Semantically-oriented Text Planning for Automatic Summarization

Gerard Casamayor

TESI DOCTORAL UPF / ANY 2021

DIRECTOR DE LA TESI

Leo Wanner Departament Departament de Tecnologies de la Informació i les Comunicacions

upf. Universitat Pompeu Fabra Barcelona

Abstract

Text summarization deals with the automatic creation of summaries from one or more documents, either by extracting fragments from the input text or by generating an abstract de novo. Research in recent years has become dominated by a new paradigm where summarization is addressed as a mapping from a sequence of tokens in an input document to a new sequence of tokens summarizing the input. Works following this paradigm apply supervised deep learning methods to learn sequence to sequence models from a large corpus of documents paired with human-crafted summaries. Despite impressive results in automatic quantitative evaluations, this approach to summarization also suffers from a number of drawbacks.

One concern is that learned models tend to operate in a black-box fashion that prevents obtaining insights or results from intermediate analysis that could be applied to other tasks -an important consideration in many real-world scenarios where summaries are not the only desired output of a natural language processing system. Another significant drawback is that deep learning methods are largely constrained to languages and types of summary for which abundant corpora containing human authored summaries is available. Albeit researchers are experimenting with transfer learning methods to overcome this problem, it is far from clear how effective these methods are and how to apply them to scenarios where summaries need to adapt to a query or to user preferences.

In those cases where it is not practical to learn a sequence to sequence model, it is convenient to fall back to a more traditional formulation of summarization where the input documents are first analyzed, then a summary is planned by selecting and organizing contents, and the final summary is generated either extractively or abstractively –using natural language generation methods in the latter case. By separating linguistic analysis, planning and generation, it becomes possible to apply different approaches to each task. This thesis focuses on the text planning step.

Drawing from past research in word sense disambiguation, text summarization and natural language generation, this thesis presents an unsupervised approach to planning the production of summaries. Following the observation that a common strategy for both disambiguation and summarization tasks is to rank candidate items –meanings, text fragments– we propose a strategy, at the core of our approach, that ranks candidate lexical meanings and individual words in a text. These ranks contribute towards the creation of a graph-based semantic representation from which we select non-redundant contents and organize them for inclusion in the summary. The overall approach is supported by lexicographic databases that provide cross-lingual and cross-domain knowledge, and by textual similarity methods used to compare meanings with each other and with the text.

The methods presented in this thesis are tested on two separate tasks, disambiguation of word senses and named entities, and single-document extractive summarization of English texts. The evaluation of the disambiguation task shows that our approach produces useful results for tasks other than summarization, while evaluating in an extractive summarization setting allows us to compare our approach to existing summarization systems. While the results are inconclusive with respect to state-of-the-art in disambiguation and summarization systems, they hint at a large potential for our approach.

Resum

El resum automàtic de textos és una tasca dins del camp d’estudi de processament del llenguatge natural que versa sobre la creació automàtica de resums d’un o més documents, ja sigui extraient fragments del text d’entrada or generant un resum des de zero. La recerca recent en aquesta tasca ha estat dominada per un nou paradigma on el resum és abordat com un mapeig d’una seqüència de paraules en el document d’entrada a una nova seqüència de paraules que resumeixen el document. Els treballs que segueixen aquest paradigma apliquen mètodes d’aprenentatge supervisat profund per tal d’aprendre model seqüència a seqüència a partir d’un gran corpus de documents emparellats amb resums escrits a mà. Tot i els resultats impressionants en avaluacions quantitatives automàtiques, aquesta aproximació al resum automàtic també té alguns inconvenients.

Un primer problema és que els models entrenats tendeixen a operar com una caixa negra que impedeix obtenir coneixements o resultats de representacions intermèdies i que puguin ser aplicat a altres tasques. Aquest és un problema important en situacions del món real on els resums no son l’única sortida que s’espera d’un sistema de processament de llenguatge natural. Un altre inconvenient significatiu és que els mètodes d’aprenentatge profund estan limitats a idiomes i tipus de resum pels que existeixen grans corpus amb resums escrits per humans. Tot i que els investigadors experimenten amb mètodes de transferència del coneixement per a superar aquest problema, encara ens trobem lluny de saber com d’efectius son aquests mètodes i com aplicar-los a situacions on els resums s’han d’adaptar a consultes o preferències formulades per l’usuari.

En aquells casos en que no és pràctic aprendre models de seqüència a seqüència, convé tornar a una formulació més tradicional del resum automàtic on els documents d’entrada s’analitzen en primer lloc, es planifica el resum tot seleccionant i organitzant continguts i el resum final es genera per extracció o abstracció, fent servir mètodes de generació de llenguatge natural en aquest últim cas. Separar l’anàlisi lingüístic, la planificació i la generació permet aplicar estratègies diferents a cada tasca. Aquesta tesi tracta el pas central de planificació del resum.

Inspirant-nos en recerca existent en desambiguació de sentits de mots, resum automàtic de textos i generació de llenguatge natural, aquesta tesi presenta una estratègia no supervisada per a la creació de resums. Seguim l'observació de que el rànquing d'ítems (significats o fragments de text) és un mètode comú per a tasques desambiguació i de resum, i proposem un mètode central per a la nostra estratègia que ordena significats lèxics i paraules d'un text. L'ordre resultant contribueix a la creació d'una representació semàntica en forma de graf des de la que seleccionem continguts no redundants i els organitzem per a la seva inclusió en el resum. L'estratègia general es fonamenta en bases de dades lexicogràfiques que proporcionen coneixement creuat entre múltiples idiomes i àrees temàtiques, i per mètodes de càlcul de similitud entre texts que fem servir per comparar significats entre sí i amb el text.

Els mètodes que es presenten en aquesta tesi son posats a prova en dues tasques separades, la desambiguació de sentits de paraula i d'entitats amb nom, i el resum extractiu de documents en anglès. L'avaluació de la desambiguació mostra que la nostra estratègia produeix resultats útils per a tasques més enllà del resum automàtic, mentre que l'avaluació del resum extractiu ens permet comparar el nostre enfocament a sistemes existents de resum automàtic. Tot i que els nostres resultats no representen un avenç significatiu respecte a l'estat de la qüestió en desambiguació i resum automàtic, suggereixen que l'estratègia té un gran potencial.

Contents

List of Figures

List of Tables

Glossary

AMR Abstract Meaning Representation. xi, 33, 34, 35, 36, 46, 58, 63, 75, 76, 77, 85, 106, 111, 157

AP Average Precision. 133, 134

AS Automatic Summarization. 1, 2, 5, 6, 7, 8, 11, 12, 17, 18, 22, 24, 25, 27, 31, 35, 38, 42, 44, 51, 141, 142, 150, 151, 155, 156

BERT Bidirectional Encoder Representations from Transformers. 127, 128, 129

BFS BabelNet First Sense. 140

BIO Beginning-Inside-Outside. 30, 31

BoW Bag-of-Words. 127, 135

BSD Bilexical Semantic Dependencies. 33, 34

CCG Combinatory Categorial Grammar. 27

CDTB Chinese Discourse Treebank. 37, 38

DCG Discounted Cumulative Gain. 132

DRS Discourse Representation Structure. 33

DSyntS Deep Syntactic Structure. 27, 28, 29, 75

DUC Document Understanding Conference. 61, 62, 63, 142

EDS Elementary Dependency Structures. 33, 34, 75

EDU Elementary Discourse Unit. 59

EL Entity Linking. 14, 18, 19, 23, 24, 25, 27, 29, 41, 44, 46, 47, 51, 60, 61, 66, 94, 122, 136, 139, 156

ES Entity Salience. 60

FOL First Order Logic. 33

HITS Hyperlink Induced Topic Search. 52, 54

HPSG Head-driven Phrase Structure Grammar. 27, 28

IDF Inverse Document Frequency. 53, 63

IE Information Extraction. 5, 17, 18, 22, 40, 41, 47, 130

ILP Integer Linear Programming. 57

KB Knowledge Base. 35, 40, 42, 43, 52, 68, 71, 72

KE Knowledge Extraction. xi, 19, 43, 44, 45, 46

LC Lambda Calculus. 33, 34

LD Linked Data. 43, 44

LDM Linguistic Discourse Model. 37

LSA Latent Semantic Analysis. 60

LTAG Lexicalized Tree Adjoining Grammar. 27, 28

MAP Mean Average Precision. 133, 134

MFS Most Frequent Sense. 138

MLMS Masked Language Model Scoring. 128

MRP Mean Reciprocal Rank. 132

MTT Meaning Text Theory. 27

MWE Multi-Word Expression. 14, 27, 29, 47, 65, 70, 71, 72, 73, 82, 83, 84, 85, 89, 93, 94, 131, 132

NDCG Normalized Discounted Cumulative Gain. 132

NE Named Entity. 5, 23, 31, 36, 45, 63, 76, 140, 145, 156

NER Named Entity Recognition. 18, 19, 21, 22, 23, 24, 25, 27, 29, 44, 46, 47, 48, 58, 60

NIF NLP Interchange Format. 44

NLG Natural Language Generation. 1, 2, 3, 4, 5, 6, 7, 8, 12, 14, 28, 35, 56, 64, 65, 81, 99, 106, 116, 153, 155, 156

NLP Natural Language Processing. 1, 3, 13, 17, 26, 34, 35, 43, 44, 46, 73

NLU Natural Language Understanding. 5, 18, 47, 153, 156

Open IE Open Information Extraction. xi, 18, 19, 40, 41, 42, 46, 57, 58, 65

PDTB Penn Discourse TreeBank. 37, 38

PoS Part-of-Speech. 41, 43, 55, 56, 57, 63, 137

QA Question Answering. 34, 142

RDA Relational Discourse Analysis. 37

RDF Resource Description Framework. 43, 48

ROUGE Recall-Oriented Understudy for Gisting Evaluation. 58, 62, 142, 143

RST Rhetorical Structure Theory. xi, 37, 38, 39, 47, 59, 157

RST-DT RST Discourse Treebank. 38

SIF Smooth Inverse Frequency. 127

SoA State of the Art. 5, 13, 14, 25, 46, 67, 122, 123, 128, 139, 144, 148, 154, 155

SRL Semantic Role Labelling. 29, 30, 31, 32, 33, 39, 44, 48, 54, 57, 58, 65

SSyntS Surface Syntactic Structure. 75

SW Semantic Web. 40, 43, 44

TAC Text Analysis Conference. 142

UCCA Universal Conceptual Cognitive Annotation. 33, 34, 157

UD Universal Dependencies. 26, 27, 65, 69, 71, 72, 73, 75, 77, 85, 104, 106, 111, 121, 123, 124, 145, 157

WFS WordNet First Sense. 138, 140, 141

WMD Word Mover's Distance. 127

WSD Word Sense Disambiguation. 14, 18, 23, 24, 25, 27, 29, 47, 51, 59, 60, 66, 94, 122, 136, 137, 138, 139, 140

Chapter 1

INTRODUCTION

Automatic Summarization (AS) is an Natural Language Processing (NLP) task concerned with the production of informative, non-redundant and linguistically well-formed summaries that capture the gist of one or more natural language documents. Summaries produced by a summarization system are considered *abstracts* if they rewrite or paraphrase the input text and *extracts* if they reproduce fragments of the input verbatim, a distinction that goes back to the early days of research on AS (Luhn, 1958) and that has given way to abstractive and extractive paradigms.

The traditional view on abstractive summarization is that the system attempts to replicate the steps followed by human authors when writing abstracts. According to Mani (2001), an abstractive summarizer starts by analyzing the source text in order to gain a deep understanding of the information communicated in it. This understanding results in an explicit intermediate representation of the information extracted from the text that, in a second phase, is transformed into a representation or plan of the summary to be generated. In a last synthesis phase, the plan is mapped back to natural language using Natural Language Generation (NLG) methods. Abstractive summarization presents significant challenges due to the limitations of the methods available for analysis and generation, and the fact that human authors use both linguistic competence and extra-linguistic knowledge to write abstracts (Torres-Moreno, 2014). This situation has resulted in many abstractive summarization systems being tightly constrained to specific domains, languages and genres.

Challenges associated with abstractive summarization led to extractive methods being considered the practical approach to AS for a number of years (Nenkova and McKeown, 2011). Works following the extractive paradigm select fragments of the source documents, most often sentences, to compose a summary by putting together selected fragments into an extract of the original text. This approach is attractive to practitioners because it requires neither deep understanding nor NLG. Extractive summaries, however, suffer from a number of limitations. First and foremost, they cannot attain the same levels of linguistic quality as a summary generated following an abstractive approach. It is notoriously difficult for extractive summarizers to guarantee that the text fragments used to compose the summary will result in a coherent and fluent text. Besides, extractive methods require that both input and output texts are in the same language, and lack flexibility to adapt their output to different summary styles.

In addition to the traditional abstractive and extractive paradigms, a third paradigm has emerged in recent years where abstracts are produced directly by paraphrasing the input without going through any of the separate phases and intermediate representations associated with abstractive summarization. This paradigm, henceforth referred to as *paraphrastic summarization*, has been bolstered by deep learning methods and *Sequence to sequence (seq2seq)* models (Sutskever et al., 2014) that map the sequence of tokens in the input text directly to a sequence of summary tokens. Seq2seq models are achieving impressive results on automatic qualitative evaluations and have become dominant in the extractive and paraphrastic approaches to summarization –albeit the latter is often portrayed as abstractive in the literature despite exhibiting a low degree of abstraction (Zhang et al., 2018). Notwithstanding their success and widespread usage in the field, seq2seq models are also constrained by some severe limitations.

Most seq2seq summarization models are trained on large corpora of texts paired with handwritten summaries. New languages and domains require abundant training data, but sizable datasets available to the research community are largely confined to English texts belonging to the news domain. Researchers have begun experimenting with transfer learning methods to overcome this limitation. Such methods aim to transfer knowledge from pretrained language models learned from massive collections of unannotated texts, quite often in multiple languages, to tasks and languages with little or no training data. As we write, new research is being published describing new insights and applications of transfer methods to NLP tasks. This is a very recent research direction, however, and there are many open questions regarding the performance of adapted models for summarization when compared to seq2seq models trained directly on the target language and domain. Empirical evidence suggests that pretrained models exhibit poor performance in NLG and, in particular, fail at maintaining coherence (Ruder et al., 2019). Assessing the actual portability of these models to other languages and domains is made more difficult because, for the most part, they are being evaluated only with English newswire datasets. Furthermore, research on seq2seq and transfer learning for summarization has yet to consider scenarios where summaries need to be tailored to variable user requirements such as queries or user preferences, an area that received abundant research before the emergence of seq2seq models.

