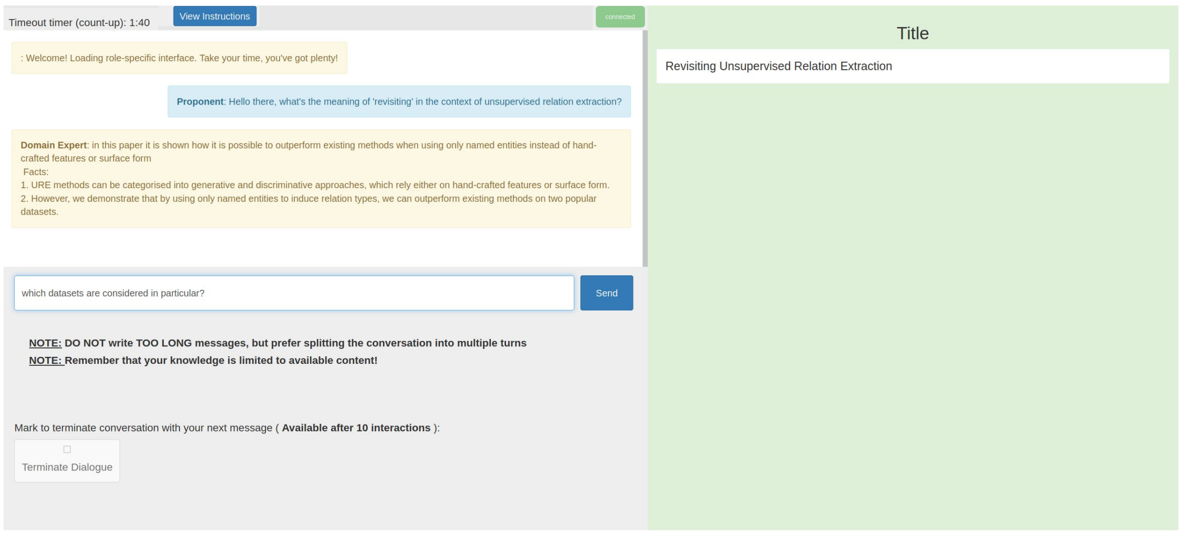
**Data Collection Interfaces** We develop role-specific collection interfaces based on the Mephisto library[1]. We opt for a standalone data collection application that does not impose additional participation steps apart from initial data submission. Upon connection between dialogue partners, associated paper content is loaded, and dialogue roles are attributed to each participant. Coherently with the given formulation, no worker payment was considered, but an award system is devised to encourage attendance further. To avoid `DE` message without supporting facts, the system notifies when no facts have been highlighted upon message sent. Additionally, a hint system alerts both dialogue partners to suggest discussing previous dialogue turns to encourage opinions exchange. Figure B.1 reports the data collection interfaces for both dialogue roles. Both interfaces reflect the asymmetrical nature of the dialogue by restricting information to individual roles.

**Implementation Limitations** The major limitation of our implementation concerns the chosen interface library. More precisely, the Mephisto library has been developed for crowd-sourcing. In crowdsourcing, there are no restrictions about joining a specific task instance, i.e. a dialogue in our setting. However, we require advanced access control operations to guarantee participants can only enter their scheduled dialogue sessions. For this reason, we couldn't allow

[1]https://github.com/facebookresearch/Mephisto

parallel dialogue sessions. This limitation significantly reduced the number of possible dialogue sessions and, thus, dialogues. We opted for an existing data collection platform to avoid the burden of building one from scratch. Nonetheless, there might be a need to define a custom crowdsourcing platform to overcome this major limitation. We leave platform improvements as future work.

**Collection Issues** Considering the two data collection rounds, we identified 68 dialogue sessions in total. However, events like participant disconnection, reply timeout, and session skipping lowered the number of dialogues with a sufficient number of messages to 41. We could overcome some encountered issues with a more advanced data collection implementation. We leave this as future work.



: Welcome! Loading role-specific interface. Take your time, you've got plenty!

**Proponent**: Hello there, what's the meaning of 'revisiting' in the context of unsupervised relation extraction?

**Domain Expert**: in this paper it is shown how it is possible to outperform existing methods when using only named entities instead of hand-crafted features or surface form
Facts:
1. URE methods can be categorised into generative and discriminative approaches, which rely either on hand-crafted features or surface form.
2. However, we demonstrate that by using only named entities to induce relation types, we can outperform existing methods on two popular datasets.

which datasets are considered in particular?

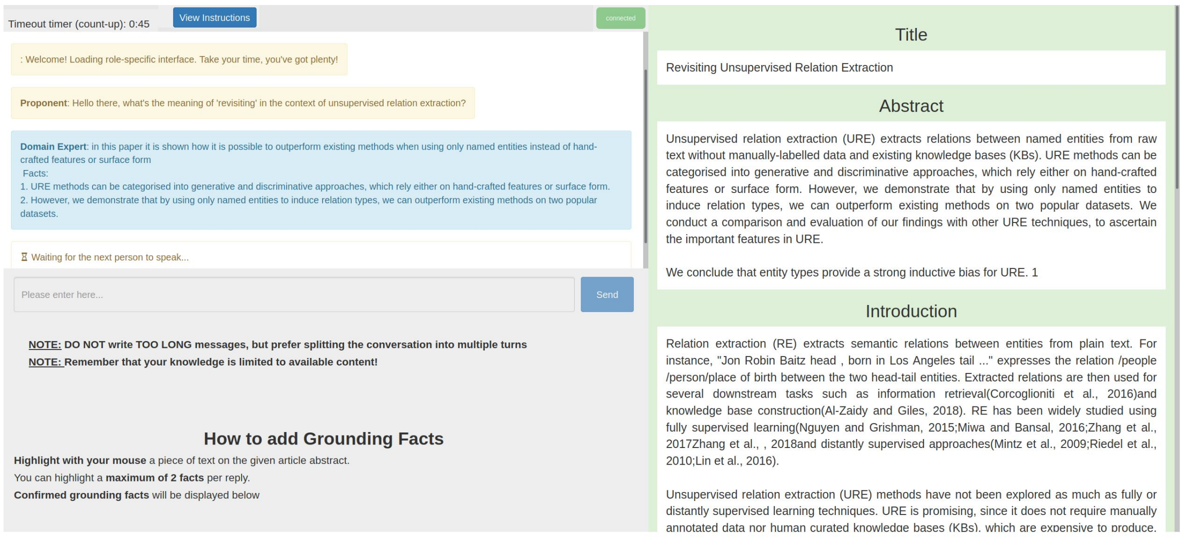
**NOTE:** DO NOT write TOO LONG messages, but prefer splitting the conversation into multiple turns
**NOTE:** Remember that your knowledge is limited to available content!

Mark to terminate conversation with your next message ( **Available after 10 interactions** ):

Title

Revisiting Unsupervised Relation Extraction

(a)



: Welcome! Loading role-specific interface. Take your time, you've got plenty!

**Proponent**: Hello there, what's the meaning of 'revisiting' in the context of unsupervised relation extraction?

**Domain Expert**: in this paper it is shown how it is possible to outperform existing methods when using only named entities instead of hand-crafted features or surface form
Facts:
1. URE methods can be categorised into generative and discriminative approaches, which rely either on hand-crafted features or surface form.
2. However, we demonstrate that by using only named entities to induce relation types, we can outperform existing methods on two popular datasets.

⌛ Waiting for the next person to speak...

Please enter here...

**NOTE:** DO NOT write TOO LONG messages, but prefer splitting the conversation into multiple turns
**NOTE:** Remember that your knowledge is limited to available content!

How to add Grounding Facts

**Highlight with your mouse** a piece of text on the given article abstract.
You can highlight a **maximum of 2 facts** per reply.
**Confirmed grounding facts** will be displayed below

Title

Revisiting Unsupervised Relation Extraction

Abstract

Unsupervised relation extraction (URE) extracts relations between named entities from raw text without manually-labelled data and existing knowledge bases (KBs). URE methods can be categorised into generative and discriminative approaches, which rely either on hand-crafted features or surface form. However, we demonstrate that by using only named entities to induce relation types, we can outperform existing methods on two popular datasets. We conduct a comparison and evaluation of our findings with other URE techniques, to ascertain the important features in URE.

We conclude that entity types provide a strong inductive bias for URE. 1

Introduction

Relation extraction (RE) extracts semantic relations between entities from plain text. For instance, "Jon Robin Baitz head , born in Los Angeles tail ..." expresses the relation /people /person/place of birth between the two head-tail entities. Extracted relations are then used for several downstream tasks such as information retrieval(Corcoglioniti et al., 2016)and knowledge base construction(Al-Zaidy and Giles, 2018). RE has been widely studied using fully supervised learning(Nguyen and Grishman, 2015;Miwa and Bansal, 2016;Zhang et al., 2017Zhang et al., , 2018and distantly supervised approaches(Mintz et al., 2009;Riedel et al., 2010;Lin et al., 2016).

Unsupervised relation extraction (URE) methods have not been explored as much as fully or distantly supervised learning techniques. URE is promising, since it does not require manually annotated data nor human curated knowledge bases (KBs), which are expensive to produce.

(b)

Figure B.1: Data collection interfaces for (a) P and (b) DE. On the right side, title, abstract and introduction sections of the paper are reported for DE. Conversely, P's view is solely restricted to the paper title.

# Appendix C

# Legal Rationales for Unfair Clause Detection

We report the complete legal rationales listing for the remaining unfairness category of interest.

**Limitation of liability clauses**

Service providers often dedicate a considerable portion of their ToS to disclaiming and limiting liabilities. Clauses falling under this category stipulate that the duty to pay damages is limited or excluded for certain kinds of losses and under certain conditions. Most of the circumstances under which these limitations are declared significantly affect the balance between the parties' rights and obligations and are unlikely to pass the Directive's unfairness test. In particular, clauses excluding liability for broad categories of losses or causes were marked as potentially unfair, including those containing blanket phrases like "to the fullest extent permissible by law". Conversely, we marked clauses meant to reduce, limit, or exclude the liability for physical injuries, intentional harm, or gross negligence as clearly unfair [154, 176]. The analysis of the dataset enabled us to identify 19 legal rationales for (potentially) unfair limitation of liability, which maps different questionable circumstances under which the ToS reduce or exclude liability for losses or injuries. We defined a corresponding identifier [ID] for each rationale, as shown in Table C.1.

Table C.1: Legal rationales for the legal qualification of Limitation of Liability unfairness.

| ID | Legal Rationale |
|---|---|
| extent | Except as required by law, or to the fullest extent permissible by applicable law, the provider is not liable, and users are solely responsible for ensuring that the Terms of Use/Service comply with all laws, rules and regulations, and the use of the platform its on their own risk. |

| | |
|---|---|
| gen | The clause introduces a general and non-specific limitation and exclusion of liability, such as liability for various things, liability arising out of or in connection with the service and the Terms. |
| discontinuance | The provider is not liable for any technical problems, failure, inability to use the service, suspension, disruption, modification, discontinuance, unavailability of service, any unilateral change, unilateral termination, unilateral limitation including limits on certain features and services or restriction to access to parts or all of the Service without notice |
| compharm | The provider is not liable for harm or damage to hardware and software, including viruses, malware, worms, trojan horses, or any similar contamination or destructive program. |
| anydamage | The provider is not liable for any special, direct and/or indirect, punitive, incidental or consequential damage, including negligence, and broad categories of damages, including harm or failure. |
| amount | The compensation for liability or aggregate liability is limited to, or should not exceed, a certain total amount, or that the sole remedy is to stop using the service and cancel the account, or that you can’t recover any damages or losses. |
| thirdparty | The provider is not liable for any action, errors, omissions, representations, warranties, breaches or negligence taken from third parties, third-party providers services, suppliers or other people, acts of any government and authority, including service and products, additional costs, copyright compliance, legality or decency of material, content and link posted by others, including users. |
| security | The provider is not liable for any damage deriving from a security breach, including any unauthorised access, alteration and modification of data, data transmission. |
| dataloss | The provider is not liable for any disclosure, damage, destruction, corruption, failure to store or loss of data and material. |
| reputation | The provider is not liable for reputational and goodwill damages, loss. |
| anyloss | The provider is not liable for any loss , or broad categories of loss, resulting from the use of the service and or of the website, including lost profits, lost opportunity, lost business or lost sales, data loss, loss of goodwill, loss of reputation. |
| awareness | The provider is not liable even if he was, or should have been, aware or have been advised about the possibility of any damage or loss. |

| | |
|---|---|
| contractfailure | The provider is not liable for any failure in performing contract and terms, obligations, including unavailability or failure in providing products and services, breach of agreement, lack of performance. |
| dataloss | The provider is not liable for any loss of data. |
| ecoloss | The provider is not liable for any loss of profits, loss of income, lost opportunity, lost business or lost sales, loss of revenue. |
| content | The provider is not liable for any information stored or processed within the Services, inaccuracies or error of information, content and material posted, software, products and services on the website, including copyright violation, defamation, slander, libel, falsehoods, obscenity, pornography, profanity, or objectionable material. |
| liabtheories | The provider is not liable under different theories of liability, including tort law, contract law, strict liability, statutory liability, product liability and other liability theories. |
| grossnegligence | The provider is not liable for gross negligence. |
| injury | The provider is not liable for intentional offence and damage, physical or personal injury, including, emotional distress and death. |

As noted above, each (potentially) unfair `LTD` clause within the data set may be relevant, and thus indexed with the corresponding IDs, under more than one legal rationale. As an example, consider the following clause taken from the Box terms of service (retrieved on August 2017) and classified as potentially unfair:

```
To the extent not prohibited by law, in no event will Box, its affiliates resellers,
officers, employees, agents, suppliers or licensors be liable for:  any direct, incidental,
special, punitive, cover or consequential damages (including, without limitation, damages
for lost profits, revenue, goodwill, use or content) however caused, under any theory of
liability, including, without limitation, contract, tort, warranty, negligence or otherwise,
even if Box has been advised as to the possibility of such damages.
```

The clause above, to the extent not prohibited by law, limits the provider's liability by kind of damages, i.e., broad category of losses (e.g. loss of data, economic loss and loss of reputation); by standard of care, since it states that the provider will never be liable even in case of negligence and awareness of the possibility of damages; by causal link (e.g. special, incidental and consequential damages) as well as by any liability theory. As a consequence, the clause has been linked to the following identifiers: `extent, anydamage, reputation, dataloss, ecoloss, awareness, liabtheories`. As a further example, consider the following clause taken from the Endomondo terms of service (retrieved on May 2018) and classified as clearly unfair:

> `Except as otherwise set out in these Terms, and to the maximum extent permitted by applicable law, we are not responsible or liable, either directly or indirectly, for any injuries or damages that are sustained from your physical activities or your use of, or inability to use, any Services or features of the Services, including any Content or activities that you access or learn about through our Services (e.g., a Third-Party Activity such as a yoga class), even if caused in whole or part by the action, inaction or negligence of Under Armour or by the action, inaction or negligence of others.`

To the maximum extent permitted by the law, the clause above limits the service provider's liability by the causal link with broad categories of potential damages (i.e., to encompass direct and/or indirect damages), by cause (i.e., service interruption and/or unavailability, third party actions and content published, stored, and processed within the service), and by kind, in particular for personal injury, resulting from an act or an omission of the supplier. Thus, it has been linked to the following identifiers: `extent, anydamage, injury, thirdparty, content, discontinuance`.

### Content Removal

Content removal clauses give the provider a right to modify and/or delete user's content, including in-app purchases, and sometimes specifies the conditions under which the service provider may do so. As in the case of unilateral termination, clauses that indicate conditions for content removal were marked as potentially unfair, whereas clauses stipulating that the service provider may remove content in his full discretion, and/or at any time for any or no reasons and/or without notice nor possibility to retrieve the content were marked as clearly unfair. Under this category, we identified 17 legal rationales, shown in Table C.2.

Table C.2: Legal rationales for the Content Removal category.

| **ID** | **Legal Rationale** |
|---|---|
| `nonotice` | `The provider has the right to remove content and material without prior notice` |
| `noretrieve` | `The provider has the right to remove content and material without the possibility to retrieve such content and material.` |
| `fulldiscretion` | `The provider has the right to remove content and material for any reason, at its full discretion.` |
| `lawviolation` | `The provider has the right to remove content and material in order to comply with applicable law, if he believes in good faith that there is a case of law violation, including intellectual property infringments.` |
| `termviolation` | `The provider has the right to remove content and material if he believes that there is a case violation of terms such as acount tranfer, policies, standard, code of conduct.` |

| | |
|---|---|
| `objcontent` | `The provider has the right to remove content and material that he considers to be offensive, obscene, abusive, harmful, objectable, inaccurate, inappropriate.` |
| `comply` | `The provider has the right to remove content and material in order to comply with the order or request of a court, law enforcement , other administrative agency or governmental body.` |
| `serviceprotection` | `The provider has the right to remove content and material that he considers to be harmful to or as creating threats for the provider's property, Site or Services, or consumers.` |
| `criteriafailure` | `The provider has the right to remove content and material if there is a failure in meeting any applicable quality or eligibility criteria.` |
| `complaint` | `The provider has the right to remove content and material in case of complaints about users' performance, conduct, published content and information.` |
| `rating` | `The provider has the right to remove content and material in case of poor Ratings or Reviews.` |
| `fraudprevention` | `The provider has the right to remove content and material in order to prevent fraud and illegal activities.` |
| `personalsafety` | `The provider has the right to remove content and material to protect personal safety.` |
| `liability` | `The provider has the right to remove content and material if they could subject the provider to liability.` |
| `tpright` | `The provider has the right to remove content and material if they constitute a violation of third party rights, including trademarks.` |
| `susp` | `The provider has the right to remove content and material upon suspension or termination.` |
| `inactive` | `The provider has the right to remove content and material in case of users' inactivity.` |

Each (potentially) unfair clause falling under the content removal category has been indexed with one or more identifiers of rationales. Consider, for instance, the following examples taken from the terms of service of TikTok (retrieved on October 2018) and Pokemon GO (retrieved on July 2016):

> `In addition, we have the right - but not the obligation - in our sole discretion to remove, disallow, block or delete any User Content (i) that we consider to violate these Terms, or (ii) in response to complaints from other users or third parties, with or without notice and without any liability to you.`
>
> `Niantic further reserves the right to remove any User Content from the Services at any time and without notice and for any reason.`

The first clause above, previously classified as potentially unfair, has been linked to the following IDs: `termviolation`, `complaint`, `nonotice`, `tpright`. Conversely, the second clause, previously classified as clearly unfair, has been linked to the IDs `IDs: nonotice, fulldiscretion`. Such practices would have no means to influence their content.

### Unilateral Termination

The unilateral termination clause gives the provider the right to suspend and terminate the service or the contract and sometimes details the circumstances under which the provider claims to have a right to do so. From the consumer's perspective, a situation where the agreement may be dissolved at any time and for any reason could seriously undermine the whole purpose of entering into the contract.

These clauses may skew the balance of power between the parties and be considered (potentially) unfair whenever the consumer has a reasonable interest in preserving the contract's longevity, given the foreseeably invested time and effort in the services, e.g., by importing and storing digital content. This is all the more true if the trader does not provide a reasonably long notice period allowing the consumer to migrate to another service (e.g., withdrawing and transferring all the digital content elsewhere).

Unilateral termination clauses that specify reasons for termination were marked as potentially unfair. In contrast, clauses stipulating that the service provider may suspend or terminate the service for any reason and/or without notice were marked as clearly unfair. Under this (potentially) unfair clause category, we identified 28 different legal rationales.

Table C.3: Legal rationales for the legal qualification of Unilateral Termination unfairness.

| ID | Legal Rationale |
|---|---|
| fraud_abuse_illegal | The contract or access may be terminated where the user has been engaging in illegal or unlawful activity, including fraudulent behaviour, abusive, misusive or otherwise harmful behaviour, or for reasons of safety or fraud prevention. |
| breach | The contract or access can be terminated where the user fails to adhere to its terms, or community standards, or the spirit of the ToS or community terms, including inappropriate behaviour, using cheats or other disallowed practices to improve their situation in the service, deriving disallowed profits from the service, or interfering with other users' enjoyment of the service or otherwise puts them at risk, or is investigated under any suspision of misconduct. |
| no_grounds | The clause mentions the contract or access may be terminated but does not state the grounds for termination. |
| misinfo | The clause mentions the contract or access may be terminated where the user has provided false, outdated or incomplete information. |

| | |
|---|---|
| infring_tp_rights | The contract or access may be terminated in cases of infringement upon rights of others, including copyrights or other intellectual property rights, including termination for repeat infringers. |
| multiple | The contract or access may be terminated in cases of a single user holding or controlling multiple accounts, or multiple use of a single account. |
| cred_security | The contract or access may be terminated where the user fails to maintain the security of the login credentials and/or a security breach occurs. |
| dormant | The contract or access may be terminated where the account has been left dormant for a prescribed time. |
| user_bad_rep | The contract or access may be terminated where the user fails to maintain a prescribed level of reputation. |
| reference | The contract or access may be terminated but refers to grounds formulated elsewhere. |
| content_violation | The contract or access may be terminated where the user has entered content into the service which is, or is deemed to be, infringing upon the rights of others or in violation with the terms of service. |
| payments | The contract or access may be terminated where the user has not been meeting their payment obligations, or withdrawing payments, e.g. via chargeback. |
| gen_rights_violation | The clause generally states the contract or access may be terminated where the user has violated the rights of the service provider or other entities. |
| over_quota | The contract or access may be terminated where the user has been violating the time, storage or other limits of the service. |
| insolvency | The contract or access may be terminated where one of the parties has been declared insolvent, bankrupt, has a court receiver or a similar officer appointed, or proceedings are pending in regard to any of these. |
| shutdown | The contract or access may be terminated where the service is being shutdown or ceases to be available for any other reasons. |
| no_consent | The contract or access may be terminated where the user's consent is missing or withdrawn, or where the user otherwise objects to the terms, policy or any change thereof. |
| sex_of | The contract or access may be terminated where the user is a registered sex offender, or engaged or attempted to engage in sexual conduct with minors, or has been involved with child pornography. |
| parole | The contract or access may be terminated where the user has engaged in a parole or probation violation. |
| viability_-eligibility | The contract or access may be terminated where the provision of the service to the user is no longer economically viable, or where the user is not eligible for the service. |

| | |
|---|---|
| dispute | The contract or access may be terminated where the user engages in a dispute with the service provider or owner. |
| legal_reasons | The contract or access may be terminated to comply with legal requirements, or as a result of a request put in by authorities, or for broadly specified legal reasons. |
| tech_reasons | The clause broadly states the contract or access may be terminated for technical reasons. |
| any_reasons | The clause generally states the contract or access may be terminated for any reason, without cause or leaves room for other reasons which are not specified. |
| force_majeure | The clause generally states the contract or access may be terminated in an event of a force majeure, act of God or other unforeseen events of a similar nature. |
| providers_exposure | The contract or access may be terminated where the user's actions or content create a risk of legal exposure, or damage to the provider's reputation. |
| protect_rights | The clause generally states the contract or access may be terminated to protect the rights and/or interests of the service provider or a third party. |
| no_notice | The clause generally states that the contract or access may be terminated without notice or simply posting it on the website and/or the trader is not required to observe a reasonable period for termination. |

As examples, consider the following clauses, taken from the DeviantArt (effective date not available) and Academia (retrieved on May 2017) terms of service:

> `Furthermore, you acknowledge that DeviantArt reserves the right to terminate or suspend accounts that are inactive, in DeviantArt's sole discretion, for an extended period of time (thus deleting or suspending access to your Content)`

> `Academia.edu reserves the right, at its sole discretion, to discontinue or terminate the Site and Services and to terminate these Terms, at any time and without prior notice.`

The first clause above, previously classified as potentially unfair, has been linked to the `dormant` ID, since it states that the service provider will suspend or terminate apparently dormant accounts, also deleting or suspending access to consumers' digital contents. Conversely, the second clause was previously classified as clearly unfair and it has been linked to the IDs `no_grounds, no_notice` since the service provider claims the right to unilaterally terminate both the contract and the service without prior notice and grounds for termination are completely missing.