Another important limitation of seq2seq models is that they operate in a black-box fashion and completely forgo any intermediate representation from which useful insights could be gained about the information communicated in the text. In traditional approaches to summarization, the knowledge extracted from the texts to summarize is useful not just for the creation of summaries but also for any other tasks an NLP system may be tasked with. This is an important requirement in many real-world natural language systems where text-based summaries are just one of many desired outcomes. For these systems to fully benefit from a summarization strategy, it is important to follow an approach that produces not only summaries but also an interpretable representation of both the contents extracted from the source documents and of the results of their assessment for inclusion in the summary. Having an interpretable representation is also important for summarizers that seek to enrich summaries with contents not present in the source documents, be it background information, perspectives or opinions, etc.

In this thesis, we present an approach to summarization that aims to overcome some of the limitations of the current trend of seq2seq models for summarization -dependency on large training datasets, no reusable insights, few guarantees on coherence. Our proposed approach to planning summaries follows the conventional view on abstractive summarization according to which generating a summary involves the analysis of the input text, the creation of an intermediate representation, the selection and organization of contents, and the realization of these contents into natural language. Our approach covers the creation of an intermediate representation and the selection and organization of contents. These tasks are known in NLG literature as *text planning* (Reiter and Dale, 1997), a term that we adopt in this thesis document. Having a text planning step separate from text analysis and generation allows us to reuse planning results for other tasks and makes it easier to implement mechanisms that enforce coherence in the resulting text.

We propose an intermediate graph-based representation of contents that can be obtained using a variety of text analysis tools and supports the application of domain and language-independent methods for planning summaries. This independence from domain and language is made possible thanks to the large multilingual lexical databases that have been developed in recent years and which provide knowledge covering multiple domains (Färber et al., 2015; Färber et al., 2018). In our experiments, we use a dependency parser and BabelNet (Navigli and Ponzetto, 2012) in combination with our own methods to instantiate this representation.

The proposed approach uses ranking methods that contribute both towards the instantiation of the intermediate representation and towards the selection of contents from it for inclusion in the summary. Additional graph-based methods inspired in research on NLG are applied to remove redundancy and enforce coherence. In order to avoid dependencies on summarization datasets, all the methods proposed in this thesis are unsupervised.

Our claim that our text planning approach is a valid alternative to State of the Art (SoA) methods and that it can be used for other tasks is tested empirically by applying it to the task of disambiguating word senses and Named Entity (NE) meanings against BabelNet, and to the production of extractive summaries in English. While our empirical evaluation is limited to these two tasks, we will argue that the approach is potentially useful for other languages, tasks and types of summaries.

In the remainder of this chapter we provide some general context. We start in Section 1.1 and Section 1.2 by framing text planning in the wider contexts of the NLG and summarization research fields. In Section 1.3, we draw precise boundaries on the research covered by this thesis and formulate the research questions and goals of this thesis. Finally, we conclude this chapter by presenting the structure of the thesis document in Section 1.4. Many of the topics covered in this introduction will be brought up again in Chapters 2 and 3 and put into context with references to the SoA.

1.1 Text Planning as an NLG Task

NLG deals with the production of natural language from information encoded in some machine representation format. Over the years, research on NLG has produced methods that have been applied to a wide range of tasks in both "text-to-text" and "data-to-text" applications. In text-to-text applications, such as AS, the system starts from textual sources and applies Natural Language Understanding (NLU) and Information Extraction (IE) methods to analyze the text. The results of this analysis constitute the starting point for NLG. In data-to-text applications, on the other hand, NLG starts from existing data that may have been obtained from sources other than natural language.

Over the years, practitioners have identified a number of tasks in NLG (Reiter and Dale, 1997), the most prominent of which are:

1. Text planning, also known as macroplanning, which includes the following subtasks:
   - Content selection, which determines what contents are to be communicated.
   - Discourse structuring, which organizes contents into a text plan that guarantees coherence in the output text.
2. Sentence planning or microplanning, which is further subdivided into:
   - Lexicalization, which maps input contents onto language-specific lexical entries.
   - Aggregation, which merges partially overlapping content and linguistic structures to avoid repetition and to improve the fluency of the output.
   - Generation of referring expressions, which addresses the generation of anaphora and references according to a model of the reader's world.
3. Surface realization, which maps the specifications obtained from the preceding tasks onto a syntactically, morphologically and orthographically correct text.

Not all applications of NLG involve addressing all and each one of these tasks, however. The complexity and overall architecture of an NLG system depends, to a large extent, on the characteristics of their input, context and expected output. Thus, content selection is not needed if all input contents must be verbalized, while discourse structuring and sentence planning can be omitted if the coherence and fluency of the output text are not important. Even surface realization can be bypassed if texts can be rendered using templates or canned text.

This division into tasks has some parallels with similar views on the summarization process. AS has been described as comprising three fundamental phases, one where the source text is analyzed, another where the results of the analysis are transformed for summarization purposes, and a last one where the natural language summary is rendered (Jones and Endres-Niggemeyer, 1995; Mani and Maybury, 1999). The transformation phase can be related to the text planning step of NLG systems, and synthesis to sentence planning and surface realization. This correspondence is not clear-cut, however. In foundational works in AS, transformation was often seen as comprising some condensation operations that went beyond content selection, such as aggregation and generalization, while discourse structuring was omitted (Paice, 1980; Molina, 1995; Mani and Maybury, 1999; Mani, 2001).

In practice, content selection becomes the only of the above tasks guaranteed to be considered by all approaches to summarization. Extractive approaches need neither sentence planning nor surface realization for obvious reasons and, as we will see in Chapter 3, many systems based on paraphrasing the source text do not use NLG methods either. In the case of abstractive summarizers, the number of NLG tasks being addressed depends on the nature of the intermediate representation obtained following analysis and the requirements imposed on the summary by the context.

Focusing on text planning, the prominence of content selection has led many summarization systems to ignore coherence concerns and consequently do not address discourse structuring. Looking at recent surveys of the state of the art in AS (Nenkova and McKeown, 2011; Torres-Moreno, 2014; Yao et al., 2017), it becomes evident that systems addressing ordering concerns constitute a fraction of the research on the area and, in many cases, do not include a content structuring step, but instead preserve the order and overall structure of a document by virtue of operating on a discourse representation of it.

In this thesis we present methods that cover both subtasks of macroplanning, that is, content selection and structuring. As we will see, our approach to structuring is limited to finding an ordering of contents that maximizes local coherence, and is not underpinned by any theory of discourse as is the case of more elaborate planners. While acknowledging their importance, both sentence planning and linguistic aggregation fall outside the scope of the research presented here.

1.2 Text Planning for Automatic Summarization

While AS can be succinctly defined as the task of producing a shorter version of a text that retains its most important aspects, a more nuanced description of the field requires some fundamental notions. Mani (2001) listed some of these notions:

- Informativeness: the coverage that a summary has of the information in the input text.
- Salience: the importance of information relative to the whole informational content of a document or collection of documents.
- Coherence: how are parts of the summary related to each other and how they contribute to the whole.
- Redundancy: the degree to which the same information is repeated across a text.

Taking these concepts on board, summarization can be redefined as the task of producing a shorter version of a text that is also informative, salient, coherent and non-redundant. AS -like NLG- is a broad and diverse research field encompassing many different scenarios, each of them determining the importance and precise nature of the notions above. Table 1.1 attempts to provide a general view of this variety by listing some of the dimensions over which summarization systems may vary, grouped on the basis of whether they are related to the source documents of a summarization system, to the output summary or to the context in which summaries are produced. While similar distinctions have been made in the literature and used to characterize general approaches to summarization systems (Hovy and Marcu, 1998; Mani and Maybury, 1999; Mani, 2001), here we focus on distinctions relevant to planning aspects –leaving aside concerns about input analysis and synthesis of summaries. Our discussion of the dimensions shown in Table 1.1 will serve to delimit the scope of the research presented in this thesis.

Source	
Domain	Topics and genre of the input texts, e.g. biographies, scientific texts, news articles, etc.
Size	Single or multiple documents, social media posts versus long documents.
Language	Are all the input documents in the same language?
Summaries	
Length	Single sentence, single paragraph or multiple paragraphs.
Language	Is the summary in the same language as the input?
Type	Extractive, paraphrased or abstractive
Context	
User profile	User type (e.g. layman, expert) and preferences, update summaries, etc.
User request	Summary focused on specific aspects, e.g. an entity or topic.
Communicative goals	Indicative, informative or critical

Table 1.1: Dimensions of summarization systems with respect to their source documents, output summaries and context.

Different domains bring about different conceptions of what is relevant or salient and how is information communicated in the text. In order to avoid having to manually encode this understanding for each domain using rules or other symbolic representations, researchers estimate salience using numerical methods. As we will see in Chapter 3, popular cross-domain approaches to relevance estimation include finding the most central topics amongst all those detected in a document or using supervised learning to learn relevance models from copora of documents and gold summaries.

The number and size of the input texts determine the importance of strategies for redundancy removal and planning coherent summaries. Redundancy and coherence are greater concerns in multi-document summarization due to the increased risk of finding replicated information across documents and the challenge presented by integrating contents coming from different texts. Redundancy is less critical when producing summaries from a single short text but, in contrast, coarse-grained selection methods such as sentence extraction risk including too much irrelevant information. The length of the target summaries is also a decisive factor for the adoption of alternative planning strategies. Summaries consisting of a single or a few sentences have less need for structuring contents than those targeting longer texts where coherence becomes more important. The relation between the source text and the summary is often expressed as a compression rate parameter to the summarization system (Mani, 2001).

A system can be either monolingual or multilingual depending on whether it is capable of processing documents in one or more than one language. In addition, a cross-lingual system is also capable of producing summaries in a different language than that of the input. The cost of supporting additional languages can be reduced if the system operates on a language-independent representation of the input, as this makes it possible to address selection and structuring independently of the input and output languages. This is the strategy followed by many participants in the MultiLing workshops (Giannakopoulos, 2013; Giannakopoulos et al., 2015, 2017) that aim at promoting the development of language-independent methods for AS.

The type of the target summary also plays a significant role. If one is willing to accept the limitations of extractive summaries, then the structuring of contents can be reduced to ordering selected fragments. Extractive or paraphrased summaries can be obtained with shallow linguistic analysis, while generating an abstract requires knowledge about the information communicated in the text that can only be obtained using deep analysis methods. This variation in the depth of the analysis has a big impact on the type of information that a text planning strategy must deal with. Different planning strategies are required depending on the type of analysis output, which can range from representations or features based on words and their position in the text to linguistic representations at various levels of abstraction, and conceptual representations –these representations and related methods are described in detail in Chapters 2 and 3.

Contextual factors also come into play when designing a strategy for planning summaries. The presence of a profile of a user or a user query have given rise to research lines into specific types of summaries such as query-oriented and update summaries (Jones and Endres-Niggemeyer, 1995). Notions like salience and redundancy are substantially redefined in the presence of such factors. Redundancy, for instance, must take into account what the user already know. Consequently, planning strategies must be able to adapt the selection and structuring of contents. This adaptive behavior is unlikely to be learned from corpora, which implies that the methods used for planning must be able to accommodate for user bias explicitly.

The distinction between indicative, informative and critical summaries (Mani, 2001) predates research on AS and is inherited from library and information sciences. It refers to the communicative goals of the author, or system in our case, with respect to the information it wants to provide to the user. Thus, indicative summaries point at where relevant information may be found in the input while informative summaries communicate this information directly, and critical or affective summaries attempt to express views on the input and influence the perception of the addressee.

While informative summaries are by far the most researched area in AS, authors have taken inspiration in affective NLG (de Rosis and Grasso, 1999) to imprint summaries with views and emotions not communicated in the input text.

1.3 Research Goals

In this thesis, we propose an approach for planning informative summaries from one or more documents. We delimit the scope of our research by relating it to each of the dimensions listed in Table 1.1:

- We do not commit to texts belonging to any specific genre, domain or area of knwoledge. All methods proposed in this thesis are domain-independent.
- Our approach is valid for summarization of single or multiple documents of any length.
- Our methods are language-independent and special care is placed into choosing linguistic resources and tools that are available for multiple languages.
- We assume that the input text and the target summary are in the same language.
- The approach is suited for both extractive, paraphrased and abstractive summarization.
- Neither a user profile nor a user request are considered in our approach.
- Our approcah produces informative summaries. No additional communicative goals or summary types are considered.

When discussing the limitations of seq2seq models dominating current research on AS, we mentioned their black box nature and dependency on large datasets containing documents and gold summaries. We believe that addressing analysis, planning and generation separately offers a number of potential advantages, namely (i) re-using methods and intermediate results, (ii) portability to other languages and domains, (iii) adaptability to queries and user preferences, (iii) improved coherence. This belief constitutes the main hypothesis of this thesis.

While this is a very wide hypothesis, we wish to contribute a partial answer to it by presenting a viable text planning strategy that follows our vision and an empirical evaluation that compares it against SoA summarization systems. We focus on the following points, which also constitute the research goals for this thesis:

1. Contribute a text planning strategy the results of which can be effectively used both for producing summaries and for other tasks,
2. implement this strategy using unsupervised methods that do not require training corpora,
3. integrate methods to guarantee a coherent presentation of contents in the summary and
4. keep all the above methods independent from the language of the texts and the topics they touch upon.

In attempting to satisfy these goals, we make the following contributions:

- We describe a flexible graph-based and language-independent representation of the linguistic meaning of one or more documents, which we refer to as *planning graph*.
- We detail how to instantiate planning graphs out of natural language texts based on a mixture of our own methods and off-the-shelf multilingual NLP tools and resources.
- We propose a domain and language-independent unsupervised approach to text planning based on this representation that covers both selection of contents and finding an optimal order for coherent presentation in the summary.

- We empirically test the results of this text planning strategy to the tasks of Word Sense Disambiguation (WSD), Entity Linking (EL), and extractive summarization.

Our approach follows the observation that unsupervised and graph-based ranking methods have been applied in the past by researchers to address both disambiguation and summarization problems. Thus, we apply ranking of lexical meanings as a strategy for the disambiguation of words and Multi-Word Expression (MWE)s, and for semantically-oriented selection and structuring of contents for summarization purposes. Operating on a meaning-based representation is key to ensuring that our planning methods remain language-independent. Cross-domain operation is supported by leveraging large lexical knowledge bases that have been made available in recent years and which cover both word senses and encyclopedic knowledge.

Since other NLG tasks like sentence planning and surface realization are beyond the scope of this thesis, we test our methods on the production of extractive summaries that do not require a realization component and which allows us to evaluate our planning methods on their own. Nevertheless, existing text generators could be used to produce abstracts from the results of our planning approach, a prospect that will be discussed in Chapter 7.

1.4 Structure of the Thesis

This thesis is structured as follows. In Chapter 2, we provide an overview of the fundamentals on which our approach is based, i.e., existing text analysis tools, methods and resources. In Chapter 3, we survey the SoA in the field of summarization, placing special emphasis on the knowledge extracted from text, the methods used to obtain this knowledge and the approaches to select and organize contents for the production of summaries. These two chapters provide the background for the planning graph representation presented in Chapter 4 and the means to obtain it from natural language. Based on this representation, Chapter 5 describes our approach to text planning. We empirically test this approach in relation to the research goals by evaluating it on the tasks of disambiguating lexical meanings and producing extractive summaries in English. The experiments and general evaluation procedure are described in Chapter 6, while the results are discussed in Section 7.1. In Chapter 7, we look at how our research has helped in answering the research questions of this thesis. Finally, in Section 7.2, we give insights on how this research can be continued in the future.

Chapter 2

FUNDAMENTALS

In the previous chapter, we argued that going back to a more traditional view on AS presented a number of advantages over alternative formulations. Recall that this traditional view dictates that the documents to be summarized are analyzed first in order to extract knowledge about their contents. This knowledge is then used to apply text planning methods that select what contents should be communicated and to determine the overall organization of these contents in the summary. Recall also that the nature of the knowledge extracted by the analysis of the text has a profound impact on the approach followed for text planning. For this reason, it is necessary to determine what exactly we mean by *knowledge* extracted from text before choosing an approach to text planning, and characterize this knowledge in terms of its scope, its depth and its overall organization.

This characterization, in turn, requires knowing what NLP and IE analysis tools, methods and resources are available for extracting it from natural language. The goal of this chapter is to review the state of affairs in text analysis by looking into the strengths and weaknesses of available tools in relation to the goals of our research. This overview will provide the necessary background for our argumentation in Chapter 4 in favor of planning graphs as an intermediate representation for summarization, and the choice of specific tools and resources used to obtain planning graphs from text. Our overview is divided into sections covering popular NLU and IE tasks. Rather than attempting to give a full account of the theoretical frameworks and methods applied to each task, we adopt a more practical outlook and focus on those aspects that have the greatest impact on planning summaries and are most relevant to our research goals. For each, we will describe the knowledge, linguistic or otherwise, targeted in each task, the annotations or representations typically produced, the availability of tools and resources, the dependencies on other analysis tasks, and their application for summarization purposes. While this chapter includes references to works in the field of AS, the state of the art will be described in detail in Chapter 3.

In order to organize our review, we classify tasks based on two dimensions, as shown in Figure 2.1. The first dimension distinguishes between tasks that produce “language-oriented” representations that describe linguistic aspects of the text, functional, semantic or presentational, and tasks that produce “knowledge-oriented’ representations that attempt to model the contents of the text beyond linguistic considerations. In the first category we include tasks such as syntactic parsing, semantic parsing, discourse analysis and WSD, while the second category comprises tasks like Named Entity Recognition (NER), EL, Open Information Extraction (Open IE) and closed relation extraction [1]. This distinction is relevant to our research goals because knowledge-oriented representations are likely to be closer to the actual contents of a text and make it easier to apply unsupervised language-independent text planning strategies. Language-based representations, however, may be easier to obtain from multiple languages, genres and domains.

We further divide NLU and IE tasks into those that produce “comprehen-

[1] We exclude from the chapter IE tasks related to ontology learning such as taxonomy induction, as they use natural language corpora as a means to model knowledge about a domain rather than aiming at extracting the actual meaning or information conveyed in the texts.

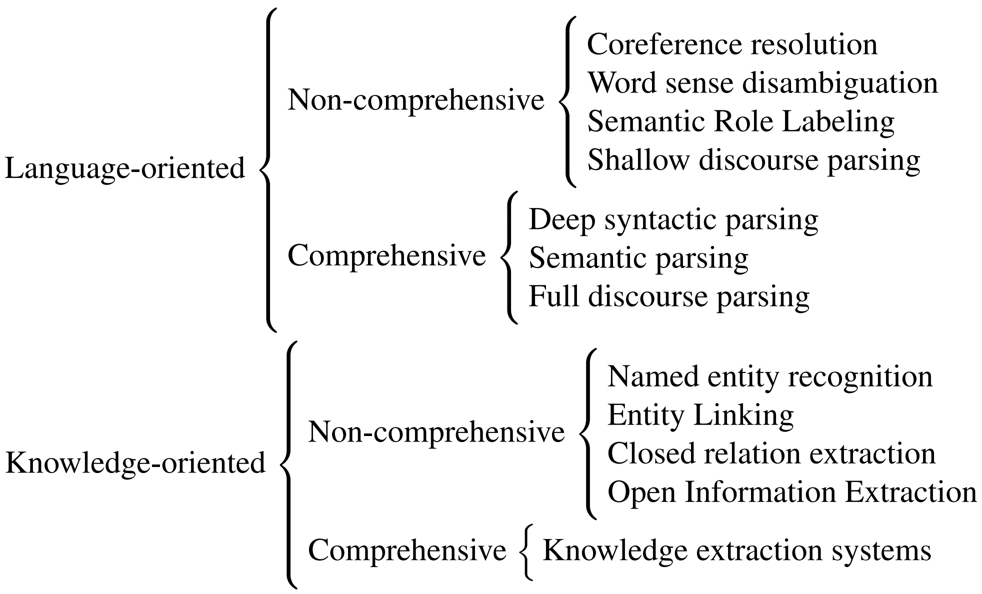


Figure 2.1: Classification of analysis tasks

sive" and those that produce "non-comprehensive" representations. The former attempt to produce unified representations covering the whole text, e.g., a syntactic analysis of a sentence or a discourse analysis of a text, or representations that model the overall meaning of a text, i.e., Knowledge Extraction (KE) systems. Non-comprehensive representations, on the other hand, produce multiple disconnected annotations, either because they focus on certain linguistic phenomena or aspects of meaning, e.g., coreference resolution, NER and EL, or because they do not produce a single connected representation, e.g., closed relation extraction and Open IE. This second distinction is also relevant to our research goals as non-comprehensive representations tend to provide a partial view on a sentence or document, making it more difficult to assess the analyzed text as a whole. Nonetheless, analysis tools targeting "non-comprehensive" provide in-depth knowledge about specific aspects of language or meaning that is often not included in comprehensive representations.

In order to illustrate the knowledge extracted in each task, we will use the following text fragment as a running example:

> *John Major met Jacques Chirac in London to discuss nuclear energy, two months after meeting in Paris. This, however, was not his first encounter with the French president in the British capital.*

In the following, we will review each of the tasks enumerated in Figure 2.1. The chapter will end with a brief discussion of common traits between all tasks and how they can serve our goals.

2.1 Coreference Resolution

Purpose and scope: The goal of coreference resolution is to identify linguistic expressions in a document that denote the same entity. This task typically involves resolving anaphoric and cataphoric expressions, the meaning of which can only be determined in relation to one or more preceding expressions (antecedents) or subsequent expressions (postcedents) respectively. However, not all anaphoric expressions are considered coreferent, as anaphora and their antecedents can refer to related but distinct entities, e.g., *door* and *car* in *John entered the car and closed the door*.

Research on coreference resolution usually focuses on entity coreference between nominal expressions, with other types of anaphora such as event coreference and null proforms (ellipsis) receiving less attention (Lu and Ng, 2018). Research is also typically constrained to resolving expressions that can be successfully interpreted using linguistic context, as opposed to exophoric expressions such as deictics whose interpretation depends on extralinguistic content (Poesio et al., 2011).

Output representation: Coreference resolution tools produce sequences of annotated mentions to a specific entity or event. In these sequences or coreference chains, one of the annotations is marked as the main mention and often corresponds to a fully qualified reference to the entity. Remaining annotations in a chain are either repetitions or abbreviated forms, e.g. proforms. Coreference chains, which usually span over multiple sentences in a text, are not connected to each other and only cover expressions that corefer. In addition, coreference resolution tools do not attempt to determine the entity being denoted, and only annotate the linguistic expressions involved. Consequently, coreference chains can be classified as language-oriented, non-comprehensive representations.

Tools and resources: Existing approaches to solving coreference can be characterized by their coverage of coreference phenomena, by the languages supported and by the analysis tools they depend on. While the CoNLL 2011 and 2012 shared tasks in coreference resolution (Pradhan et al., 2011, 2012) were based on multilingual corpora –English, Arabic, and Chinese– annotated with various types of coreference, including event coreference, most participating systems and tools that have become available since then only cover the case of single-antecedent (or postcedent) nominal coreference, and have limited or no multilingual support (Kübler and Zhekova, 2016).

Apart from coreference annotations, the corpora used in the CoNLL shared tasks were annotated with syntactic parses, named entities, disambiguated word senses and semantic roles for predicate arguments. This set of annotations reflects the most common types of analysis required by coreference solving methods. This can be confirmed by looking at existing coreference tools, for instance the Stanford CoreNLP coreference module (Clark and Manning, 2015), the Berkeley entity resolution system (Durrett and Klein, 2014) and the Illinois Coreference Package (Bengtson and Roth, 2008; Peng et al., 2015). All these tools produce coreference chains from texts in English and only cover nominal and pronominal mentions [2]. With respect to their requirements, the first two expect their input to contain syntactic analysis and NER annotations, while the Illinois system only requires NER annotations.

[2] Stanford CoreNLP also supports Chinese

Use for AS: Coreference resolution has been applied to improve summarization systems that rely on metrics or features based on lexical chains or frequency (Nenkova and McKeown, 2011; Fang and Teufel, 2016) and to improve the coherence in extractive systems by ensuring that anaphoric expressions always have an antecedent (Durrett et al., 2016). Some of these works have applied off-the-shelf tools (Bing et al., 2015; Durrett et al., 2016), but in many cases the authors choose to implement their own strategies tailored to summarization purposes (Cheung and Penn, 2014; Fang and Teufel, 2016).

Example: Figure 2.2 shows four coreference chains taken from our example. In the top-left chain, the possessive pronoun *his* is an anaphoric expression that can be resolved to corefer with *John Major*. The chain at the bottom-left of the figure shows three text fragments –*met*, *this* and *his first encounter*– denoting the same event. Identifying the remaining two chains, the first one containing *Jacques Chirac* and *French President*, and the second containing *London* and *British Capital*, requires knowledge beyond that provided by the text. Only tools using For this reason, most coreference tools would not be able to annotate them.

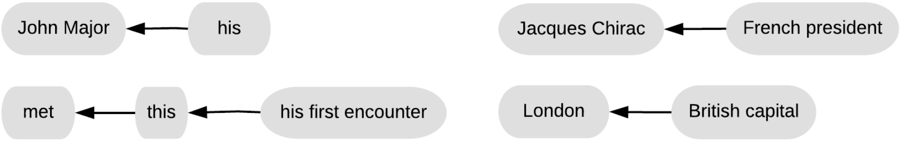


Figure 2.2: Coreference chains in our example.

2.2 NER, WSD and EL

Purpose and scope: Named Entity Recognition (NER) is an IE task consisting of aligning proper names with a semantic type according to the real-world objects they denote, but without resolving the referred entity. Various sets of entity types have been proposed over the years for the

NER task, including hierarchical ontologies that allow annotation at different levels of granularity. Most sets include labels for semantic types such as ‘person’, ‘location’ and ‘organization’.

Word Sense Disambiguation (WSD) is the task of choosing the right meaning of a word according to the context in which it occurs. Candidate meanings are obtained from dictionaries containing word senses. Entity Linking (EL), sometimes referred to as “named entity disambiguation” or “normalization”, is another disambiguation task that seeks to determine the entity denoted by a word or multi-word expression according to a database or encyclopedic resource, e.g., Wikipedia.

Output representation: Tools addressing NER, WSD or EL produce annotations over linguistic expressions. In the case of NER, these annotations contain a reference each to specific semantic categories, while WSD and EL establish links to entries in a dictionary containing word meanings or an encyclopedic resource, respectively. While WSD annotates single content words of any grammatical category, both NER and EL annotate nominal expressions. As in the case of coreference resolution, their annotations do not constitute a comprehensive representation.

Tools and resources: Approaches to NER can be divided into those employing knowledge-based methods to detect mentions of entities according to some encyclopedic resource and those that are based on linguistic information alone. Tools in the first group are often based on Wikipedia or domain-specific resources, while the second group are primarily based on models trained on annotated corpora. A wealth of NE-annotated corpora and public tools are available, particularly for English, but they do not always annotate with exactly the same sets of entity categories (Li et al., 2018). While some off-the-shelve tools support a large number of languages, e.g., POLYGLOT (Al-Rfou et al., 2015) and spaCy (Honnibal and Montani, 2017) have substantial multilingual support, cross-domain NER is still confined to research papers reporting experiments with deep transfer and active learning methods.

The two disambiguation tasks, WSD and EL, are distinguished above all by the type of sense inventory they use. Dictionaries contains lemmas of words belonging to any grammatical category and, in the case of nouns, they are largely constrained to common nouns. The most commonly used dictionary for WSD is, by far, WordNet (Miller, 1995). Conversely, Encyclopedic databases tend to focus on nominal expressions and, in particular, on proper nouns used as names for real word entities. The emergence of mappings between dictionaries, encyclopedias and other databases, e.g., BabelNet (Navigli and Ponzetto, 2012), has led to joint approaches to WSD and EL gaining prominence in recent years, e.g., (Moro et al., 2014; Weissenborn et al., 2015).

Methods for disambiguation can be distinguished between those that are based on the local context of the word or words to disambiguate, and those that use a global context. The former select a window of words to extract features from, e.g., (Lin, 1997; Navigli, 2009; Mendes et al., 2011), while the latter model whole documents using graph-based or other structured representations, e.g., (Agirre and Soroa, 2009; Navigli and Lapata, 2010; Han et al., 2011; Moro et al., 2014; Usbeck et al., 2014; Weissenborn et al., 2015). Global contexts can be used to perform collective disambiguation of all words in a text so that the choice of meanings for each word is influenced by all other words in the text. A few of these tools support multiple languages, e.g., DBPedia Spotlight (Mendes et al., 2011), AGDISTIS (Usbeck et al., 2014) and Babelfy (Moro et al., 2014).

Use for AS: NER annotations have been used as features to estimate the relevance of contents, under the assumption that their presence indicates potentially salient information. In systems that assume that repetition of contents is an indicator of relevance, named entities also serve to improve frequency counts by treating multi-word proper nouns as a single lexical unit, e.g., counting *John Kennedy* as a single occurrence of the name rather than counting separately for *John* and *Kennedy*. Disambiguation against WordNet is used to obtain *lexical chains*, sequences of semantically related noun phrases which have been applied to extractive summarization in a number of works reviewed by Nenkova and McKeown (2011), and also in more recent works, e.g., (Fang and Teufel, 2016).