### Unilateral Changes

Under this category, we identified 7 different legal rationales, as reported in Table C.4, mapping the different circumstances under which service providers claim their right to unilaterally amend and modify the terms of service and/or the service. Unilateral change clauses were always considered as potentially unfair, since the ECJ has not yet issued a judgment in this regard, though the Annex to the Directive contains several examples supporting such a qualification.

Unilateral change clauses are particularly worrisome in cases where the proposed amendment significantly impact the consumers' rights, thus creating a disproportionate balance between the parties. This is particularly true whenever consumers are either not given any opt-out options, where no consent to the new terms is requested, or where no notification to the consumers is given.

Table C.4: Legal rationales for the Unilateral Change category.

| **ID** | **Legal Rationale** |
|---|---|
| anyreason | The provider has the right for unilateral change of the contract/services/goods/features for any reason at its full discretion, at any time. |
| nowarning | The provider has the right for unilateral change of the contract/services/goods/features with no notice to the consumer . |
| justposted | The provider has the right for unilateral change of the contract/services/goods/features where the notification of changes is left at a full discretion of the provider such as by simply posting the new terms on their website without a notification to the consumer. |
| consresp | The provider has the right for unilateral change of the contract/services/goods/features where it is the consumer's responsibility to regularly check the terms for any updates. |
| againsterms | The provider has the right for unilateral change of the contract/services/goods/features if the consumer violates the Terms (as a consequence only limited or no services might be provided). |
| lawchange | The provider has the right for unilateral change of the contract/services/goods/features to reflect changes in law, regulatory requirements at their own discretion. |
| update | The provider has the right for unilateral change of the contract/services/goods/features to maintain a level of flexibility to amend and update services, including discontinuation. |

As relevant examples, consider the following clauses taken from the Academia (retrieved on May 2017) and Endomondo (retrieved on January 2016) terms of service:

```
     Academia.edu reserves the right, at its sole discretion, to modify the Site, Services
and these Terms, at any time and without prior notice.
```

```
    With new products, services, and features launching all the time, we need the
flexibility to make changes, impose limits, and occasionally suspend or terminate certain
offerings.
```

The first clause above is representative of the largest group of the unilateral change clauses, stating that the provider has the right for unilateral change of the contract and/or the services and/or the provided goods and/or features, for any reason, at its full discretion and at any time and without notice. Thus, it has been linked to the following IDs: `anyreason, nowarning`. The second clause, stating that the service provider has the right to make generic unilateral changes to maintain a level of flexibility, has been linked to the ID `update`.

# Appendix D

# Partial Strong Supervision in Unfair Clause Detection

To further assess the contribution of direct rationales supervision, i.e. *strong supervision*, Figures D.1-D.5 compare ground truth total rationales usage with respect to employed MANN models. Categories are sorted (descending order) based on ground truth values. In some cases (see Figure D.1-D.2), strong supervision significantly drives MANN models towards ground truth usage, without negatively affecting the performance. In other cases, for instance CR and LTD (Figure D.3-D.4) it does not appear to have any significant effect. More fine-grained results seen in Table 4.7 also confirm these observations. These empirical results could be due to the way rationales are formulated and distributed across unfair examples and/or to the strength of strong supervision regularization during training. For the latter case, Figure D.5 shows an additional example where *strong supervision* is excessively affecting the training phase, causing memory over-usage. We plan to further investigate this issue in future extensions.

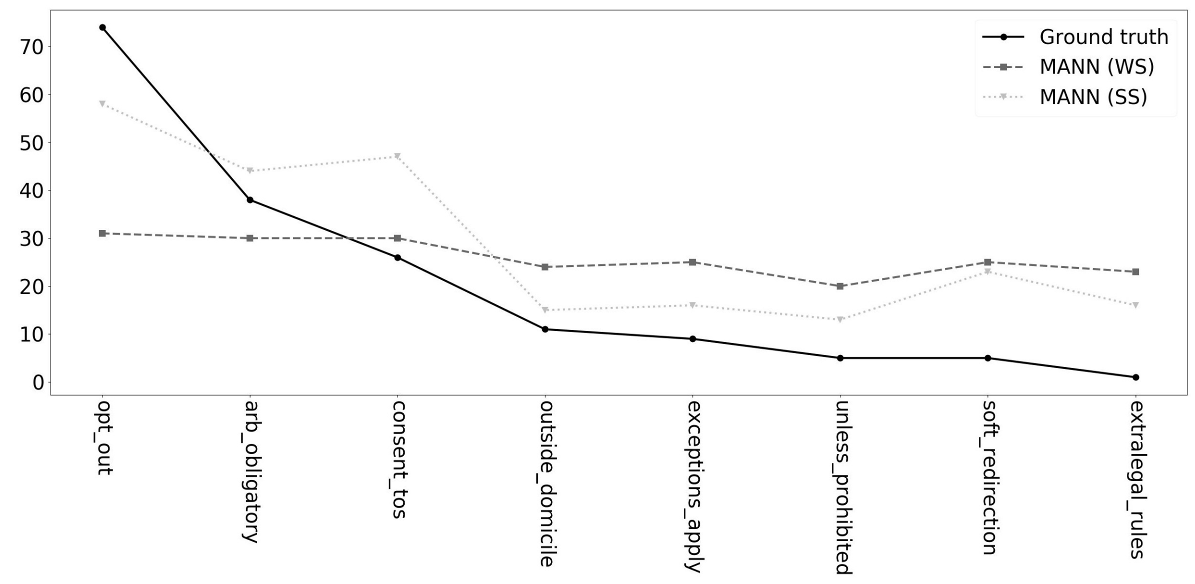


Figure D.1: Total rationales usage in ground truth unfair examples concerning Arbitration (A) unfairness category.

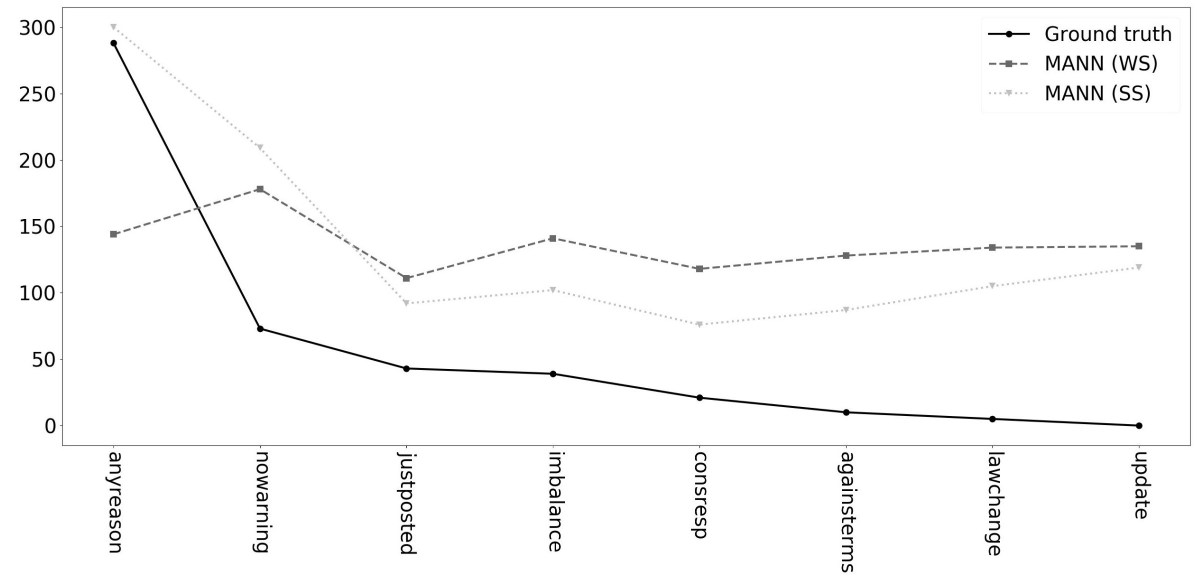


Figure D.2: Total rationales usage in ground truth unfair examples concerning Unilateral Change (CH) unfairness category.

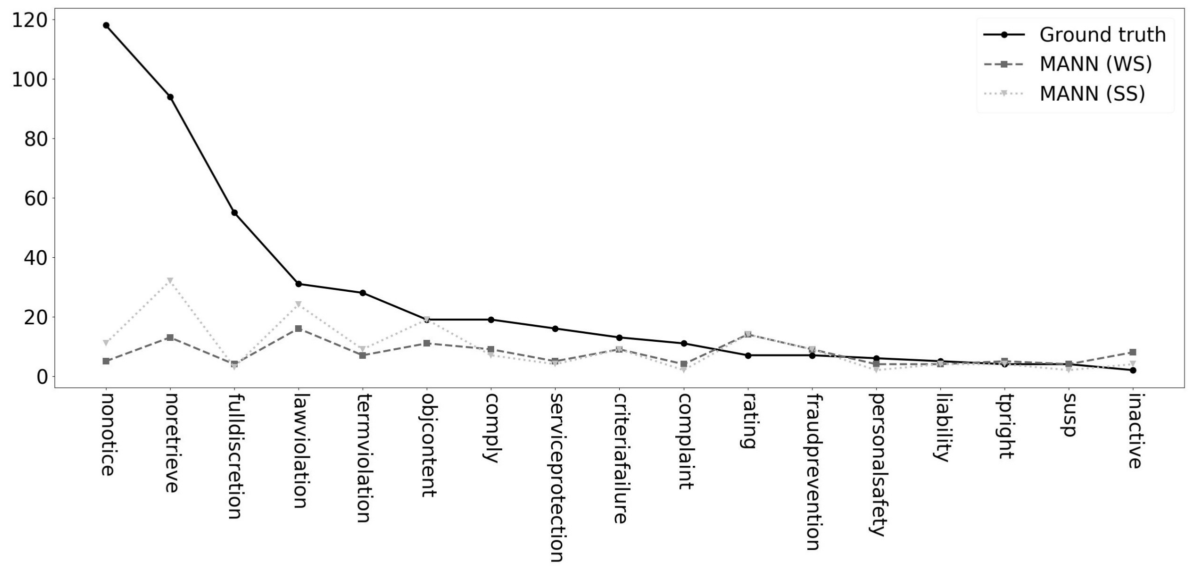


Figure D.3: Total rationales usage in ground truth unfair examples concerning Content Removal (CR) unfairness category.