EL has seen little use by the AS community, a situation that can be attributed to the limitations of many existing tools. The importance of salient entities in relation to identifying the topics of a document means that content selection methods for summarization are highly sensitive to errors in the detection and disambiguation of entities. Unfortunately, SoA EL tools not only suffer from relatively low precision and recall compared to simple baselines, but in many cases they also produce ambiguous references. This is the case of wikifiers producing pointers to disambiguation pages in Wikipedia that include completely irrelevant entities. This state of affairs has led some researchers to apply their own methods to improve EL in the context of a summarization system, e.g., (Trani et al., 2016; Amplayo et al., 2018). We will go back to these works in Chapter 3, where we will discuss similarities with our approach.

Example: Applied to the first sentence in our example, a NER tool would mark *John Major* and *Charles Chirac* as 'person', and *London* and *Paris* as 'location'. Depending on the typeset, *French* and *British* may be annotated with labels like 'misc', 'country' and 'nationality', amongst many others. Temporal expressions like *two months* are also annotated by some tools. A WSD tool would assign disambiguated senses to most single words in our example. WordNet, for instance, covers nouns, verbs, adjectives and adverbs. Albeit WordNet is not restricted to common nouns and single words (in our example it covers *London*, *Paris* and *nuclear energy*), encyclopedic resources like Wikipedia have a much larger coverage of proper names and multi-word expressions. Thus, an EL tool should be able to find the right sense for mentions not contained in WordNet such as *John Major*, *Charles Chirac*, *French president*, and *British capital*.

2.3 Syntactic Parsing

Purpose and scope: Syntactic parsing is the task of producing a linguistic analysis, in the form of a parse tree, that reflects the structure of words within a sentence. Syntactic parsing follows two paradigms, constituency and dependency-based parsing –the latter becoming more dominant in recent years. In dependency parsing, parse trees are composed of words and relations between them, as opposed to constituency-based approaches that introduce additional nodes to indicate grammatical constructs. Another distinction in syntactic parsing differentiates between surface parsing, which is concerned with functional aspects of language, and deep parsing, which produces more abstract analyses that reflect predicate-argument structures at the syntactic level and other relations between meaning-bearing words such as attributive, appenditive or coordinative relations.

Output representation: Syntactic parses are ordered, rooted trees and, consequently, must (i) be connected, (ii) possess a single root, (iii) have a single parent for each node and (iv) cannot contain any cycles. Constituency trees distinguish between terminal nodes (leaves of the tree) that correspond to words in the sentence and internal nodes that indicate grammatical categories. Dependency trees only contain terminal nodes and have edge labels indicating dependency relations between nodes. In both cases, terminal nodes are sorted according to the order of appearance of words in the sentence. Syntactic parses are comprehensive linguistic representations of sentences. While shallow syntactic parses have terminal nodes for all words in a sentence, deep syntactic trees ignore function words and only contain nodes for content words.

Tools and resources: Surface syntactic parsing is a very active field of research, with several competitions held regularly in which participants are evaluated. As in many other NLP tasks, deep learning methods trained on manually annotated corpora are by far the preferred approach these days. Corpora can be annotated on a number of language-specific tagsets –sets of syntactic relations–, e.g., Penn Treebank for English (Taylor et al., 2003), AnCora for Spanish and Catalan (Martí et al., 2007). More recently, the Universal Dependencies (UD) initiative (Smith et al., 2018) has promoted the creation of corpora in over 70 languages using a common syntactic annotation standard. This has led to UD-trained parsers becoming popular in the field (Qi et al., 2018; Che et al., 2018). Furthermore, the training of cross-lingual models using transfer learning methods has enabled parsing even in languages for which no UD-corpora exists (Ammar et al., 2016; Schuster et al., 2019).

A number of deep syntactic parsers have been developed over the years that produce different representations according to various theories of deep syntax, mainly lexicalized grammars that incorporate predicate-argument relations, e.g., the C&C parser (Clark and Curran, 2007) for Combinatory Categorial Grammar (CCG) (Steedman, 2000), the Enju parser (Sagae et al., 2007) for Head-driven Phrase Structure Grammar (HPSG) (Proudian and Pollard, 1985), and several parsers (Lopez, 2000; Chiang, 2000) for Lexicalized Tree Adjoining Grammar (LTAG) (Joshi and Schabes, 1997). More recently, Ballesteros et al. (2016) presented a dependency-based parser for Chinese, English and Spanish based on the Deep Syntactic Structure (DSyntS) formalism of the Meaning Text Theory (MTT) (Mel'čuk, 1988). On-going efforts in developing deep UD annotation guidelines and treebanks (Droganova and Zeman, 2019) may result in new deep parsers being developed in the near future.

Recent annotation schemas that incorporate tags to indicate MWEs (de Marneffe et al., 2014; Nivre et al., 2016) and strategies for integrating dependency parsing and multiword detection (Candito and Constant, 2014) facilitate the unification of parse trees with semantic analysis produced by coreference, NER, WSD and EL tools.

Use for AS: As we will see in Chapter 3, surface syntactic parsers have been used extensively in summarization systems to construct intermediate representations from which contents are extracted. A number of works have experimented with the creation of dependency graphs for whole documents by merging the dependency trees of individual sentences. In addition, syntactic relations have also been applied to detect redundant text fragments, compress or simplify sentences, create or learn patterns, select relevant bigrams and to perform phrase-based extraction (Gambhir and Gupta, 2017; Yao et al., 2017).

There are a few cases of researchers who have experimented with deep parsers for summarization purposes. An early example is Barzilay et al.

(1999), who used DSyntSs structures to find common functional configurations across multiple sentences and combine them into sentences using NLG methods. Jing and McKeown (2000) followed a similar approach, but aggregating parse trees based on the LTAG formalism. More recently, the Enju parser has been applied to rank text fragments on the basis of their HPSG analysis for extractive summarization purposes (Yan and Wan, 2015). Deep syntactic parsers based on DSyntSs have also been applied to specialized abstractive summarization in fields like medical texts and patents (Da Cunha et al., 2007; Mille and Wanner, 2008), and efforts are being invested into applying it to general-purpose multilingual summarization (Mille et al., 2016). However, overall adoption of deep syntactic parsing in summarization approaches has been limited.

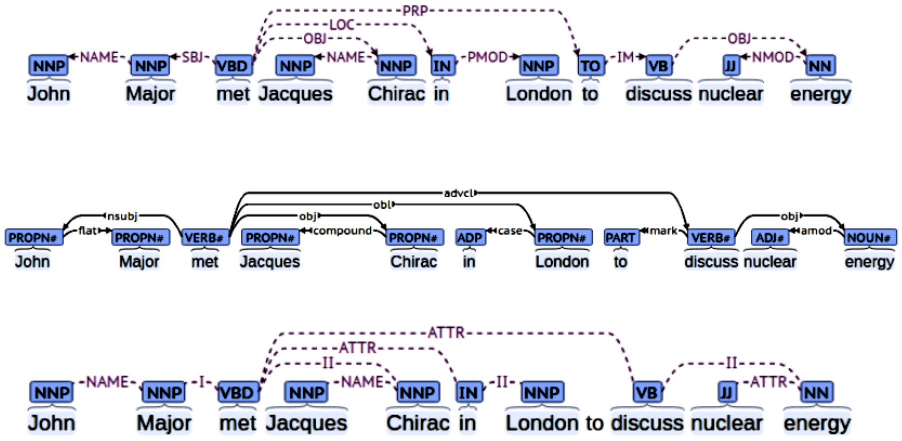

PRP
LOC
OBJ
NAME
SBJ
NAME
PMOD
IM
OBJ
NMOD
NNP NNP VBD NNP NNP IN NNP TO VB JJ NN
John Major met Jacques Chirac in London to discuss nuclear energy
advcl
obl
nsubj
obj
flat
compound
case
mark
obj
amod
PROPN PROPN VERB PROPN PROPN ADP PROPN PART VERB ADJ NOUN
John Major met Jacques Chirac in London to discuss nuclear energy
ATTR
ATTR
II
NAME
I
NAME
II
II
ATTR
NNP NNP VBD NNP NNP IN NNP VB JJ NN
John Major met Jacques Chirac in London to discuss nuclear energy

Figure 2.3: Syntactic parses of a fragment of the first sentence of our example.

Example: Figure 2.3 shows various dependency-based syntactic parses for a fragment of the first sentence of our running example. The top parse corresponds to a Penn Treebank surface analysis, the middle one to a UD surface analysis and the one at the bottom to a DSyntS analysis. All three parses are directed trees where nodes correspond to words in the sentence and edges are labeled with a dependency relation: the node the edge stems from being the governor of the relation and the node pointed by the edge being the dependent of the relation. The diagram at the bottom of Figure 2.3 depicts a deep parse tree where predicate-argument relations are indicated with edges labeled 'I' and 'II' pointing to the first and second arguments of the verb *met* respectively. While the two surface syntactic trees contain all words in the sentence, the deep tree excludes functional words, i.e., the particle *to* in our example.

Moving closer to a lexicalist view of syntactics, dependency-based syntactically annotated corpora use special relation tags to indicate MWEs, e.g. (de Marneffe et al., 2014; Nivre et al., 2016). This can be seen in UD and DSyntS parses at the top and bottom of Figure 2.3, which use 'NAME' relations to indicate the proper names *John Major* and *Jacques Chirac*. This type of relations facilitate strategies for integrating dependency parsing with tools that annotate MWEs such as NER, WSD and EL tools (Candito and Constant, 2014).

2.4 Semantic Role Labeling

Purpose and scope: Semantic Role Labelling (SRL) consists in detecting relations between predicative words and linguistic expressions that act as semantic arguments of the predicate –expressions that complete its meaning according to a frame-based view of lexical semantics (Fillmore, 1968; Jackendoff, 1972). SRL tools operate at the interface between syntax and semantics, and are divided into tools that annotate syntactic frames and tools that annotate semantic frames. The former annotate each argument with a role indicating its position in the syntactic frame of the predicate, sometimes referred to as theta roles, while the latter associate semantic roles such as *agent*, *patient* or *instrument*, often referred to as thematic roles.

The syntactic realization of the semantic arguments of a predicative meaning is governed by the lexical unit that realizes this meaning in the text. For this reason, approaches to SRL rely on dictionaries that associate predicative words with syntactic or semantic frames.

Output representation: Many semantic role labelers produce Beginning-Inside-Outside (BIO) annotations to mark predicate-argument structures. This annotation style has some limitations, like not being able to represent overlapping predicate-argument structures. Other tools use a more elaborate representation consisting of a forest of potentially overlapping elementary trees (trees of depth 1). In both cases, predicates are annotated with a reference to a syntactic or semantic frame in a reference dictionary, while arguments are annotated as text spans associated with a theta or semantic role. The annotations produced by SRL tools are linguistic in nature and cannot be regarded as comprehensive representations as they focus only on those parts of a text that are part of predicate-argument structures.

Tools and resources: As mentioned before, tools for SRL are usually based on some dictionary of predicative words. PropBank (Palmer et al., 2005) and NomBank (Meyers et al., 2004) are English dictionaries that, for each lexical unit, enumerate their set of syntactic frames, *framesets* in PropBank terminology, that correlate with coarse-grained word senses. VerbNet (Schuler, 2005) is another English dictionary that contains both syntactic and semantic frames for verbal senses. Its semantic frames specify thematic roles and semantic selectional restrictions, e.g., 'animate' or 'organization'. Besides, VerbNet also classifies verb senses into to a hierarchy of 270 verb classes, reflecting similar semantic and syntactic properties. FrameNet (Ruppenhofer et al., 2006) does not contain syntactic frames, but associates lexical units belonging to several grammatical categories with semantic frames, in the sense of Fillmore (Fillmore, 1976). Versions of FrameNet for languages other than English have also been made available over the years. In addition to these dictionaries, linguists have also created mappings between them and others resources like WordNet and BabelNet, e.g., SemLink (Palmer, 2009) and Predicate Matrix (de Lacalle et al., 2016).

The development of SRL tools has been supported by a number of SemEval and CoNLL competitions and associated datasets (Mitkov et al., 2016).

Most approaches address the task as a sequential prediction problem of BIO tags using supervised learning methods –and more recently neural networks– and other analysis tools such as chunkers, syntactic parsers and NE recognizers. End-to-end labelers detect predicates in the text, disambiguate between their possible frames and annotate text spans as their arguments. Many SRL tools, however, do not address all these tasks. It is common for works in SRL, for instance, to assume that predicates are already given and focus only on the remaining tasks.

Multilingual support has been an important concern for practitioners in the field, particularly since the SemEval-2009 shared task on semantic parsing published SRL-annotated texts in 7 languages (Hajic et al., 2009). Since then, multilingual SRL has continued to be an active topic of research (Mulcaire et al., 2018; He et al., 2019). Despite this growth in interest, most publicly available end-to-end tools are limited to English, e.g., SENNA (Collobert et al., 2011) and Illinois (Punyakanok et al., 2008) labelers for PropBank, mateplus (Roth and Woodsend, 2014; Roth and Lapata, 2015) and LTH (Johansson and Nugues, 2007, 2008) for PropBank and FrameNet, and Shalmaneser (Erk and Pado, 2006) and Semafor (Chen et al., 2010) for FrameNet.

Use for AS: SRL with PropBank has been applied mostly in extractive summarizers (Chali and Joty, 2008; Wang et al., 2008; Aksoy et al., 2009; Yan and Wan, 2014) to improve similarity calculations across sentences by focusing on the events denoted by predicates. Instead of comparing text fragments as sequences or bags of words, the comparison is done on tuples extracted using a semantic role labeler that are treated as individual events. FrameNet SRL with Semafor (Chen et al., 2010) has also been used with similar purposes, both in extractive (Han et al., 2011) and in abstractive summarization (Li et al., 2015).

Example: Figure 2.4 shows some of the FrameNet frames that can be assigned to the first sentence of our example. The two uses of the verb *meet* are assigned the same frame ‘Meet_with’ but have different fillers for the role “Place”. Fillers for the roles “Party_1” and “Party_2” are the same in both cases. While they can be derived from the syntactic argu-

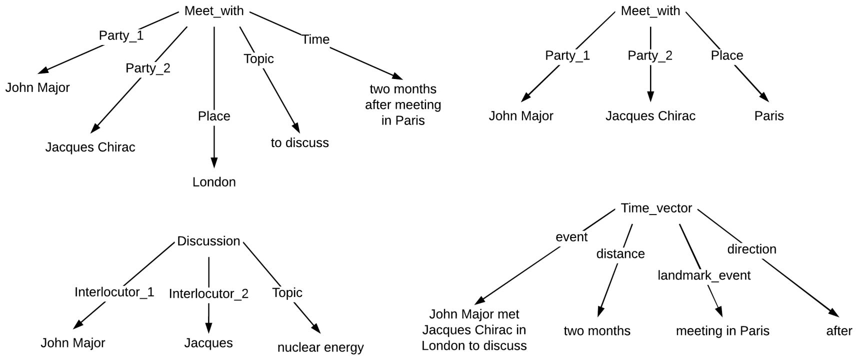


Figure 2.4: FrameNet annotations of the first sentence of our example.

ments of the main verb, in the second use of *meet*, a SRL tool would not be able to determine them based on syntax alone. It would face the same situation when determining the fillers for the roles "Interlocutor_1" and "Interlocutor_2". Unlike PropBank and VerbNet, FrameNet indexes words with grammatical categories other than verbs. Thus, the preposition *after* indexes the frame 'Time_vector' in FramNet, which is also shown in Figure 2.4.

2.5 Semantic Parsing

Purpose and scope: Semantic analysis is concerned with obtaining a representation of the part of meaning of natural language determined by linguistic form. It does not attempt to obtain a full interpretation of utterances that would require knowledge about the domain, the communicative goals of the speaker, etc. While tasks like coreference resolution and SRL are also related with aspects of meaning, *semantic parsing* is usually used to refer to the task of producing a comprehensive representation of the linguistic meaning of a sentence, i.e., an analysis that covers all meaning-bearing words in a sentence.

Like deep syntactic parsers and SRL tools, the semantic representations produced by parsers incorporate predicate-argument structure. However they may also explicitly model other types of semantic phenomena such as negation, modality, conjunction, comparatives, possessives, etc. Semantic parsers produce sentence-level comprehensive representations like deep syntactic parsers do, but are not constrained to producing parse trees, but more general directed graphs that can be unconnected, contain multiple roots and display reentrancies (nodes with multiple parents). Compared to SRL, semantic analysis involves considering how meanings are composed beyond predicate-argument structure and towards building a representation of the meaning of whole sentences.

Output representation: Semantic parsing is a field of research rife with alternative representations, a situation caused by a lack of agreement on what aspects of meaning should be represented and how.

Many semantic parsers produce formal representations based on logical formalisms, mostly First Order Logic (FOL) and Lambda Calculus (LC). Such parsers may adhere to theories of language that go beyond linguistic meaning, e.g., Discourse Representation Structure (DRS) (Kamp and Reyle, 1993) and lambda dependency-based compositional semantics (Liang, 2013). As a rule, these formal representations can be translated to a more intuitive, and sometimes simplified, graph-based representation. Nevertheless, a number of semantic parsers produce graph-based semantic representations without a direct logic-based correlate, e.g., Elementary Dependency Structures (EDS) (Oepen and Lønning, 2006), Abstract Meaning Representation (AMR) (Banarescu et al., 2013), Universal Conceptual Cognitive Annotation (UCCA) (Abend and Rappoport, 2013), Bilexical Semantic Dependencies (BSD) (Koller et al., 2019), and others (Kuhlmann and Oepen, 2016).

Distinctions between alternative representations can be drawn both in structural and content terms. As shown in the categorization of structures by Kuhlmann and Oepen (2016) and Koller et al. (2019), structural differences lead to many different types of graphs with diverging properties, i.e., connectivity, reentrancy, cycles, single versus multiple roots.

Abend and Rappoport (2017) and Koller et al. (2019) also showed that, in terms of contents, predicate structures occupy a central place in most representations, but differ in their coverage of other semantic aspects such as quantification and scope, event types, polarity, tense, coreference relations, semantic roles, grounding, and many others.

Differences can also been identified according to the level of abstraction. All semantic representations abstract away differences in realization derived from grammatical categories by reflecting predicate-argument structure directly, and also attempt to represent how word meanings contribute to sentence-level meaning. However, some representations adhere more closely to the principle of compositionality, resulting in parsers that mirror sentence structure and in particular syntax, e.g., LC-based formalisms and EDS, while others do away with syntactic considerations, e.g., AMR and UCCA. Another aspect related to abstraction is the anchoring of symbols onto sentence tokens. Logic-based representations and also some graph-based like BSD have a strong anchoring to tokens, while EDS and UCCA graphs contain nodes that map to either token substrings or multiple tokens. AMR is completely unanchored: its semantic analyses do not contain alignments of nodes to tokens.

One last important distinction is the support for cross-lingual representation. Formalisms tend to sacrifice universality to a certain degree in favor of detailed representation of semantic phenomena. Recent formalisms are being proposed that take the opposite approach and prescribe highly portable but coarse-grained representations, e.g., LC-based UDepLambda (Reddy et al., 2017) and UCCA.

Tools and resources: Following the general trend in NLP, statistical learning methods are the most prominent approach to learn parsing models. Manual annotation of semantic analysis can be a formidable task. This has led researchers to look for ways to reduce the cost of training models. One approach is to look for methods that require less training data. Researchers have experimented with deriving semantic representations from syntactic parses (Reddy et al., 2016) and applying weak supervision methods in the context of Question Answering (QA) (Pasupat and

Liang, 2015) and specific Knowledge Base (KB)s (Poon, 2013).

An alternative approach is to reduce the cost of manual annotation by adopting intuitive graph-based representations and effective annotation guidelines for the creation of large *sembanks*. These representations are receiving a lot of attention from researchers, as evidenced by recent semantic parsing events, i.e., various semantic parsing tasks in recent editions of SemEval (Oepen et al., 2014, 2015; May, 2016; May and Priyadarshi, 2017; Hershcovich et al., 2019) and the 2019 shared task on cross-framework meaning representation parsing (Oepen et al., 2019).

The vast majority of parsers available for semantic parsing have models for English only, or just a few more languages in the best of the cases. This is in part due to the lack of training data, but is also caused by the fact that most representations were not designed with cross-lingual representation in mind. As elsewhere in NLP, there is ongoing research on applying deep transfer methods to support multilingual semantic parsing (Duong et al., 2017).

Use for AS: The wide variety in semantic representations poses a significant challenge for developing summarization methods based on semantic parsing. This is one of the main reasons behind the scarcity of summarization systems leveraged by semantic analysis of the input text, another reason being the difficulty in generating natural language from meaning representations in the case of abstractive summarizers.

AMR constitutes an exception to this trend (Liu et al., 2015; Takase et al., 2016; Dohare and Karnick, 2017; Vilca and Cabezudo, 2017; Hardy and Vlachos, 2018), perhaps driven by the interest it has raised within the NLG community that has resulted in the development of linguistic generators (Flanigan et al., 2016; Song et al., 2016; Pourdamghani et al., 2016; Konstas et al., 2017; Ferreira et al., 2017; Schick, 2017; May and Priyadarshi, 2017).

Example: Figure 2.5 shows an AMR parse for the first sentence of our example. The resulting representation includes predicate-argument relations, indicated by ARG labels, and variables for entities and events, indi-

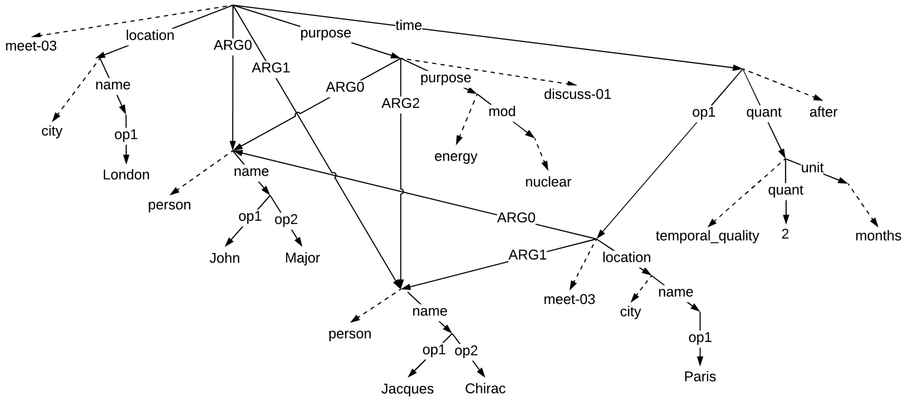


Figure 2.5: Output of the AMR analysis produced with the parser by Vanderwende et al. (2015).

cated by dashed arrows. The example also shows how the AMR formalism models intra-sentential coreference, NEs and temporal expressions. Instances of the types ‘person’ and ‘city’ in the figure are examples of how NEs are modeled in AMR, while the analysis of *two months after* illustrates the representation of temporal expressions.

2.6 Discourse Parsing

Purpose and scope: Discourse analysis goes beyond sentence boundaries to uncover how parts of a text are structured into a coherent whole. Discourse parsers are concerned with two types of relations, semantic relations holding between the information communicated in the text, e.g., cause or temporal precedence relations between assertions, beliefs or events, and pragmatic relations between text fragments, e.g., elaboration or justification relations indicating the use or purpose of one text fragment with respect to another. This distinction stems from theories of discourse, where the two categories are referred to by terms like ‘informational’ and ‘intentional’ (Moore and Pollack, 1992), or ‘subject-matter’ and ‘presentational' (Mann and Thompson, 1988). While some semantic parsers are also capable of detecting semantic relations, discourse parsers are not limited to relations within a single sentence.

Output representation: Research on discourse parsing can be divided into works on shallow discourse parsing and on full hierarchical parsing. Shallow parsers annotate binary discourse relations between text spans, e.g., "temporal precedence" or "pragmatic contrast", and do not seek to produce a comprehensive or connected representation of a whole text. Full hierarchical parsing, by contrast, attempts to establish a single structure covering a whole document.

Most shallow parsers adhere to the annotation scheme of the Penn Discourse TreeBank (PDTB) (Prasad et al., 2008, 2014) and produce unconnected representations that contain two types of binary relations, explicit and implicit. The first type are indicated by connectives belonging to certain grammatical classes and can have as arguments a multiple number of clauses or sentences. On the contrary, implicit relations are not marked by any word in the text and only hold between adjacent sentences. The set of relations available for annotation in PDTB is organized hierarchically, allowing annotation at different levels of granularity.

The structures produced by full parsers are dictated by theories of discourse structure such as Rhetorical Structure Theory (RST) (Mann and Thompson, 1988), Linguistic Discourse Model (LDM) (Polanyi, 1988), Relational Discourse Analysis (RDA) (Moser and Moore, 1996), amongst others. Despite the existence of alternative representations, most research on full parsing has targeted RST. RST parses are root, ordered trees where intermediate nodes correspond to n-ary RST relations and leaves to non-overlapping text spans, while edges indicate the role of each child in the parent relation –nucleus or satellite.

Tools and resources: Shallow discourse parsing has been largely supported by the PDTB (Prasad et al., 2008, 2014) and the smaller Chinese Discourse Treebank (CDTB) (Zhou and Xue, 2012). The 2015 and 2016 editions of the CoNLL shared task on shallow discourse parsing (Xue et al., 2015, 2016) asked participants to annotate newswire texts in English or Chinese with PDTB-based binary relations. Most participating systems trained models on the PDTB and CDTB treebanks.

The RST Discourse Treebank (RST-DT) (Carlson et al., 2001), containing 385 news articles in English from the Penn Treebank, has been the main resource for developing full discourse parsers (Morey et al., 2017). Similar RST-based treebanks have been developed for other languages, leading to some research on multilingual parsing (Søgaard et al., 2017; Muller et al., 2019). While discourse-annotated corpora have fostered significant research on full discourse parsing, the reduced size of the datasets available for training has led researchers to rely on other analysis tools to derive full discourse analysis, mostly syntactic parsers, e.g., (Li et al., 2014; Ji and Eisenstein, 2014; Wang et al., 2017). Even parsers that do not derive discourse parses explicitly from syntactic parses tend to use rich feature sets that require preprocessing the text with other tools (Soricut and Marcu, 2003; Sporleder and Lapata, 2005; Hernault et al., 2010).

Use for AS: Linguists have long argued that discourse structures can be used to identify informative parts of a text, as shown experimentally by Ono et al. (1994), Marcu (1998) and Marcu (2000), who used RST trees to support extractive summarization. Discourse-related information has been applied to several summarization systems –see the surveys on AS by Nenkova and McKeown (2011), Gambhir and Gupta (2017) an Yao et al. (2017). Louis et al. (2010) studied the correlation of discourse features with the selection of sentences in gold extractive summaries, distinguishing between structural features obtained from annotated RST trees in the RST-Bank, and features that indicate shallow discourse relations obtained from the PDTB. Their results indicated that structural discourse features derived from RST were strong predictors of relevance and that lexical similarity constitutes a very strong an easy to implement alternative to full discourse parsing.

These conclusions, coupled with the fact that performance of off-the-shelve RST parsers has been found to be insufficient for summarization purposes (Hirao et al., 2013), have led practitioners to dismiss discourse parsers in favor of cohesion and coherence devices indicating potential discourse relations (Chan, 2006; Abend and Rappoport, 2013; Ferreira et al., 2014; Gabriel et al., 2019) or to develop discourse parsing methods geared towards summarization (Gerani et al., 2014; Yoshida et al., 2014; Gerani et al., 2016). To the best of our knowledge, no shallow parsers have been applied for summarization purposes.

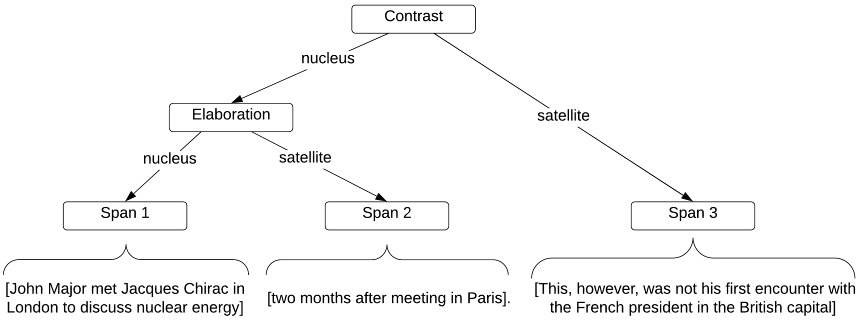

Contrast
nucleus
satellite
Elaboration
nucleus
satellite
Span 1
Span 2
Span 3
[John Major met Jacques Chirac in London to discuss nuclear energy]
[two months after meeting in Paris].
[This, however, was not his first encounter with the French president in the British capital]

Figure 2.6: Output of the CODRA RST parser.