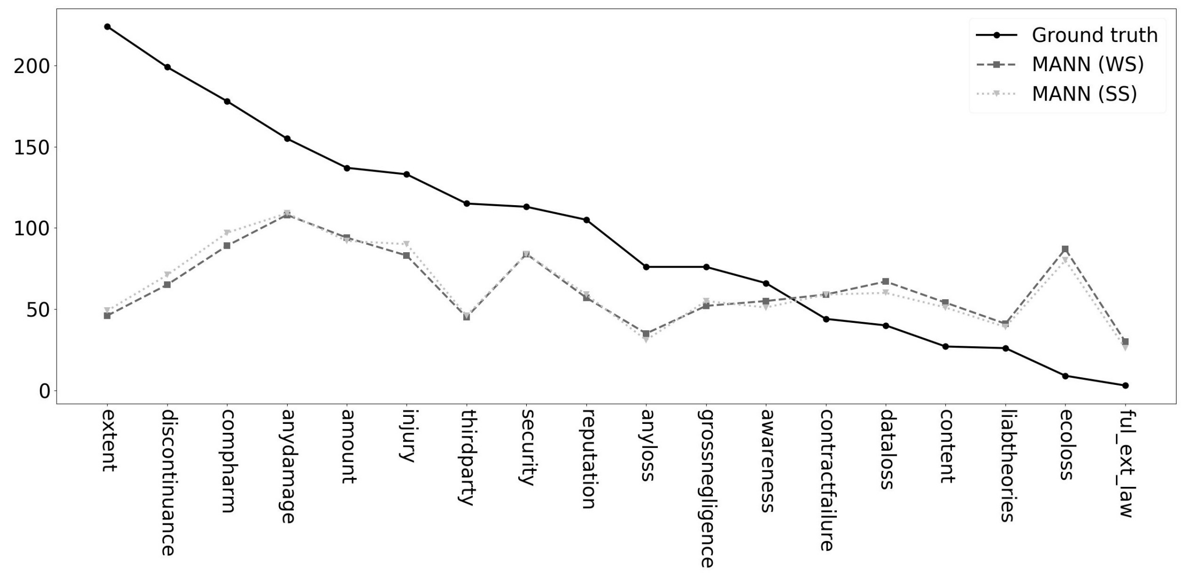


Figure D.4: Total rationales usage in ground truth unfair examples concerning Limitation of Liability (LTD) unfairness category.

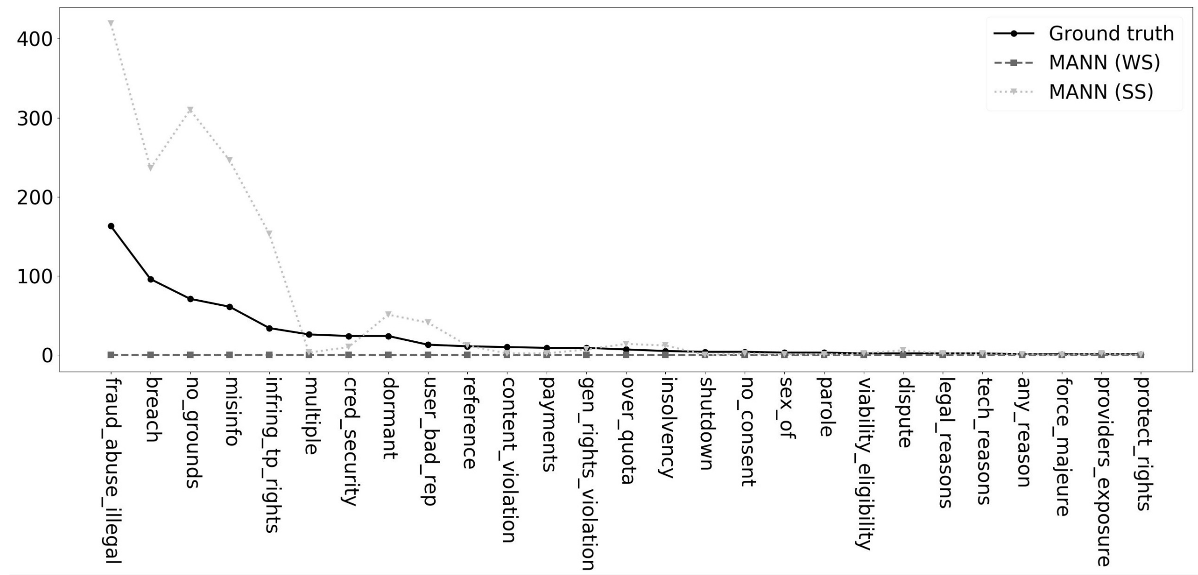


Figure D.5: Total rationales usage in ground truth unfair examples concerning Unilateral Termination (TER) unfairness category.

# Bibliography

[1] Abeer ALDayel and Walid Magdy. Stance detection on social media: State of the art and trends. *Information Processing & Management*, 58(4):102597, 2021.

[2] Muhammad Amith, Yaoyun Zhang, Hua Xu, and Cui Tao. Knowledge-based approach for named entity recognition in biomedical literature: a use case in biomedical software identification. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 386–395. Springer, 2017.

[3] Prithviraj Ammanabrolu and Mark Riedl. Transfer in deep reinforcement learning using knowledge graphs. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 1–10, Hong Kong, November 2019. Association for Computational Linguistics.

[4] Ion Androutsopoulos and Prodromos Malakasiotis. A survey of paraphrasing and textual entailment methods. *ArXiv*, abs/0912.3747, 2010.

[5] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82–115, 2020.

[6] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In Karl Aberer, Key-Sun Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *The Semantic Web*, pages 722–735, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[7] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021.

[8] Stephen H. Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. Hinge-loss markov random fields and probabilistic soft logic, 2017.

[9] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In *LAW@ACL*, 2013.

[10] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andrew Ballard, Justin Gilmer, George Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks, 2018.

[11] Iz Beltagy, Kyle Lo, and Arman Cohan. Scibert: Pretrained language model for scientific text. In *EMNLP*, 2019.

[12] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *ArXiv*, abs/2004.05150, 2020.

[13] Nihar Bendre, Hugo Terashima Marín, and Peyman Najafirad. Learning from few samples: A survey, 2020.

[14] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*, pages 437–478. Springer, 2012.

[15] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proc. of the 26th Int. Conf. on Machine Learning*, ICML '09, page 41–48. ACM, 2009.

[16] Doron L. Bergman. Symmetry constrained machine learning. *Intelligent Systems and Applications*, page 501–512, Aug 2019.

[17] Tarek R. Besold, Artur d'Avila Garcez, Sebastian Bader, Howard Bowman, Pedro Domingos, Pascal Hitzler, Kai-Uwe Kuehnberger, Luis C. Lamb, Daniel Lowd, Priscila Machado Vieira Lima, Leo de Penning, Gadi Pinkas, Hoifung Poon, and Gerson Zaverucha. Neural-symbolic learning and reasoning: A survey and interpretation, 2017.

[18] Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. One spring to rule them both: Symmetric amr semantic parsing and generation without a complex pipeline. In *AAAI*, 2021.

[19] Michele Bevilacqua, Tommaso Pasini, Alessandro Raganato, and Roberto Navigli. Recent trends in word sense disambiguation: A survey. In *IJCAI*, 2021.

[20] Bin Bi, Chen Wu, Ming Yan, Wei Wang, Jiangnan Xia, and Chenliang Li. Generating well-formed answers by machine reading with stochastic selector networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7424–7431, Apr. 2020.

[21] Jiang Bian, Bin Gao, and Tie-Yan Liu. Knowledge-powered deep learning for word embedding. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 132–148. Springer, 2014.

[22] Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. Spectral clustering with graph neural networks for graph pooling, 2020.

[23] Mahdi Bohlouli, Jens Dalter, Mareike Dornhöfer, Johannes Zenkert, and Madjid Fathi. Knowledge discovery from social media using big data-provided sentiment analysis (somabit). *Journal of Information Science*, 41(6):779–798, 2015.

[24] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250, 2008.

[25] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. Large-scale simple question answering with memory networks. *CoRR*, abs/1506.02075, 2015.

[26] Antoine Bordes and Jason Weston. Learning end-to-end goal-oriented dialog. *CoRR*, abs/1605.07683, 2016.

[27] Matko Bošnjak, Tim Rocktäschel, Jason Naradowsky, and Sebastian Riedel. Programming with a differentiable forth interpreter. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 547–556. JMLR.org, 2017.

[28] Samuel R. Bowman, Christopher Potts, and Christopher D. Manning. Recursive neural networks can learn logical semantics. In *Proceedings of the 3rd Workshop on Continuous*

*Vector Space Models and their Compositionality*, pages 12–21, Beijing, China, July 2015. Association for Computational Linguistics.

[29] Eric Brill. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Comput. Linguistics*, 21(4):543–565, 1995.

[30] Jamie Callan and Teruko Mitamura. Knowledge-based extraction of named entities. In *Proceedings of the eleventh international conference on Information and knowledge management*, pages 532–537, 2002.

[31] Erik Cambria, Daniel Olsher, and Dheeraj Rajagopal. Senticnet 3: a common and common-sense knowledge base for cognition-driven sentiment analysis. In *Twenty-eighth AAAI conference on artificial intelligence*, 2014.

[32] Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. e-snli: Natural language inference with natural language explanations, 2018.

[33] Yixin Cao, Lifu Huang, Heng Ji, Xu Chen, and Juanzi Li. Bridge text and knowledge by learning multi-prototype entity mention embedding. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1623–1633, 2017.

[34] Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. Retrieve, rerank and rewrite: Soft template based neural summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 152–161, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[35] Lucas Carstens and Francesca Toni. Towards relation based argumentation mining. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 29–34, Denver, CO, June 2015. Association for Computational Linguistics.

[36] Sarath Chandar, Sungjin Ahn, Hugo Larochelle, Pascal Vincent, Gerald Tesauro, and Yoshua Bengio. Hierarchical memory networks. *CoRR*, abs/1605.07427, 2016.

[37] Jiaao Chen, Jianshu Chen, and Zhou Yu. Incorporating structured commonsense knowledge in story completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6244–6251, 2019.

[38] Jiaoyan Chen, Yuxia Geng, Zhuo Chen, Ian Horrocks, Jeff Z. Pan, and Huajun Chen. Knowledge-aware zero-shot learning: Survey and perspective. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence,*

*IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 4366–4373. ijcai.org, 2021.

[39] Xinyun Chen, Chang Liu, and Dawn Song. Tree-to-tree neural networks for program translation. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 2552–2562, 2018.

[40] Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. In Jian Su, Xavier Carreras, and Kevin Duh, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 551–561. The Association for Computational Linguistics, 2016.

[41] Jason P. C. Chiu and Eric Nichols. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, 4:357–370, 2016.