Example: Figure 2.6 depicts the RST tree resulting from running the CODRA parser (Joty et al., 2015) on our example. The text is divided into three spans which constitute elementary discourse units, i.e., the leaves of the RST tree. An "elaboration" relation holds between the first two spans, indicating that the "satellite" span elaborates on the information given in the "nucleus" span. As indicated by the discourse marker *however*, the third span is linked to the first two through a "contrast" relation of which it is the satellite.

2.7 Relation Extraction

Purpose and scope: Relation extraction refers to uncovering conceptual relations in the text, a task that overlaps to a certain extent with other tasks like SRL and semantic and discourse parsing. While the first two tasks look at intra-sentence semantic relations as determined by linguistic form, relation extraction is not constrained by linguistic realization, literal meaning or sentence boundaries. Compared to discourse parsing, relation extraction is not concerned with structural, presentational or rhetorical aspects of text.

Research on extracting relations is split between the "closed IE" and "Open IE" paradigms. The first is addressed using knowledge-based, domain-specific methods for the extraction of predefined binary relations, either specified by the user or given as part of an ontology (Sarawagi, 2008; Konstantinova, 2014). A current trend in this paradigm is the large-scale extraction of relations based on learned lexico-syntactic patterns, e.g., (Etzioni et al., 2005; Carlson et al., 2010), and to target relations in popular Semantic Web (SW) ontologies (Barrière, 2016), in particular in DBPedia (Hellmann et al., 2013a; Lehmann et al., 2015) and YAGO (Mahdisoltani et al., 2015).

Breaking away from the closed IE paradigm, but keeping with the trend of large-scale, pattern-based relation extraction, Open IE (Yates et al., 2007; Banko et al., 2007) aims for open class, schema-less extraction of relational tuples from text. With the goal of efficiently addressing large-scale extraction of tuples, systems following this paradigm tend to eschew deep analysis of texts in favor of shallow processing, resulting in tuples where linguistic predicates act as labels indicating relations between text fragments. Unlike predicate-argument structures unveiled by semantic analysis tools, these tuples do not need to follow the syntactic realization of a predicate and its arguments in the text.

Output representation: Closed IE tools produce representations according to a target schema that defines types of relations, their participants and roles. Depending on the scope of the relation extraction task, participating entities may be left undefined, marked in the text or be resolved against entities or types of an ontology or KB. In the opposed Open IE paradigm, extracted relations are tuples of the form *(noun phrase, relation phrase, noun phrase)*, where all phrases are extracted verbatim from the text. The vast majority of approaches to relation extraction target binary relations, regardless of the paradigm they belong to.

Relation extraction tools do not attempt to find further relations between the relations they extract from the text and, consequently, produce non-comprehensive representations. These relations, however, aim to model the information conveyed in the text rather than describe any language or text-related aspects. This is the reason why we classified relation extraction as knowledge-oriented task.

Tools and resources: Closed IE is supported by a number of datasets such as the dataset created for the SemEval-2010 Task 8 (Hendrickx et al., 2010), the Freebase-annotated fragment of the New York Times Annotated Corpus (Riedel et al., 2010), the TAC Relation Extraction Dataset (Zhang et al., 2017) and FewRel (Han et al., 2018). The systems developed using these datasets are limited to extracting from English texts the binary relations annotated in each of them. The sets of relations annotated in each vary in size from 6 to over a hundred different types. Tools addressing closed IE tend to use supervised learning based on both lexical and syntactic cues.

Open IE systems rely mostly on conventional linguistic analysis like Part-of-Speech (PoS) tagging, chunking and dependency parsing to apply manual or learned relation extraction patterns (Niklaus et al., 2018). The Open IE community has applied their methods to a significant number of languages (Claro et al., 2019). Open IE systems produce vast amounts of tuples with a high level of redundancy caused by synonymous phrases and no clear specification of what constitutes a valid relational tuple. Recent efforts are being invested into creating large benchmarks corpora that supports the evaluation and training of Open IE models in a similar way to the datasets used for closed relation extraction.

Another line of research attempts to bring together the benefits of Open IE, i.e., open domain and large-scale extraction, and closed extraction, i.e., obtaining semantically defined relations. This is achieved either by integrating semantic types into pattern extraction (Nakashole et al., 2012), using entities obtained from EL as relation arguments (Dutta et al., 2013), or using Open IE as the basis for KB population (Soderland et al., 2013).

Use for AS: Open IE has been applied in a number of tasks related to AS, such as redundancy detection in extractive approaches to multi-document summarization (Christensen et al., 2013, 2014) and summary evaluation (Yang et al., 2016). Oliveira et al. (2016) look into the effectiveness of features based on Open IE tuples applied to producing single and multi-document extractive summaries. In addition, methods for pattern learning borrowed from research on relation extraction have been applied to headline generation (Alfonseca et al., 2013; Pighin et al., 2014).

A more direct application of Open IE to summarization is to take the tuples extracted by an Open IE system as the starting point for selecting contents, e.g., (Wang et al., 2016; Falke and Gurevych, 2017; Wities et al., 2017). This approach involves composing a summary out of the selected tuples, either by using them verbatim or by applying linguistic aggregation methods to fuse multiple tuples into a non-redundant sentence.

Example: Figure 2.7 shows the output of the Stanford Open IE system (Angeli et al., 2015) when applied to our example. As mentioned above, extracted relations often revolve around a verbal phrases. In our example, the phrases *met* and *was not* have led to the extraction of two relations. The use of chunkers or syntactic parsers means that relation arguments quite often align with syntactic arguments of the corresponding verbs, as is the case of the relations shown in Figure 2.7.

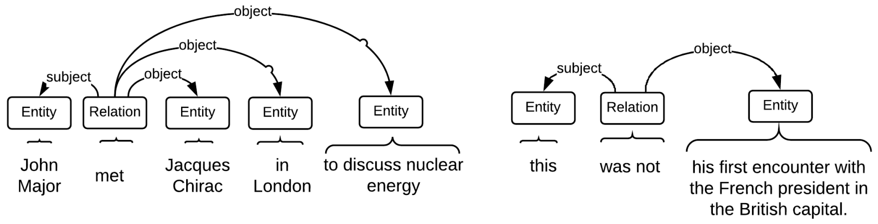


Figure 2.7: Output of the Open IE system by Angeli et al. (2015).

2.8 Knowledge Extraction Systems

Purpose and scope: KE systems use SW standards and Linked Data (LD) publishing principles to produce an integrated representation from the execution of text processing pipelines. These standards include vocabularies for expressing linguistic and conceptual terms, formats for serializing data in a machine-processable way and a specification of what Web protocols and mechanisms can be used to exchange information between text-processing services. The major challenge faced by these systems is to reconcile annotations produced by multiple analysis tools and integrate them in a single representation that is anchored to the text, encodes multiple layers of linguistic analysis and contains links to knowledge resources.

Output representation: KE systems seek to maximize the reuse of existing SW resources in order to boost interoperability with other services. These resources include linguistic ontologies that encode the annotations produced by NLP tools –tokens, lemmas, PoS tags, syntactic and semantic relations and others–, linguistic databases –SW versions of WordNet, FrameNet, etc.– and KBs –DBPedia, BabelNet, YAGO, etc.

Following LD principles, the resulting representations are serialized using Resource Description Framework (RDF) triples of the form *(subject, predicate, object)*. Thus, for instance, a triple may indicate the offset of a token in a text, a PoS tag associated with a particular token, the type of a dependency relation or the role filled by a predicate argument, and so on. Collections of RDF triples lend themselves naturally to be presented as labeled directed graphs where vertices correspond to the subjects and objects of the RDF triples, and edges to the predicates. It follows then that the output of KE systems can be seen as an RDF graph. A text planning method must be aware of the ontologies used for the creation of an RDF graph in order to be able to correctly interpret it; different KE systems may adopt different strategies or ontologies to encode the same linguistic data.

While RDF graphs produced by SW aim for comprehensive representation of the knowledge encoded in natural language texts, they include both linguistic information and extracted knowledge. The use of SW and LD standards should facilitate using automatic reasoning and other knowledge-based methods to infer additional knowledge from the output of these systems.

Tools and resources: Existing KE systems include LODifier (Augenstein et al., 2012), FRED (Gangemi et al., 2017) and PIKES (Corcoglioniti et al., 2016a). The first two use analysis pipelines that integrate a semantic parser with other tools addressing NER, EL, adjective analysis and coreference resolution separately. PIKES does not use a semantic parser. Instead, it integrates SRL annotations with syntactic parses. All these systems are limited to English texts.

The development of KE systems is supported by linguistic ontologies and LD datasets. These range from models for text annotations like NLP Interchange Format (NIF) (Hellmann et al., 2013b) and EARMARK (Peroni and Vitali, 2009), lexical models like LexInfo (Cimiano et al., 2011) and PreMOn (Corcoglioniti et al., 2016b), and mappings between lexical resources and other datasets like BabelNet (Navigli and Ponzetto, 2012), FrameBase (Rouces et al., 2017) and FrameSter (Gangemi et al., 2016).

Use for AS: to the best of our knowledge, KE systems have not been yet used for summarization purposes. A possible explanation for this situation is that these systems focus on knowledge representation and publishing aspects such as modeling linguistic information with SW standards and its integration with other data using LD principles, but have so far failed to contribute a useful representation for NLP purposes or to add substantial knowledge relative to the NLP tools and pipelines they rely on.

Example: A graph-based representation of the output of PIKES for our example is shown in Figure 2.8. Nodes in the graph correspond to variables indicating the entities, concepts and events referred in the sentence, and edges indicate roles. Types associated to variables are indicated in red. Thus, predicative words can be labeled with PropBank senses and FrameNet frames, e.g., ‘meet.03’ and ‘Come_together’ for *meet*, ‘discuss.01' and 'Discussion' for *discuss*. Their arguments are connected with roles taken from either of these resources, e.g., "Party_1", "loc", or with generic role labels if a suitable role could not be found, e.g., "mod". The system also assigns NE types, e.g., 'Organization' to "energy_2", and types taken from ontologies, e.g., 'SocialInteraction' to *meet*, *discuss* and *meeting*. Whenever the system determines that an entity referred from the text is found in DBPedia, it introduces a reference to it, e.g., 'dbpedia:John_Major'. The graph also shows how PIKES models temporal and multi-word expressions.

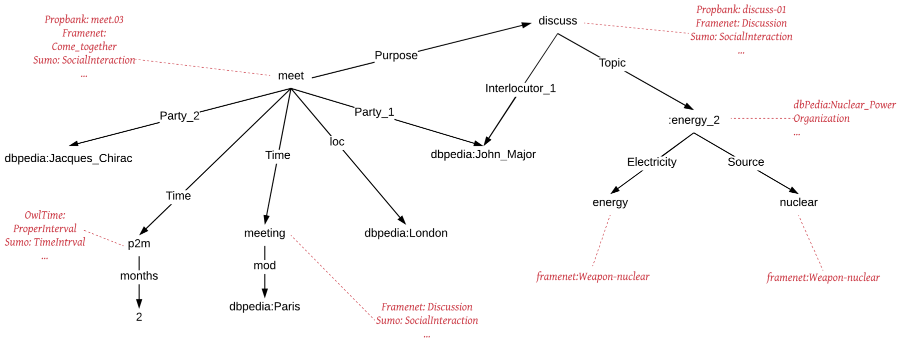


Figure 2.8: Output of the PIKES KE system for the first sentence of our example.

The name *San Francisco* has been correctly resolved against the DBpedia entry for the city. A FrameNet frame and roles have been assigned to the verb *marry* and its arguments, while no frame has been assigned to *meet*. Both events have been linked with a temporal relation named after the preposition *after*. While pronouns are correctly resolved to their nominal antecedents, resulting in a single node for each chain, event coreference between verb *married* in the first sentence and *marriage* in the second is not resolved. This reflects the current status of coreference as explained in Section 2.1. The diagram also shows how complex nominal phrases with compositional meaning such as *first marriage* are modeled using quality and taxonomic relations.

2.9 Towards a Common Representation for Text Planning

Ideally, text planning should start from an abstract and complete representation of the contents in a natural language text. Obtaining such a representation may be possible in some scenarios, but unfortunately it is not realistic to expect this for texts in any language and belonging to any domain. Limitations in SoA tools often impair analysis in terms of scope (leading to non-comprehensive representations), abstraction, multilingual support and cross-domain portability.

Wishing to strike a balance between these factors, we look at several trends that are common across the tasks surveyed in this chapter. This trends will motivate our choice of representation and the means to obtain it from natural language, both described in Chapter 4.

2.9.1 The Trade-off Between Scope and Abstraction

Approaches to knowledge-oriented tasks tend to sacrifice scope in favor of achieving abstract, non-linguistic representations. Thus, NER and EL tools detect entities but not relations holding between them. NER and closed relation extraction are restricted to pre-defined sets of entity and relation types. In cases where researchers seek to widen the scope, they do so by sacrificing depth of analysis, as is the case of the text-based tuples produced by Open IE tools. KE systems aim for representations that are both comprehensive and possess a deep level of analysis. They do so by integrating the output of multiple tools but, as we have seen, the few such systems that have been developed are limited to English and have seen no adoption by the NLP community yet.

While semantic parses reviewed in Section 2.5 are geared towards linguistic meaning, they are the most abstract comprehensive representations of meaning obtainable using off-the-shelf tools. Formalisms like AMR cover various aspects of meaning, but their analyses are limited by sentence boundaries. Full discourse parsers produce document-level representations, but at the expense of depth; discourse relations in parse trees hold between arbitrary text spans not necessarily aligned with specific propositions, entities or word meanings. This is, in part, the result of theories of discourse not sharing an agreement towards what constitutes the basic atomic unit of discourse structures, or elementary discourse units in RST terminology.

Of all tasks producing comprehensive representations, syntactic parsing as a whole offers the greatest multilingual support. Syntactic trees do not model meaning as semantic graphs do and, unlike discourse parsers, are limited to sentence boundaries. Nevertheless, the fact that modern dependency parsers can accommodate or even detect MWEs (Candito and Constant, 2014; de Marneffe et al., 2014; Nivre et al., 2016) facilitates merging dependency trees with single and multi-word annotations produced by coreference, NER, WSD and EL tools. As we will see in Chapters 4 and 6, we adopt this approach to create document-level semantic representations for our experiments.