[42] Jaemin Cho, Min Joon Seo, and Hannaneh Hajishirzi. Mixture content selection for diverse sequence generation. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3119–3129. Association for Computational Linguistics, 2019.

[43] Edward Choi, Mohammad Taha Bahadori, Le Song, Walter F Stewart, and Jimeng Sun. Gram: graph-based attention model for healthcare representation learning. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 787–795, 2017.

[44] Gobinda G Chowdhury. Natural language processing. *Annual review of information science and technology*, 37(1):51–89, 2003.

[45] Giuseppe Contissa, Koen Docter, Francesca Lagioia, Marco Lippi, Hans-Wolfgang Micklitz, Przemyslaw Palka, Giovanni Sartor, and Paolo Torroni. Automated processing of privacy policies under the EU General Data Protection Regulation. In *Legal Knowledge and Information Systems: JURIX 2018: The Thirty-first Annual Conference*, volume 313 of *Frontiers in Artificial Intelligence and Applications*, pages 51–60. IOS Press, 2018.

[46] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[47] Pádraig Cunningham and Sarah Jane Delany. k-nearest neighbour classifiers - a tutorial. *ACM Computing Surveys*, 54(6):1–25, Jul 2022.

[48] Kristijonas Cyras, Antonio Rago, Emanuele Albini, Pietro Baroni, and Francesca Toni. Argumentative XAI: A survey. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 4392–4399. ijcai.org, 2021.

[49] Sławomir Dadas. Combining neural and knowledge-based approaches to named entity recognition in polish. In *International Conference on Artificial Intelligence and Soft Computing*, pages 39–50. Springer, 2019.

[50] Marco Damonte and Shay B. Cohen. Structural neural encoders for amr-to-text generation. In *NAACL*, 2019.

[51] Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. A dataset of information-seeking questions and answers anchored in research papers. *ArXiv*, abs/2105.03011, 2021.

[52] Johannes Daxenberger, Steffen Eger, Ivan Habernal, Christian Stab, and Iryna Gurevych. What is the essence of a claim? cross-domain claim identification. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2055–2066. Association for Computational Linguistics, 2017.

[53] Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. Universal Dependencies. *Computational Linguistics*, 47(2):255–308, 07 2021.

[54] Luc De Raedt and Angelika Kimmig. Probabilistic (logic) programming concepts. *Mach. Learn.*, 100(1):5–47, jul 2015.

[55] Ghaith Dekhili, Ngoc Tan Le, and Fatiha Sadat. Augmenting named entity recognition with commonsense knowledge. In *Proceedings of the 2019 Workshop on Widening NLP*, 2019.

[56] Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel. Lifted rule injection for relation embeddings. In Jian Su, Xavier Carreras, and Kevin Duh, editors, *Proceedings*

*of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1389–1399. The Association for Computational Linguistics, 2016.

[57] Fabrizio Detassis, Michele Lombardi, and Michela Milano. Teaching the old dog new tricks: Supervised learning with constraints, 2021.

[58] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[59] Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Gated-attention readers for text comprehension. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1832–1846. Association for Computational Linguistics, 2017.

[60] Bhuwan Dhingra, Manzil Zaheer, Vidhisha Balachandran, Graham Neubig, Ruslan Salakhutdinov, and William W Cohen. Differentiable reasoning over a virtual knowledge base. In *International Conference on Learning Representations*, 2019.

[61] Michelangelo Diligenti, Marco Gori, and Claudio Saccà. Semantic-based regularization for learning and inference. *Artificial Intelligence*, 244:143–165, 2017. Combining Constraint Solving with Mining and Learning.

[62] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. Wizard of wikipedia: Knowledge-powered conversational agents. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[63] Ivan Donadello, Luciano Serafini, and Artur S. d'Avila Garcez. Logic tensor networks for semantic image interpretation. In Carles Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1596–1602. ijcai.org, 2017.

[64] Biao Dong, Haoze Yu, and Haisheng Li. A knowledge graph construction approach for legal domain. *Tehnicki Vjesnik-technical Gazette*, 28:357–362, 2021.

[65] Xiangyu Dong, Wenhao Yu, Chenguang Zhu, and Meng Jiang. Injecting entity types into entity-guided text generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 734–741, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[66] David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. Convolutional networks on graphs for learning molecular fingerprints. *CoRR*, abs/1509.09292, 2015.

[67] Pradheep Elango. Coreference resolution: A survey. *University of Wisconsin, Madison, WI*, 2005.

[68] Kevin Ellis, Lucas Morales, Mathias Sablé-Meyer, Armando Solar-Lezama, and Josh Tenenbaum. Learning libraries of subroutines for neurally–guided bayesian program induction. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[69] Herbert B. Enderton. *A mathematical introduction to logic*. Academic Press, 1972.

[70] Richard Evans and Edward Grefenstette. Learning explanatory rules from noisy data. *J. Artif. Intell. Res.*, 61:1–64, 2018.

[71] Angela Fan, Claire Gardent, Chloé Braud, and Antoine Bordes. Using local knowledge graph construction to scale seq2seq models to multi-document inputs. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4184–4194. Association for Computational Linguistics, 2019.

[72] Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard H. Hovy. A survey of data augmentation approaches for NLP. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 968–988. Association for Computational Linguistics, 2021.

[73] Jenny Rose Finkel and Christopher D. Manning. Joint parsing and named entity recognition. In *NAACL*, 2009.

[74] Ferdinando Fioretto, Pascal Van Hentenryck, Terrence W. K. Mak, Cuong Tran, Federico Baldo, and Michele Lombardi. Lagrangian duality for constrained deep learning. In *ECML PKDD*, 2020.

[75] Jennifer Foster, Özlem Çetinoğlu, Joachim Wagner, Joseph Le Roux, Joakim Nivre, Deirdre Hogan, and Josef van Genabith. From news to comment: Resources and benchmarks for parsing the language of web 2.0. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 893–901, Chiang Mai, Thailand, November 2011. Asian Federation of Natural Language Processing.

[76] Andrea Galassi, Marco Lippi, and Paolo Torroni. Argumentative link prediction using residual networks and multi-objective learning. In *ArgMining Workshop*, pages 1–10, November 2018.

[77] Andrea Galassi, Marco Lippi, and Paolo Torroni. Attention in natural language processing. *IEEE Transactions on Neural Networks and Learning Systems*, 32(10):4291–4308, Oct 2021.

[78] Andrea Galassi, Marco Lippi, and Paolo Torroni. Investigating logic tensor networks for neural-symbolic argument mining. 2021.

[79] Andrea Galassi, Marco Lippi, and Paolo Torroni. Multi-task attentive residual networks for argument mining. *CoRR*, abs/2102.12227, 2021.

[80] Marcus Galdia. *Lectures on Legal Linguistics*. Peter Lang Verlag, Berlin, Germany, 2017.

[81] Kuzman Ganchev, Joao Graca, Jennifer Gillenwater, and Ben Taskar. Posterior regularization for structured latent variable models. *Journal of Machine Learning Research*, 11(67):2001–2049, 2010.

[82] Artur d'Avila Garcez, Tarek R Besold, Luc De Raedt, Peter Földiak, Pascal Hitzler, Thomas Icard, Kai-Uwe Kühnberger, Luis C Lamb, Risto Miikkulainen, and Daniel L Silver. Neural-symbolic learning and reasoning: contributions and challenges. In *2015 AAAI Spring Symposium Series*, 2015.

[83] Ruiying Geng, Binhua Li, Yongbin Li, Jian Sun, and Xiaodan Zhu. Dynamic memory induction networks for few-shot text classification. In Dan Jurafsky, Joyce Chai, Natalie

Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 1087–1094. Association for Computational Linguistics, 2020.

[84] Mor Geva and Jonathan Berant. Learning to search in long documents using document structure. In *COLING*, 2018.

[85] Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. A knowledge-grounded neural conversation model. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[86] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. *CoRR*, abs/1704.01212, 2017.

[87] Kevin Gimpel, Nathan Schneider, Brendan T. O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *ACL*, 2011.

[88] Luca Giuliani. *Extending the Moving Targets Method for Injecting Constraints in Machine Learning*. PhD thesis.

[89] Goran Glavaš and Ivan Vulić. Explicit retrofitting of distributional word vectors. In *Proc. of the 56th Annual Meeting of the ACL*, pages 34–45, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[90] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N project report, Stanford*, 1(12):2009, 2009.

[91] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[92] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural Turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

[93] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *CoRR*, abs/1410.5401, 2014.

[94] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, (7626):471, 2016.

[95] Nancy Green. Identifying argumentation schemes in genetics research articles. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 12–21, 2015.

[96] Shai Gretz, Yonatan Bilu, Edo Cohen-Karlik, and Noam Slonim. The workweek is the best time to start a family – a study of GPT-2 based claim generation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 528–544, Online, November 2020. Association for Computational Linguistics.

[97] Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O. K. Li. Search engine guided non-parametric neural machine translation. *CoRR*, abs/1705.07267, 2017.

[98] Jian Guan, Fei Huang, Minlie Huang, Zhihao Zhao, and Xiaoyan Zhu. A knowledge-enhanced pretraining model for commonsense story generation. *Trans. Assoc. Comput. Linguistics*, 8:93–108, 2020.

[99] Jian Guan, Yansen Wang, and Minlie Huang. Story ending generation with incremental encoding and commonsense knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6473–6480, 2019.

[100] Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. Knowledge graph embedding with iterative guidance from soft rules. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[101] Kadri Hacioglu. Semantic role labeling using dependency trees. In *COLING*, 2004.

[102] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. *CoRR*, abs/1706.02216, 2017.

[103] Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[104] Braden Hancock, Paroma Varma, Stephanie Wang, Martin Bringmann, Percy Liang, and Christopher Ré. Training classifiers with natural language explanations. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 1884–1895. Association for Computational Linguistics, 2018.

[105] Yanchao Hao, Yuanzhe Zhang, Kang Liu, Shizhu He, Zhanyi Liu, Hua Wu, and Jun Zhao. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 221–231, 2017.

[106] Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. Tracking the world state with recurrent entity networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[107] Todd Hester, Matej Vecerík, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, Gabriel Dulac-Arnold, John P. Agapiou, Joel Z. Leibo, and Audrunas Gruslys. Deep q-learning from demonstrations. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 3223–3230. AAAI Press, 2018.

[108] Annerieke Heuvelink. Cognitive models for training simulations. *Unpublished doctoral dissertation, Vrije Universiteit Amsterdam, The Netherlands*, 2009.

[109] Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The goldilocks principle: Reading children's books with explicit memory representations. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[110] Julia Hirschberg and Christopher D. Manning. Advances in natural language processing. *Science*, 349(6245):261–266, 2015.

[111] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard H. Hovy, and Eric P. Xing. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, 2016.

[112] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. Toward controlled generation of text. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017,*

*Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1587–1596. PMLR, 2017.

[113] Zhiting Hu, Zichao Yang, Ruslan Salakhutdinov, and Eric Xing. Deep neural networks with massive learned knowledge. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1670–1679, 2016.

[114] Zhiting Hu, Zichao Yang, Russ R Salakhutdinov, LIANHUI Qin, Xiaodan Liang, Haoye Dong, and Eric P Xing. Deep generative models with learnable knowledge constraints. In *Advances in Neural Information Processing Systems*, pages 10501–10512, 2018.

[115] Ziniu Hu, Ting Chen, Kai-Wei Chang, and Yizhou Sun. Few-shot representation learning for out-of-vocabulary words. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4102–4112. Association for Computational Linguistics, 2019.