2.9.2 Cross-lingual and Cross-domain Text Analysis

Deep learning methods are having a profound impact on NLU, an impact that mirrors the impact of seq2seq models for summarization. In both cases, neural networks have become nearly-standard approaches and new research is looking into transfer methods to adapt pretrained language models to tasks, domains and languages for which training data is scarce. As in the case of summarization, it is yet unclear whether transfer methods applied to text analysis tasks can match the performance of dedicated models trained specifically for one task and in one language. Furthermore, while syntax has been shown to be implicitly captured in pretrained models, these models fail to capture long-range relations and semantic information (Ruder et al., 2019). This casts a shadow on the prospect of having adaptable neural models capable of producing comprehensive semantic analysis for multiple languages in the near future.

At the time of writing this thesis, many publicly available NLU and IE tools still have limited multilingual support and have been evaluated only on texts belonging to a few domains, mostly the journalistic domain. Since one of our research goals is to develop language and domain-independent methods for planning summaries, it is important not to rely on a starting representation of contents that requires analysis tools that cannot be easily applied to multiple languages or types of text. In the light of what has been said in the previous sections, this rules out coreference resolution, SRL, semantic parsing, discourse parsing, closed relation extraction and knowledge extraction. In contrast, syntactic parsing, NER and lexical meaning disambiguation have a better outlook when it comes to cross-lingual support; due to their nature they benefit more easily from transfer knowledge or large multilingual lexical resources.

2.9.3 Graph-based Representations

Another common trait of the representations produced by the tools analyzed in this chapter, and in particular of those producing comprehensive representations, is that they are either graphs or can be represented as such. This is indeed the case of syntactic trees, semantic graphs, hierarchical discourse trees, and RDF graphs. Despite this structural similarity, these representations are very different in terms of contents. On the one hand, vertices can be words, text-spans, word senses, resolved entities or linguistic artifacts belonging to different levels of linguistic analysis. On the other hand, edges tend to be linguistic in nature, but belong to different levels of linguistic analysis depending on the task, as mentioned in Section 2.7, tools capable of extracting non-linguistic relations do not produce comprehensive representations.

Given that, as we will see in Chapter 3, graphs are also widely used for summarization purposes, we adopt a graph-based representation for our approach. Semantically-oriented text planning requires that this graph representation is not based on more than text fragments or language-specific descriptions, as is the case of many of the representations that have been discussed so far. For this reason, we also assume that our representation is semantic in the sense that it is based on meanings –either word senses or linked entities. This decision is supported by the availability of large multilingual lexical resources that contain meanings covering multiple areas of knowledge, such as those mentioned in Section 2.2.

While trees are sufficient for some linguistic representations such as syntactic trees, graphs are needed for representing semantic phenomena like shared arguments, realized with reentrancies, or multiple top-level relations that require structures with multiple roots. Consequently our representation for text planning will be a directed graph rather than a tree, as graphs are needed to accommodate representations of semantic phenomena without loss of generality.

Chapter 3

STATE OF THE ART

AS is a vast and diverse field with a long history of research. Trying to cover the whole field would be a formidable undertaking. Consequently, we focus on specific areas of research that are relevant to our approach and research goals. Recall from Section 1.3 that one of our goals is to follow a text planning strategy for summarization that produces useful insights for downstream tasks, a goal that implies building an interpretable representation of the contents in the input text and their assessment for inclusion in the summary. Recall also that we wish to rely on unsupervised methods only, and that we want to exploit similarities between graph-based ranking methods applied in the field of summarization and for disambiguation tasks like WSD and EL.

In accordance with these goals, we will focus our review on works that apply unsupervised graph ranking methods and, more generally, to works that adopt graph representations. Another goal set in the introduction to this thesis is to use a domain-independent and language-independent strategy for planning. For this reason, we will also pay attention to approaches to summarization that instantiate representations closer to meaning and which are not dependent on the language of the input text.

This chapter is organized as follows. Section 3.1 looks at the graph-based ranking paradigm, an approach that has enjoyed widespread adoption in extractive summarization works and that is largely supported by unsupervised centrality algorithms. In Section 3.2, we look at works that create graph-based representations where vertices correspond to words in the input text and edges are established using textual or syntactic relations. The works surveyed in Section 3.3 use deep analysis, i.e., semantic or discourse analysis, to construct graphs from which summary contents are selected, while those in Section 3.4 experiment with entities linked to KBs. Given their prominence in recent years, no review of the state of the art would be complete without a reference to the current state of affairs in neural summarization, which is the topic of Section 3.5. We close the chapter in Section 3.6 with a discussion on how our approach compares to the works surveyed.

3.1 Graph-based Ranking

Graph-based ranking methods have a long history in extractive summarization. They involve creating a graph representation where vertices correspond to contents in a document or documents to summarize and edges indicate similarity relations between these contents. Nodes in the graph are then ranked according to their salience in the graph, usually through the application of centrality algorithms like PageRank (Page et al., 1999) or Hyperlink Induced Topic Search (HITS) (Kleinberg, 1999). Graph-based ranking has been a popular approach in extractive summarization, the resulting ranks being used to select sentences.

The TextRank (Mihalcea and Tarau, 2004) and LexRank (Erkan and Radev, 2004) extractive summarizers are credited with kick-starting the approach. Both systems model the input text as an undirected graph where vertices correspond to sentences and edges indicate similarity between pairs of sentences. Differences in approach between the two works are representative of the design choices faced by researchers when adopting graph-based ranking for summarization purposes. A first important distinction lies in the method followed to calculate similitude between pairs of sentences. A second distinction is the design of the graph on which the ranking algorithm operates.

Looking at the first distinction, TextRank calculates pairwise similarity values using normalized lexical overlap between sentences, while LexRank uses cosine similarity between vectors containing Inverse Document Frequency (IDF) values for words in a sentence. Researchers have tried other methods for calculating similarity values. Chali and Joty (2008), for instance, use tree kernels to compare sentences via their deep syntactic trees, while Biased LexRank (Otterbacher et al., 2009) applies sentence-level unigram language models to compute non-symmetric similarity values. Yin and Pei (2015) use neural networks to learn vector-based sentence representations that can be used to calculate similarity using cosine distance.

In some cases, similarity thresholds are used to limit the number of edges in the graph, leading to sparser graphs. Erkan and Radev (2004) experimented with similarity thresholds in LexRank to produce a sparser ranking graph, as the density of the graph (understood as the average degree of its vertices) influences the speed with which the ranking algorithm converges to a stationary distribution. Otterbacher et al. (2009) limit outgoing edges in Biased LexRank to the 20 most similar sentences, a move that speeds up convergence of the ranking algorithm and improves the quality of the resulting summaries.

Edge weights can be tuned using metrics that assess sentences on their own, rather than comparing them to other sentences. In Biased LexRank, the authors use sentence-to-topic similarity values that act as prior relevance values for sentences. In URANK (Wan, 2010), edge weights are biased with priors that reward sentences according to their position in a document. Wan (2010) adds edges to the graph according to multiple similarity functions that reflect different types of relations, e.g., similarity across sentences in the same document, sentences in different summaries.

Some approaches introduce special nodes in the graph to focus summarization on certain topics or queries. In TGRAPH (Parveen et al., 2015), for instance, a bipartite graph is created where the vertex set is classified into sentences and topics, and weighted edges indicate lexical overlap between pairs of one sentence and one topic. After determining the topics of a document using topic modeling techniques, the authors apply HITS to obtain a rank of sentences according to their overall similarity to the document topics. Li and Li (2014) also create a graph containing nodes for topics and establish different types of relations across topics and sentences, but use the topic models directly to derive edge weights.

Ranking graphs may also include parts of sentences as vertices. In GRAPHSUM (Baralis et al., 2013), vertices in the ranking graph are frequent sets of content words mined from the sentences to rank, while edges indicate both positive and negative numerical correlations between sets observed in the document. SSRank (Yan and Wan, 2014) applies SRL to build a graph containing not only sentences but also verbs, roles and phrases obtained from an SRL tool. The authors also experiment with word clustering and dependency parsing to reduce the number of predicate-nodes and therefore the size of the ranking graph.

Some systems apply a re-ranking step designed to reduce redundancy between extracted sentences. These systems usually use the similarity metric measure used for the ranking again to exclude candidate sentences that have high similarity to any of the sentences already in the summary (Wan and Yang, 2006; Otterbacher et al., 2009; Yan and Wan, 2014). Bipartite graphs that include vertices indicating entities or topics have also been used to re-rank extractive summaries according to their local coherence (Guinaudeau and Strube, 2013; Parveen et al., 2015).

3.2 Word Graphs

Graphs have been used as an intermediate representation of contents in a large number of works beyond those adopting graph ranking methods. A prominent strategy when producing paraphrased summaries involves creating a "word graph" where the vertex set corresponds to words found in the input text and edges indicate word-to-word relations. Works following this strategy generally approach summarization by grouping sentences into clusters of related sentences and then instantiate a graph representation for each cluster. Summary sentences are produced from the cluster graphs by extracting a salient subgraph or removing redundant parts. During the graph creation stage, words are merged into a single node if certain conditions are met such as the words being synonymous, sharing a direct hypernym, having an overlap in their context, etc.

Summarization systems using this kind of graphs can be divided into those where the graph has syntactic dependencies as edges, taken from the parse trees of individual sentences, and those where edges are established between pairs of vertices if the corresponding words are found together in the input text. Barzilay and McKeown (2005) is an early example of the first group of systems. The authors address multi-document summarization using sentence fusion methods borrowed from the field of sentence paraphrasis (Barzilay and Lee, 2003). They identify centroid sentences across multiple documents and augment their dependency trees with subtrees of other sentences in the cluster that are found to be lexically and syntactically similar. The resulting dependency graph is subsequently pruned using hand-crafted rules to produce a syntactically valid tree, which is then linearized using a generate-and-rank method based on a trigram language model.

Filippova and Strube (2008) adopt a simpler sentence fusion mechanism. They merge word-nodes in the dependency trees of a group of sentences whenever (i) they are content words, (ii) have the same or synonymous lemmas, and (iii) share the same PoS tag. Starting from the resulting dependency graph, subtree extraction is encoded as an optimization problem where the objective function incorporates both syntactic and salience criteria. The former is based on conditional probabilities of dependent words given their governors and a set of constraints that enforce the extraction of well-formed dependency trees, while the latter is based on frequencies of words in the documents and in a corpus. Trees extracted from merged dependency graphs are rendered into a summary sentence by applying the same method as Barzilay and McKeown (2005).

Elsner and Santhanam (2011) build on Filippova and Strube (2008)’s method, but rather than deciding the alignments deterministically prior to extracting a subtree, they approach alignment and subtree extraction as a joint optimization task using supervised learning methods. Cheung and Penn (2014), on the other hand, address alignment and extraction separately, but add an additional intermediate step where the dependency graph for a cluster of similar sentences is expanded with subtrees from dependency parses of other sentences in the input document. This is done by searching for dependencies in sentences outside the cluster where the governor shares lemma and PoS with one of the words in the dependency graph. Once found, the governor along its governed subtree is added to the graph if it helps in maximizing a salience heuristic. In order to promote grammatically correct trees when addressing extraction, Elsner and Santhanam (2011) and Cheung and Penn (2014) use the same probabilistic lexicalized approach based on dependency-annotated corpora as in Filippova and Strube (2008). Filippova (2010) and Cheung and Penn (2014) also include additional constraints designed to prevent semantically unsound sentences, e.g., forbidding that lexical items in a hypernymy/hyponymy relation are members of the same coordination in the extracted tree.

All summarizers based on dependency graphs require applying NLG methods to render the extracted trees into natural language. Most systems mentioned so far apply a linearization strategy based on language models to transform extracted trees into a sequence of words. Researchers have experimented with graphs that incorporate word order in their edges instead of dependency relations. Summaries are then generated by finding shortest paths in the word graphs and ranking them according to heuristics designed to favor grammaticality and salience. These heuristics are similar to those used to extract trees from dependency graphs. Since the extracted graphs are already sorted sequences of words, no linearization step is needed.

Filippova (2010) and Filippova and Altun (2013) use just a tokenizer and a PoS tagger to summarize clusters of multiple related sentences into a single short, informative sentence. Thadani and McKeown (2013) approach sentence fusion with an Integer Linear Programming (ILP) formulation of word graphs and supervised learning, but incorporate features obtained from dependency trees. Sentence fusion based on word graphs has also been applied to summarization of written conversations in response to queries (Mehdad et al., 2014) and to multi-document summarization (Banerjee et al., 2016). While all the aforementioned works use shortest paths to extract sentences, paths can be combined in order to create more complex sentences (Ganesan et al., 2010).

3.3 Deep Graphs

The graphs discussed in the previous section are obtained using shallow linguistic analysis. Researchers have also experimented with deeper levels of analysis to obtain abstract graph-based representations closer to meaning. These abstract graphs are often constructed from the output of semantic analysis tools such as SRL, Open IE, semantic and discourse parsers.

As mentioned in Section 3.1, SSRank (Yan and Wan, 2014) creates a ranking graph that contains sentences together with phrases and predicates obtained from an SRL tool. The purpose of these phrases and predicates is to contribute shallow semantic information to the ranking of sentences. Khan et al. (2015) and Khan et al. (2018) go a step further and create a ranking graph exclusively from predicate-argument tuples extracted from multiple documents. The resulting tuples are clustered using a distance metric based on tree edit distance and WordNet-based similarity, and the tuples in each cluster are ranked using a genetic algorithm. Elements from the top ranked tuples are then combined into sentences and realized into grammatical English text using the Simple NLG generator (Gatt and Reiter, 2009).

Research on pattern extraction for paraphrastic summarization has targeted the extraction of graphs containing semantic information. Works following this approach usually extract patterns from dependency trees (Alfonseca et al., 2013; Pighin et al., 2014) or syntactically-analyzed Open IE tuples (Zhang and Weld, 2013; Li et al., 2015; Wang et al., 2016). With the goal of making the patterns more abstract, words are replaced by semantic types obtained from NER and, in the case of Li et al. (2015), also with FrameNet frames obtained from SRL. Paraphrasis are generated by finding the best-matching pattern and replacing its semantic types with actual words in the text. Searching for the best pattern can be done with a supervised model (Alfonseca et al., 2013; Pighin et al., 2014) or pre-trained word embeddings (Li et al., 2015).

A few summarization systems have adopted AMR as an intermediate semantic representation. Liu et al. (2015) use coreference resolution to merge sentence-level AMR parses into document-level graphs. Content selection is approached as a subgraph extraction task based on node and edge weights that are estimated by a relevance model trained on an AMR-annotated summarization dataset –the proxy section of the AMR Bank (Knight et al., 2014). The extraction procedure also uses a set of constraints to ensure that the resulting graphs are structurally correct AMR graphs. Lack of AMR generators at the time of writing forced the authors to evaluate their system using unigram Recall-Oriented Understudy for Gisting Evaluation (ROUGE) between summaries in the AMR Bank and a bag-of-words representation of the extracted graphs. More recently, AMR-based summarization systems have incorporated linguistic generators to produce abstractive summaries, e.g., AMR-to-text (Dohare and Karnick, 2017) or SimpleNLG (Vilca and Cabezudo, 2017).