[116] Luyang Huang, Lingfei Wu, and Lu Wang. Knowledge graph-augmented abstractive summarization with semantic-driven cloze reward. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5094–5107. Association for Computational Linguistics, 2020.

[117] Sheikh Rabiul Islam, William Eberle, Sheikh Khaled Ghafoor, and Mohiuddin Ahmed. Explainable artificial intelligence approaches: A survey. *CoRR*, abs/2101.09429, 2021.

[118] Di Jin, Zhijing Jin, Zhiting Hu, Olga Vechtomova, and Rada Mihalcea. Deep learning for text style transfer: A survey. *CoRR*, abs/2011.00416, 2020.

[119] Ashwin Kalyan, Abhishek Mohta, Oleksandr Polozov, Dhruv Batra, Prateek Jain, and Sumit Gulwani. Neural-guided deductive search for real-time program synthesis from examples. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[120] Russell Kaplan, Christopher Sauer, and Alexander Sosa. Beating atari with natural language guided reinforcement learning. *CoRR*, abs/1704.05539, 2017.

[121] Andrej Karpathy, Justin Johnson, and Li Fei-Fei. Visualizing and understanding recurrent networks. *CoRR*, abs/1506.02078, 2015.

[122] Christel Kemke. Natural language communication between human and artificial agents. In Zhong-Zhi Shi and Ramakoti Sadananda, editors, *Agent Computing and Multi-Agent Systems*, pages 84–93, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[123] Vlado Keselj. Speech and language processing (second edition) daniel jurafsky and james h. martin (stanford university and university of colorado at boulder) pearson prentice hall, 2009, hardbound, ISBN 978-0-13-187321-6. *Comput. Linguistics*, 35(3):463–466, 2009.

[124] Wai-Howe Khong, Lay-Ki Soon, Hui-Ngo Goh, and Su-Cheng Haw. Leveraging part-of-speech tagging for sentiment analysis in short texts and regular texts. In *Joint International Semantic Technology Conference*, pages 182–197. Springer, 2018.

[125] Tushar Khot, Sriraam Natarajan, Kristian Kersting, and Jude Shavlik. Learning markov logic networks via functional gradient boosting. In *2011 IEEE 11th International Conference on Data Mining*, pages 320–329, 2011.

[126] Donghyeon Kim, Jinhyuk Lee, Chan Ho So, Hwisang Jeon, Minbyul Jeong, Yonghwa Choi, Wonjin Yoon, Mujeen Sung, and Jaewoo Kang. A neural named entity recognition and multi-type normalization tool for biomedical text mining. *IEEE Access*, 7:73729–73740, 2019.

[127] Yoon Kim, Alexander M. Rush, Lei Yu, Adhiguna Kuncoro, Chris Dyer, and Gábor Melis. Unsupervised recurrent neural network grammars. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1105–1117. Association for Computational Linguistics, 2019.

[128] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, 2016.

[129] Christian Kirschner, Judith Eckle-Kohler, and Iryna Gurevych. Linking the thoughts: Analysis of argumentation structures in scientific publications. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 1–11, 2015.

[130] Alexandros Komninos and Suresh Manandhar. Dependency based embeddings for sentence classification tasks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language*

*Technologies*, pages 1490–1500, San Diego, California, June 2016. Association for Computational Linguistics.

[131] Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. Text generation from knowledge graphs with graph transformers. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2284–2293. Association for Computational Linguistics, 2019.

[132] Efstratios Kontopoulos, Christos Berberidis, Theologos Dergiades, and Nick Bassiliades. Ontology-based sentiment analysis of twitter posts. *Expert systems with applications*, 40(10):4065–4074, 2013.

[133] Kalpesh Krishna, Aurko Roy, and Mohit Iyyer. Hurdles to progress in long-form question answering. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tür, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 4940–4957. Association for Computational Linguistics, 2021.

[134] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. In *International conference on machine learning*, pages 1378–1387, 2016.

[135] Varun Kumar, Hadrien Glaude, Cyprien de Lichy, and William Campbell. A closer look at feature space data augmentation for few-shot intent classification. In Colin Cherry, Greg Durrett, George F. Foster, Reza Haffari, Shahram Khadivi, Nanyun Peng, Xiang Ren, and Swabha Swayamdipta, editors, *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP, DeepLo@EMNLP-IJCNLP 2019, Hong Kong, China, November 3, 2019*, pages 1–10. Association for Computational Linguistics, 2019.

[136] Francesca Lagioia, Federico Ruggeri, Kasper Drazewski, Marco Lippi, Hans-Wolfgang Micklitz, Paolo Torroni, and Giovanni Sartor. Deep learning for detecting and explaining unfairness in consumer contracts. In *Legal Knowledge and Information Systems: JURIX*

*2019: The Thirty-second Annual Conference*, volume 322 of *Frontiers in Artificial Intelligence and Applications*, pages 43–52. IOS Press, 2019.

[137] Christoph H. Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 951–958, 2009.

[138] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. In Kevin Knight, Ani Nenkova, and Owen Rambow, editors, *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 260–270. The Association for Computational Linguistics, 2016.

[139] Fabien Lauer and Gérard Bloch. Incorporating prior knowledge in support vector machines for classification: A review. *Neurocomputing*, 71(7-9):1578–1594, 2008.

[140] Anne Lauscher, Goran Glavaš, and Kai Eckert. ArguminSci: A tool for analyzing argumentation and rhetorical aspects in scientific writing. In *Proceedings of the 5th Workshop on Argument Mining*, pages 22–28, Brussels, Belgium, November 2018. Association for Computational Linguistics.

[141] Anne Lauscher, Goran Glavaš, and Simone Paolo Ponzetto. An argument-annotated corpus of scientific publications. In *Proceedings of the 5th Workshop on Argument Mining*, pages 40–46, Brussels, Belgium, November 2018. Association for Computational Linguistics.

[142] John Lawrence and Chris Reed. Argument mining: A survey. *Computational Linguistics*, 45(4):765–818, 2020.

[143] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021.

[144] Haoran Li, Junnan Zhu, Jiajun Zhang, Chengqing Zong, and Xiaodong He. Keywords-guided abstractive sentence summarization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8196–8203, Apr. 2020.

[145] J. Li, Aixin Sun, Jianglei Han, and Chenliang Li. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34:50–70, 2022.

[146] Juan Li, Ruoxu Wang, Ningyu Zhang, Wen Zhang, Fan Yang, and Huajun Chen. Logic-guided semantic representation learning for zero-shot relation classification. In Donia Scott, Núria Bel, and Chengqing Zong, editors, *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 2967–2978. International Committee on Computational Linguistics, 2020.

[147] Junyi Li, Tianyi Tang, Wayne Xin Zhao, and Ji-Rong Wen. Pretrained language models for text generation: A survey. *CoRR*, abs/2105.10311, 2021.

[148] Sujian Li, Liang Wang, Ziqiang Cao, and Wenjie Li. Text-level discourse dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pages 25–35. The Association for Computer Linguistics, 2014.

[149] Yujia Li, Chenjie Gu, Thomas Dullien, Oriol Vinyals, and Pushmeet Kohli. Graph matching networks for learning the similarity of graph structured objects. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 3835–3845. PMLR, 2019.

[150] Ruta Liepina, Federico Ruggeri, Francesca Lagioia, Marco Lippi, Kasper Drazewski, and Paolo Torroni. Explaining potentially unfair clauses to the consumer with the claudette tool. In *NLLP@KDD*, 2020.

[151] Yih-Kai Lin, Chu-Fu Wang, Ching-Yu Chang, and Hao-Lun Sun. An efficient framework for counting pedestrians crossing a line using low-cost devices: the benefits of distilling the knowledge in a neural network. *Multim. Tools Appl.*, 80(3):4037–4051, 2021.

[152] Marco Lippi and Paolo Frasconi. Prediction of protein $\beta$-residue contacts by Markov logic networks with grounding-specific weights. *Bioinformatics*, 25(18):2326–2333, 07 2009.

[153] Marco Lippi, Francesca Lagioia, Giuseppe Contissa, Giovanni Sartor, and Paolo Torroni. Claim detection in judgments of the eu court of justice. In *AI Approaches to the Complexity of Legal Systems*, pages 513–527. Springer, 2015.

[154] Marco Lippi, Przemysław Pałka, Giuseppe Contissa, Francesca Lagioia, Hans-Wolfgang Micklitz, Giovanni Sartor, and Paolo Torroni. CLAUDETTE: An automated detector of potentially unfair clauses in online terms of service. *Artificial Intelligence and Law*, 27(2):117–139, 2019.

[155] Marco Lippi and Paolo Torroni. Context-independent claim detection for argument mining. In Qiang Yang and Michael J. Wooldridge, editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 185–191. AAAI Press, 2015.

[156] Marco Lippi and Paolo Torroni. Argumentation mining: State of the art and emerging trends. *ACM Transactions on Internet Technology (TOIT)*, 16(2):1–25, 2016.

[157] Marco Lippi and Paolo Torroni. Margot: A web server for argumentation mining. *Expert Systems with Applications*, 65:292–303, 2016.

[158] Liu Liu, Kaile Liu, Zhenghai Cong, Jiali Zhao, Yefei Ji, and Jun He. Long length document classification by local convolutional feature aggregation. *Algorithms*, 11(8):109, 2018.

[159] Marco Loos and Joasia Luzak. Wanted: a bigger stick. on unfair terms in consumer contracts with online service providers. *Journal of consumer policy*, 39(1):63–90, 2016.

[160] Annie Louis. Natural Language Processing for Social Media. *Computational Linguistics*, 42(4):833–836, 12 2016.

[161] Anastasios Lytos, Thomas Lagkas, Panagiotis Sarigiannidis, and Kalina Bontcheva. The evolution of argumentation mining: From models to social media and emerging tools. *Information Processing & Management*, 56(6):102055, 2019.

[162] Tianle Ma and Aidong Zhang. Multi-view factorization autoencoder with network constraints for multi-omic integrative analysis. In *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 702–707. IEEE, 2018.

[163] Yukun Ma, Haiyun Peng, and Erik Cambria. Targeted aspect-based sentiment analysis via embedding commonsense knowledge into an attentive lstm. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[164] Jean Maillard, Stephen Clark, and Dani Yogatama. Jointly learning sentence embeddings and syntax with unsupervised tree-lstms. *Natural Language Engineering*, 25:433 – 449, 2019.

[165] Massimiliano Mancini, Jose Camacho-Collados, Ignacio Iacobacci, and Roberto Navigli. Embedding words and senses together via joint knowledge-enhanced training. *arXiv preprint arXiv:1612.02703*, 2016.

[166] Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Neural probabilistic logic programming in deepproblog. *Artif. Intell.*, 298:103504, 2021.

[167] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[168] Alex Marin, Roman Holenstein, Ruhi Sarikaya, and Mari Ostendorf. Learning phrase patterns for text classification using a knowledge graph and unlabeled data. In *INTERSPEECH*, 2014.

[169] Giuseppe Marra, Michelangelo Diligenti, Francesco Giannini, Marco Gori, and Marco Maggini. Relational neural machines. In Giuseppe De Giacomo, Alejandro Catalá, Bistra Dilkina, Michela Milano, Senén Barro, Alberto Bugarín, and Jérôme Lang, editors, *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain, August 29 - September 8, 2020 - Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 1340–1347. IOS Press, 2020.

[170] Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, Marco Maggini, and Marco Gori. Learning and t-norms theory. *CoRR*, abs/1907.11468, 2019.

[171] Giuseppe Marra and Ondrej Kuzelka. Neural markov logic networks. In Cassio P. de Campos, Marloes H. Maathuis, and Erik Quaeghebeur, editors, *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence, UAI 2021, Virtual Event, 27-30 July 2021*, volume 161 of *Proceedings of Machine Learning Research*, pages 908–917. AUAI Press, 2021.

[172] Tobias Mayer, Elena Cabrio, Marco Lippi, Paolo Torroni, and S. Villata. Argument mining on clinical trials. In *COMMA*, 2018.

[173] Tobias Mayer, Elena Cabrio, and S. Villata. Transformer-based argument mining for healthcare applications. In *ECAI*, 2020.

[174] Tobias Mayer, Santiago Marro, Elena Cabrio, and Serena Villata. Enhancing evidence-based medicine with natural language argumentative analysis of clinical trials. *Artificial Intelligence in Medicine*, 118:102098, 2021.

[175] Chuan Meng, Pengjie Ren, Zhumin Chen, Christof Monz, Jun Ma, and M. de Rijke. Refnet: A reference-aware network for background based conversation. In *AAAI*, 2020.

[176] Hans-W Micklitz, Przemysław Pałka, and Yannis Panagis. The empire strikes back: digital control of unfair terms of online services. *Journal of consumer policy*, 40(3):367–388, 2017.

[177] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013.

[178] A. H. Miller, W. Feng, A. Fisch, J. Lu, D. Batra, A. Bordes, D. Parikh, and J. Weston. Parlai: A dialog research software platform. *arXiv preprint arXiv:1705.06476*, 2017.

[179] Alexander H. Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. Key-value memory networks for directly reading documents. In Jian Su, Xavier Carreras, and Kevin Duh, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1400–1409. The Association for Computational Linguistics, 2016.

[180] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[181] Marvin L Minsky. Logical versus analogical or symbolic versus connectionist or neat versus scruffy. *AI magazine*, 12(2):34–34, 1991.

[182] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics, 2009.

[183] Eleonora Misino. *Deep Generative Models with Probabilistic Logic Priors*. PhD thesis.

[184] Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhanava Dalvi, Matt Gardner, Bryan Kisiel, et al. Never-ending learning. *Communications of the ACM*, 61(5):103–115, 2018.

[185] Julian Moreno-Schneider, Georg Rehm, Elena Montiel-Ponsoda, Víctor Rodríguez-Doncel, Artem Revenko, Sotirios Karampatakis, Maria Khvalchik, Christian Sageder, Jorge Gracia, and Filippo Maganza. Orchestrating nlp services for the legal domain. In *LREC*, 2020.

[186] Alessandro Moschitti. Making tree kernels practical for natural language learning. In *11th conference of the European Chapter of the Association for Computational Linguistics*, 2006.

[187] Alessandro Moschitti and Roberto Basili. Complex linguistic features for text classification: A comprehensive study. In *European conference on information retrieval*, pages 181–196. Springer, 2004.

[188] Nikola Mrksic, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gasic, Lina Maria Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve J. Young. Counter-fitting word vectors to linguistic constraints. In Kevin Knight, Ani Nenkova, and Owen Rambow, editors, *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 142–148. The Association for Computational Linguistics, 2016.

[189] Tsendsuren Munkhdalai, Alessandro Sordoni, TONG WANG, and Adam Trischler. Metalearned neural memory. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[190] Tsendsuren Munkhdalai, Alessandro Sordoni, Tong Wang, and Adam Trischler. Metalearned neural memory. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13310–13321, 2019.

[191] Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çaglar Gülçehre, and Bing Xiang. Abstractive text summarization using sequence-to-sequence rnns and beyond. In Yoav Goldberg and Stefan Riezler, editors, *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 280–290. ACL, 2016.

[192] Karthik Narasimhan, Regina Barzilay, and Tommi S. Jaakkola. Grounding language for transfer in deep reinforcement learning. *J. Artif. Intell. Res.*, 63:849–874, 2018.

[193] Roberto Navigli. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):1–69, 2009.

[194] Roberto Navigli, Mirella Lapata, et al. Graph connectivity measures for unsupervised word sense disambiguation. In *IJCAI*, volume 7, pages 1683–1688, 2007.

[195] Quoc Viet Nguyen, Chi Thang Duong, Thanh Tam Nguyen, Matthias Weidlich, Karl Aberer, Hongzhi Yin, and Xiaofang Zhou. Argument discovery via crowdsourcing. *The VLDB Journal*, 26(4):511–535, aug 2017.

[196] Jinjie Ni, Tom Young, Vlad Pandelea, Fuzhao Xue, Vinay Adiga, and Erik Cambria. Recent advances in deep learning based dialogue systems: A systematic survey. *CoRR*, abs/2105.04387, 2021.

[197] A. Opdahl and Vimala Nunavath. Big data in emergency management: Exploitation techniques for social and mobile data. *Big Data in Emergency Management: Exploitation Techniques for Social and Mobile Data*, 2020.

[198] Juri Opitz, Philipp Heinisch, Philipp Wiesenbach, Philipp Cimiano, and Anette Frank. Explainable unsupervised argument similarity rating with Abstract Meaning Representation and conclusion generation. In *Proceedings of the 8th Workshop on Argument Mining*, pages 24–35, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[199] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[200] Prasanna Parthasarathi and Joelle Pineau. Extending neural generative conversational model using external knowledge sources. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 690–695. Association for Computational Linguistics, 2018.

[201] Nikita D Patel and Chetana Chand. Selecting best features using combined approach in pos tagging for sentiment analysis. *International Journal of Computer Science and Mobile Computing*, 3(3):425–430, 2014.

[202] Juan Pavez, Héctor Allende, and Héctor Allende-Cid. Working memory networks: Augmenting memory networks with a relational reasoning module. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 1000–1009. Association for Computational Linguistics, 2018.

[203] Baolin Peng, Chenguang Zhu, Chunyuan Li, Xiujun Li, Jinchao Li, Michael Zeng, and Jianfeng Gao. Few-shot natural language generation for task-oriented dialog. In Trevor Cohn, Yulan He, and Yang Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 172–182. Association for Computational Linguistics, 2020.

[204] Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544, Online, June 2021. Association for Computational Linguistics.

[205] Barbara Plank, Anders Søgaard, and Yoav Goldberg. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*. The Association for Computer Linguistics, 2016.

[206] Edoardo Maria Ponti, Ivan Vulic, Ryan Cotterell, Roi Reichart, and Anna Korhonen. Towards zero-shot language modeling. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2900–2910. Association for Computational Linguistics, 2019.

[207] Hoifung Poon and Pedro M. Domingos. Sum-product networks: A new deep architecture. In Fábio Gagliardi Cozman and Avi Pfeffer, editors, *UAI 2011, Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence, Barcelona, Spain, July 14-17, 2011*, pages 337–346. AUAI Press, 2011.

[208] Hoifung Poon, Pedro M. Domingos, and Marc Sumner. A general method for reducing the complexity of relational inference and its application to mcmc. In *AAAI*, 2008.

[209] Lianhui Qin, Michel Galley, Chris Brockett, Xiaodong Liu, Xiang Gao, Bill Dolan, Yejin Choi, and Jianfeng Gao. Conversing by reading: Contentful neural conversation with on-demand machine reading. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5427–5436. Association for Computational Linguistics, 2019.

[210] Pengda Qin, Xin Wang, Wenhu Chen, Chunyun Zhang, Weiran Xu, and William Yang Wang. Generative adversarial zero-shot relational learning for knowledge graphs. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8673–8680. AAAI Press, 2020.

[211] Luc De Raedt, Sebastijan Dumancic, Robin Manhaeve, and Giuseppe Marra. From statistical relational to neuro-symbolic artificial intelligence. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 4943–4950. ijcai.org, 2020.

[212] Adwait Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In Eric Brill and Kenneth Church, editors, *Conference on Empirical Methods in Natural Language Processing, EMNLP 1996, Philadelphia, PA, USA, May 17-18, 1996*, 1996.

[213] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *CoRR*, abs/1908.10084, 2019.

[214] Nils Reimers, Benjamin Schiller, Tilman Beck, Johannes Daxenberger, Christian Stab, and Iryna Gurevych. Classification and clustering of arguments with contextualized word embeddings. In *ACL*, pages 567–578, Florence, Italy, 2019.

[215] Ridho Reinanda, Edgar Meij, and Maarten de Rijke. Knowledge graphs: An information retrieval perspective. *Foundations and Trends® in Information Retrieval*, 14(4):289–444, 2020.

[216] Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine learning*, 62(1-2):107–136, 2006.

[217] Ruty Rinott, Lena Dankin, Carlos Alzate, Mitesh M Khapra, Ehud Aharoni, and Noam Slonim. Show me your evidence-an automatic method for context dependent evidence detection. In *Proceedings of the 2015 EMNLP Conference*, pages 440–450, 2015.

[218] Anthony Rios and Ramakanth Kavuluru. Few-shot and zero-shot multi-label learning for structured label spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3132–3142, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[219] Tim Rocktäschel, Matko Bosnjak, Sameer Singh, and Sebastian Riedel. Low-dimensional embeddings of logic. In *Proceedings of the ACL 2014 Workshop on Semantic Parsing*, pages 45–49, 2014.

[220] Tim Rocktäschel and Sebastian Riedel. End-to-end differentiable proving. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3788–3800, 2017.

[221] Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. Injecting logical background knowledge into embeddings for relation extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1119–1129, 2015.

[222] Ribana Roscher, Bastian Bohn, Marco F. Duarte, and Jochen Garcke. Explainable machine learning for scientific insights and discoveries. *IEEE Access*, 8:42200–42216, 2020.

[223] Federico Ruggeri, Francesca Lagioia, Marco Lippi, and Paolo Torroni. Detecting and explaining unfairness in consumer contracts through memory networks. *Art. Int. and Law*, pages 1–34, 2021.

[224] Federico Ruggeri, Marco Lippi, and Paolo Torroni. Membert: Injecting unstructured knowledge into BERT. *CoRR*, abs/2110.00125, 2021.

[225] Federico Ruggeri, Marco Lippi, and Paolo Torroni. Tree-constrained graph neural networks for argument mining. *CoRR*, abs/2110.00124, 2021.

[226] Federico Ruggeri, Mohsen Mesgar, and Iryna Gurevych. Argscichat: A dataset for argumentative dialogues on scientific papers, 2022.

[227] Erik Tjong Kim Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *CoNLL*, 2003.

[228] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019.

[229] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[230] Robin Schaefer and Manfred Stede. Argument mining on twitter: A survey. *it - Information Technology*, 63:45 – 58, 2021.

[231] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *4th Int. Conf. on Learning Representations, ICLR 2016*, 2016.

[232] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer, 2018.

[233] Helmut Schmidt. Probabilistic part-of-speech tagging using decision trees. 1994.

[234] Sergej Schmunk, Wolfram Höpken, Matthias Fuchs, and Maria Lexhagen. Sentiment analysis: Extracting decision-relevant knowledge from ugc. In *Information and Communication Technologies in Tourism 2014*, pages 253–265. Springer, 2013.

[235] Sanket Shah, Anand Mishra, Naganand Yadati, and Partha Pratim Talukdar. Kvqa: Knowledge-aware visual question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8876–8884, 2019.

[236] Walid Shalaby, Wlodek Zadrozny, and Hongxia Jin. Beyond word embeddings: learning entity and concept representations from large scale knowledge bases. *Information Retrieval Journal*, 22(6):525–542, 2019.

[237] Amr Sharaf, Hany Hassan, and Hal Daumé III. Meta-learning for few-shot NMT adaptation. In *Proceedings of the Fourth Workshop on Neural Generation and Translation*, pages 43–53, Online, July 2020. Association for Computational Linguistics.

[238] Pratyusha Sharma, Antonio Torralba, and Jacob Andreas. Skill induction and planning with latent language. *CoRR*, abs/2110.01517, 2021.

[239] Baoxu Shi and Tim Weninger. Open-world knowledge graph completion. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1957–1964. AAAI Press, 2018.

[240] Sifatullah Siddiqi and Aditi Sharan. Keyword and keyphrase extraction techniques: A literature review. *International Journal of Computer Applications*, 109:18–23, 2015.

[241] Noam Slonim, Yonatan Bilu, Carlos Alzate, Roy Bar-Haim, Ben Bogin, Francesca Bonin, Leshem Choshen, Edo Cohen-Karlik, Lena Dankin, Lilach Edelstein, et al. An autonomous debating system. *Nature*, 591(7850):379–384, 2021.

[242] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[243] Haoyu Song, Weinan Zhang, Yiming Cui, Dong Wang, and Ting Liu. Exploiting persona information for diverse generation of conversational responses. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5190–5196. ijcai.org, 2019.

[244] Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. A graph-to-sequence model for amr-to-text generation. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 1616–1626. Association for Computational Linguistics, 2018.

[245] Robert Speer and Catherine Havasi. Conceptnet 5: A large semantic network for relational knowledge. In *The People's Web Meets NLP*, pages 161–176. Springer, 2013.

[246] Alessandro Sperduti and Antonina Starita. Supervised neural networks for the classification of structures. *IEEE Transactions on Neural Networks*, 8(3):714–735, 1997.

[247] Christian Stab and Iryna Gurevych. Parsing argumentation structures in persuasive essays. *Computational Linguistics*, 43(3):619–659, 2017.

[248] Christian Stab, Tristan Miller, Benjamin Schiller, Pranav Rai, and Iryna Gurevych. Cross-topic argument mining from heterogeneous sources. In *EMNLP*, pages 3664–3674, October-November 2018.

[249] Shane Storks, Qiaozi Gao, and Joyce Y. Chai. Commonsense reasoning for natural language understanding: A survey of benchmarks, resources, and approaches. *CoRR*, abs/1904.01172, 2019.

[250] Shane Storks, Qiaozi Gao, and Joyce Y. Chai. Recent advances in natural language inference: A survey of benchmarks, resources, and approaches, 2020.

[251] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *Advances in NeurIPS 2015 Conference, Montreal, Canada*, pages 2440–2448, 2015.

[252] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448, 2015.

[253] Rhea Sukthanker, Soujanya Poria, Erik Cambria, and Ramkumar Thirunavukarasu. Anaphora and coreference resolution: A review. *Information Fusion*, 59:139–162, 2020.

[254] Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W Cohen. Open domain question answering using early fusion of knowledge bases and text. *arXiv preprint arXiv:1809.00782*, 2018.

[255] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. Improved semantic representations from tree-structured long short-term memory networks. In *ACL 2015, Beijing, China*, pages 1556–1566, 2015.

[256] Kar-Han Tan and Boon Pang Lim. The artificial intelligence renaissance: deep learning and the road to human-level machine intelligence. *APSIPA Transactions on Signal and Information Processing*, 7:e6, 2018.

[257] Duyu Tang, Bing Qin, and Ting Liu. Aspect level sentiment classification with deep memory network. In Jian Su, Xavier Carreras, and Kevin Duh, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 214–224. The Association for Computational Linguistics, 2016.

[258] Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1555–1565, 2014.

[259] Ann Taylor, Mitchell Marcus, and Beatrice Santorini. *The Penn Treebank: An Overview*, pages 5–22. Springer Netherlands, Dordrecht, 2003.

[260] Kentaro Torisawa et al. Exploiting wikipedia as external knowledge for named entity recognition. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 698–707, 2007.

[261] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *NAACL*, 2003.

[262] Geoffrey G Towell and Jude W Shavlik. Knowledge-based artificial neural networks. *Artificial intelligence*, 70(1-2):119–165, 1994.

[263] Talip Ucar, Adrian Gonzalez-Martin, Matthew Lee, and Adrian Daniel Szwarc. One-shot learning for language modelling. *CoRR*, abs/2007.09679, 2020.

[264] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[265] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[266] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2692–2700, 2015.

[267] Laura von Rüden, Sebastian Mayer, Jochen Garcke, Christian Bauckhage, and Jannis Schücker. Informed machine learning - towards a taxonomy of explicit integration of knowledge into machine learning. *CoRR*, abs/1903.12394, 2019.

[268] Douglas Walton. *Informal Logic: A Pragmatic Approach*. Cambridge University Press, 2 edition, 2008.

[269] Jin Wang, Zhongyuan Wang, Dawei Zhang, and Jun Yan. Combining knowledge with deep convolutional neural networks for short text classification. In *IJCAI*, pages 2915–2921, 2017.

[270] Kai Wang, Xiaojun Quan, and Rui Wang. BiSET: Bi-directional selective encoding with template for abstractive summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2153–2162, Florence, Italy, July 2019. Association for Computational Linguistics.

[271] Tianming Wang, Xiaojun Wan, and Hanqi Jin. Amr-to-text generation with graph transformer. *Transactions of the Association for Computational Linguistics*, 8:19–33, 2020.

[272] William Yang Wang, Kathryn Mazaitis, and William W Cohen. Structure learning via parameter learning. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 1199–1208, 2014.

[273] Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Comput. Surv.*, 53(3):63:1–63:34, 2020.

[274] Dirk Weissenborn, Tomáš Kočiskỳ, and Chris Dyer. Dynamic integration of background knowledge in neural nlu systems. *arXiv preprint arXiv:1706.02596*, 2017.

[275] Orion Weller, Nicholas Lourie, Matt Gardner, and Matthew E. Peters. Learning from task descriptions. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1361–1375. Association for Computational Linguistics, 2020.

[276] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[277] Catherine Wong, Kevin Ellis, Joshua B. Tenenbaum, and Jacob Andreas. Leveraging language to learn program abstractions and search heuristics. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning,*

*ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 11193–11204. PMLR, 2021.

[278] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[279] Congying Xia, Chenwei Zhang, Xiaohui Yan, Yi Chang, and Philip S. Yu. Zero-shot user intent detection via capsule neural networks. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3090–3099. Association for Computational Linguistics, 2018.

[280] Congying Xia, Chenwei Zhang, Jiawei Zhang, Tingting Liang, Hao Peng, and Philip S. Yu. Low-shot learning in natural language processing. In *2020 IEEE Second International Conference on Cognitive Machine Intelligence (CogMI)*, pages 185–189, 2020.

[281] Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning - A comprehensive evaluation of the good, the bad and the ugly. *CoRR*, abs/1707.00600, 2017.

[282] Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic memory networks for visual and textual question answering. In *International conference on machine learning*, pages 2397–2406, 2016.

[283] Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Van den Broeck. A semantic loss function for deep learning with symbolic knowledge. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 5498–5507. PMLR, 2018.

[284] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[285] Vikas Yadav and Steven Bethard. A survey on recent advances in named entity recognition from deep learning models. In Emily M. Bender, Leon Derczynski, and Pierre Isabelle, editors, *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 2145–2158. Association for Computational Linguistics, 2018.

[286] Jie Yang, Soyeon Caren Han, and Josiah Poon. A survey on extraction of causal relations from natural language text. *CoRR*, abs/2101.06426, 2021.

[287] Xuefeng Yang and Kezhi Mao. Learning multi-prototype word embedding from single-prototype word embedding with integrated knowledge. *Expert Systems with Applications*, 56:291–299, 2016.

[288] Yuan Yang and Le Song. Learn to explain efficiently via neural logic inductive learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[289] Liang Yao, Chengsheng Mao, and Yuan Luo. Graph convolutional networks for text classification. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7370–7377. AAAI Press, 2019.

[290] Zhi-Xiu Ye and Zhen-Hua Ling. Distant supervision relation extraction with intra-bag and inter-bag attentions. In *Proceedings of the 2019 NAACL-HLT Conference*, pages 2810–2819, Minneapolis, Minnesota, June 2019. ACL.

[291] Wenpeng Yin. Meta-learning for few-shot natural language processing: A survey. *CoRR*, abs/2007.09604, 2020.

[292] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In *NeurIPS 2018, Montréal, Canada*, 2018.

[293] Dongran Yu, Bo Yang, Dayou Liu, and Hui Wang. A survey on neural-symbolic systems. *CoRR*, abs/2111.08164, 2021.

[294] Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. Diverse few-shot text classification with multiple metrics. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1206–1215. Association for Computational Linguistics, 2018.

[295] Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang. A survey of knowledge-enhanced text generation. *CoRR*, abs/2010.04389, 2020.

[296] Yue Yu, Jie Chen, Tian Gao, and Mo Yu. DAG-GNN: DAG structure learning with graph neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 7154–7163. PMLR, 2019.

[297] Sarah Zelikovitz and Haym Hirsh. Integrating background knowledge into text classification. In *IJCAI*, pages 1448–1449, 2003.

[298] Jian Zhang, Liangyou Li, Andy Way, and Qun Liu. Topic-informed neural machine translation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1807–1817, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee.

[299] Jingqing Zhang, Piyawat Lertvittayakumjorn, and Yike Guo. Integrating semantic knowledge to tackle zero-shot text classification. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1031–1040. Association for Computational Linguistics, 2019.

[300] Rui Zhang, Dimitar Hristovski, Dalton Schutte, Andrej Kastrin, Marcelo Fiszman, and Halil Kilicoglu. Drug repurposing for covid-19 via knowledge graph completion. *Journal of Biomedical Informatics*, 115:103696, 2021.

[301] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. Personalizing dialogue agents: I have a dog, do you have pets too? In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2204–2213. Association for Computational Linguistics, 2018.

[302] Tao Zhang, Congying Xia, Chun-Ta Lu, and Philip Yu. MZET: Memory augmented zero-shot fine-grained named entity typing. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 77–87, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[303] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. ERNIE: enhanced language representation with informative entities. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1441–1451. Association for Computational Linguistics, 2019.

[304] Haoxi Zhong, Chaojun Xiao, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. How does NLP benefit legal system: A summary of legal artificial intelligence. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5218–5230. Association for Computational Linguistics, 2020.

[305] Victor Zhong, Tim Rocktäschel, and Edward Grefenstette. RTFM: generalising to novel environment dynamics via reading. *CoRR*, abs/1910.08210, 2019.

[306] Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. Commonsense knowledge aware conversation generation with graph attention. In *IJCAI*, pages 4623–4629, 2018.

[307] Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. Neural question generation from text: A preliminary study. In Xuanjing Huang, Jing Jiang, Dongyan Zhao, Yansong Feng, and Yu Hong, editors, *Natural Language Processing and Chinese Computing - 6th CCF International Conference, NLPCC 2017, Dalian, China, November 8-12, 2017, Proceedings*, volume 10619 of *Lecture Notes in Computer Science*, pages 662–671. Springer, 2017.