AMR graphs do not encode information like tense and grammatical number, leading to a mismatch between the summaries text and the information conveyed morphologically in the source texts. Taking the AMR graphs produced by Liu et al. (2015) as a starting point, Hardy and Vlachos (2018) address this problem by using an AMR-to-text model guided by the source document and a seq2seq model.

Discourse-based structures are another representation used to address summarization in a number of works. RST trees have been used to select parts of a document in a way that preserves its rhetorical structure and guarantees coherence in the summary –see for instance the works surveyed by Nenkova and McKeown (2011) and Gambhir and Gupta (2017). While many of these works use discourse parses along other types of linguistic information, a number of systems operate directly on discourse-based representations to produce a coherent summary. The latter group of systems operate under the assumption that RST trees encode not only document-level structure, but also information about the relative relevance of contents, reflected by the nucleus versus satellite distinction and by the depth of the tree.

One strategy when adopting a discourse-based representation is to transform RST trees into dependency-based discourse trees where all nodes correspond to Elementary Discourse Unit (EDU)s. This the case of Hirao et al. (2013), Yoshida et al. (2014) and Hirao et al. (2015), who apply optimization methods to extract a subtree from the dependency-based discourse tree such that it maximizes a relevance function for EDUs. Gerani et al. (2014) and Gerani et al. (2016) follow a different strategy and produce aspect-based summaries of product reviews using a graph where vertices correspond to words or phrases indicating aspects of the product, and relations are obtained from the RST parses of the documents. The nodes are then ranked using a centrality algorithm biased on the relative position of aspect words in the text and their depth in the RST tree. The resulting ranks are used to extract a maximum spanning tree, which is realized as a natural language summary using a hybrid rule and template-based approach.

3.4 Linked Entities

Meaning-based summarization is a relatively unexplored field. As mentioned in Section 2.2, a number of works use WordNet or WSD to improve sentence similarity calculations in the context of extractive summarization. Named entities, commonly seen as indicators of the main topics of a text, have been used as features for various methods for sentence extraction such as classifiers. For a number of years the use of lexical resources, WSD and NER have been overshadowed by methods capable of building implicit representations of the meaning of a text, such as Latent Semantic Analysis (LSA). Despite this, advances in EL and lexical databases have motivated some researchers to experiment with linked entities and summarization tasks.

Dunietz and Gillick (2014) introduced the task of Entity Salience (ES) where entities are ranked according to how prominent they are in a document. The authors rank entities linked to Freebase using a classifier trained on a large number of examples obtained from the New York Times Annotated Corpus (Sandhaus, 2008). Examples of salient entities are obtained by running an EL tool on the corpus and observing cases of entities that occur both in a document and in its summary. At test time, their classifier predicts entity salience using features like number of mentions of the entity in a document or the position of the first mention of an entity. Following the intuition that knowing how entities relate to each other is useful for ES purposes, the authors experiment with an additional feature calculated from the ranking of entities in a document. The ranking is obtained by running a weighted version of the PageRank algorithm on a graph where nodes indicate entities and directed edges represent the probability of an entity co-occurring with another entity, the probabilities being calculated using counts obtained from the training corpus. Dunietz and Gillick (2014) report that the ranking-based feature does not result in a big improvement for their approach and speculate that this might be due to the performance of the EL tool.

The SEL system (Trani et al., 2016) jointly addresses EL and ES against Wikipedia, and its results are applied to extractive summarization. SEL uses a discrete set of four labels that grade entities from irrelevant to very relevant, and applies two supervised classifiers, one that establishes the prior probability of a candidate entity being relevant for a document and another one that predicts the salience label. The first classifier is trained on the EL dataset contributed by Hoffart et al. (2011), while the second classifier is trained using a portion of WikiNews where links to Wikipedia entities are manually annotated with saliency labels. This second classifier uses the confidence score of the first classifier as a feature, along with features extracted from the Wikipedia graph formed by following hyperlinks between pages, one of the features being a centrality value. The first classifier is used at test time to prune the set of candidates. The authors address sentence-based extractive summarization with a third classifier trained on Document Understanding Conference (DUC) datasets that uses features based on salient entities detected by SEL.

Amplayo et al. (2018) incorporate links to Wikipedia pages, including ambiguous links to disambiguation pages, in the training of a seq2seq model for abstractive summarization. They propose a special encoder that takes the sequence of Wikipedia page ids produced by an EL tool in the order in which they appear in the text, and uses sense embeddings –pre-trained vectors for Wikipedia pages– to produce a vector-based representation of a document topic. Since the topic may contain ambiguous entities, the authors propose two encoders designed to disambiguate entities, one that combines the vectors of all entities in a document and another that only combines neighboring entities. These encoders reflect local and global strategies for disambiguation. During training, the entity encoder informs the seq2seq summarizer on what is the topic of each document. Being a neural system, the summarization system by Amplayo et al. (2018) exposes neither the results of disambiguation nor the relevance of entities for each document.

3.5 Sequence to Sequence Models

The first applications of deep learning methods for summarization purposes consisted in computing similarity between sentences for extractive summarization. This involved using word and sentence embeddings to compare text fragments, leaving the actual summarization task for other methods such as optimization or graph centrality (Kågebäck et al., 2014; Yin and Pei, 2015). Inspired by the successful application of seq2seq neural architectures to machine translation, researchers began applying this paradigm to learn models capable of addressing the whole summarization task.

An early example is Rush et al. (2015), who train a seq2seq paraphrasis model for sentence compression. Their approach uses a generative language model and a summary model trained on four million pairs of sentences taken from the Gigaword corpus (Napoles et al., 2012), where each pair consists of a title and the first sentence of the corresponding article. The combined model produces state-or-the-art results when evaluated with ROUGE recall scores using Gigaword and the 2004 DUC dataset (Over et al., 2007). The model trained by Rush et al. (2015) suffers from two problems. First, rare or unseen words, e.g., named entities, tend to be excluded from the summaries even when they are salient in a document. Second, when applied to generate longer summaries the model often repeats phrases.

Cheng and Lapata (2016) adapt the neural architecture of the seq2seq paradigm to perform classification of text fragments for extractive summarization of whole documents. They train a model for extraction of sentences and another one for extraction of words. Being extractive, their models do not need to produce words outside the vocabulary of the input document and therefore are not affected by problems associated with generating infrequent words. In their sentence model, the redundancy problem is addressed by using a recursive neural network that takes into account previous extractions before generating a new one. Other neural extractive systems that have appeared since that use similar approaches, e.g. the SuMMaRuNNer system by Nallapati et al. (2017).

Another line of research has extended Rush et al. (2015) work to paraphrase longer texts. Some works have attempted to ameliorate the problems with low-frequency words by adopting neural architectures capable of extracting words from the input under certain circumstances (Nallapati et al., 2016; See et al., 2017; Tan et al., 2017). Repetition in longer texts has been addressed in a similar way as Cheng and Lapata (2016) by adopting methods based on recursive networks to remember previous choices of summary words (See et al., 2017; Paulus et al., 2018). Such neural architectures have also been proposed that, when generating summary words, avoid redundancy by forcing the model to pay attention not only to a specific part of a document but also to other parts (Chen et al., 2016). This mechanism has also been applied to improve salience, informativeness and coherence (Tan et al., 2017; Çelikyilmaz et al., 2018; Gabriel et al., 2019).

While some models are trained directly on the raw tokens of document-summary pairs (Rush et al., 2015; Chen et al., 2016; Cheng and Lapata, 2016; Nallapati et al., 2017; See et al., 2017) others use features derived from linguistic analysis. For instance, Nallapati et al. (2016) use PoS, NEs, and TF and IDF word statistics to train a headline generation model on the Gigaword corpus. Takase et al. (2016) present a direct extension of the work by Rush et al. (2015) where the same neural network is trained with encoded AMR parses obtained from the parser by Wang et al. (2015). Both Nallapati et al. (2016) and Takase et al. (2016) report better results than Rush et al. (2015) when running evaluation on the DUC-2004 and Gigaword datasets.

Research on summarization has just begun experimenting with transfer learning methods. Liu et al. (2018) use the transformer (Vaswani et al., 2017) non-recursive neural architecture to train a two-stage system comprising an extractive model, the results of which are then fed to an abstractive model. Using an extractive model to limit the input to the paraphrasis stage allows the system to handle large inputs and outputs. In their experiments, lead sections of Wikipedia pages are generated from multiple articles in Wikipedia and pages crawled from the web. Liu and Lapata (2019) use the BERT language model (Devlin et al., 2019) as an encoder for input texts and apply it for both extractive and abstractive summarization using the CNN/Daily Mail (Hermann et al., 2015), the NYT Annotated Corpus (Riedel et al., 2010) and the recently published XSum dataset (Narayan et al., 2018). Interestingly, they obtain best results after fine-tunning their BERT-based encoder twice, first for extractive summarization and then to produce paraphrased summaries.

3.6 Relation of the State of the Art to the Thesis

Our system draws inspiration from a number of works covered in this chapter. As mentioned in Chapter 1 and explained in detail in Chapter 5, the strategy followed in this thesis for planning summaries involves ranking candidate lexical meanings in order to instantiate a semantically-oriented graph representation. This graph and the results of the ranking are then used to rank individual words in the document to be summarized. Both rankings are addressed using the same method, inspired by Biased LexRank (Otterbacher et al., 2009) and URANK (Wan, 2010). In these works, sentences are ranked using a graph algorithm based on similarity across sentences, the pair-wise similarity values being tuned with heuristics that express prior values for individual sentences. We find this bias mechanism useful to combine local and global context in our ranking of meanings, and to transfer the resulting ranks to the subsequent ranking of words. The latter is achieved by using meaning ranks as priors for words.

Most of the works surveyed in Section 3.2 and Section 3.3 that construct graphs as intermediate representations follow a similar sequence of steps. First, a graph is built, often by merging sentence-level representations such syntactic or semantic parses. This is usually followed by an assessment step where a weighting function indicating relevance of contents is associated with the graph. These weights are the basis for a subgraph extraction or punning strategy applied to obtain a subset of highly relevant contents. The selected contents are then ordered, redundancies removed and, if necessary, rendered into natural language through the application of NLG. Our own graph-based approach also follows this sequence of steps and includes separate steps for graph creation, assessment, extraction, redundancy removal and ordering. Using NLG terminology, assessment and extraction correspond to content selection, while redundancy removal and ordering are considered discourse structuring tasks.

Our graph representation, presented in detail in Chapter 4, can be related to word graphs in the sense that the vertices in the graph are anchored to words in the text. Unlike most works that adopt this kind of representation, however, our vertices are also associated with lexical meanings. This meaning-oriented representation allows us to apply a language-independent strategy for text planning. Edges in the graph can accommodate a wide range of relations that extracted using analysis tools or alternatively indicate when two vertices are associated with the same lexical meaning.

In the experiments described in Chapter 6, we use a UD-based syntactic parser given that dependency parsing is a firmly established field with many tools and models available for multiple languages. This places our approach closer to systems using dependency graphs. Nevertheless, syntactic dependencies could be replaced with other types of relations between words obtained from analysis tools without substantially altering our approach to planning summaries. This is possible because our text planning strategy is largely oblivious to the nature of the edges in the graph, which allows it to operate independently of the specific tools used to instantiate the graph.

Relations produced by text analysis tools can be classified into relations holding between words or MWEs, and relations between text spans. The former include order in the text, coreference relations, syntactic and semantic dependencies, while the latter include SRL, Open IE, rhetorical and shallow discourse relations. A third group of relations have no direct correspondence with the text, as is the case of conceptual relations based on ontologies or obtained through reasoning. Our approach to text planning assumes that relations of the first type are produced during analysis, in a similar fashion to the works referenced in Section 3.2 and in Section 3.3. This choice is motivated by the fact that word to word relations can relate lexical meanings in our graph more easily than other types of relations.

While our text planning strategy ignores the nature of the edges in the graph, this is not possible when addressing linguistic surface realization. As seen in Section 3.3, deeper relations between lexical units require more sophisticated NLG strategies to produce a non-extractive summary.

Thus, language models and a few constraints are enough for word graphs based on precedence relations between words in the text, while dependency graphs require a linearization step to determine word order. Semantic parsing, on the other hand, requires a complete surface realization strategy, be it a statistical generator, a rule-based one or a combination of rules and templates. This thesis describes experiments with extractive summaries only, but linguistic generation should be considered if our approach was applied to the production of paraphrasis or abstracts.

Close to our approach, the works described in Section 3.4 involve addressing disambiguation issues and calculating the salience of meanings. Compared to Dunietz and Gillick (2014), we do not start from the output of an EL tool. Instead, we take on board the disambiguation of both word senses and named entities. While they estimate entity relatedness from an annotated corpus, we use pre-trained embeddings to compare meanings. Trani et al. (2016) rank entities, but they target EL and adopt extractive summarization as a downstream task to show the usefulness of their approach, while we do the opposite and adopt WSD and EL as downstream applications of our summarization approach. Other important differences are that our approach is completely unsupervised and therefore does not depend on multiple datasets and feature sets, and that instead of predicting salience labels, we produce a full rank of candidate entities in a document, each rank associated with a numerical value. The neural summarization system by Amplayo et al. (2018) addresses disambiguation and salience determination for linked entities, but does not expose the results of these tasks in an interpretable way. The sense embeddings they use cover 76% of the extracted entities in their experimental data, leaving all other entities non-comparable. In contrast, we use sentence embeddings calculated on lexicalizations and glosses, which means that we can compare all meanings handled by the system.

Chapter 4

A REPRESENTATION FOR TEXT PLANNING

In this chapter, we propose an intermediate representation for the task of planning summaries. This representation follows an analysis phase conducted with SoA tools and serves as the starting point for the text planning approach presented in Chapter 5. Considering the scope and goals of our research described in Chapter 1, our representation should meet the following requirements:

- Support semantically oriented text planning methods that operate independently of the language and domain of the input text,
- be versatile and simple enough to allow instantiation from a variety of analysis tools, and
- support both extractive and abstractive summarization.

The first design choice for our representation is to base it on meanings expressed by lexical units, a choice motivated by the availability of large lexical knowledge bases covering multiple languages and domains (Färber et al., 2015; Färber et al., 2018). This decision is key to support a text planning strategy that is independent from both the language of the input text and the domain or general area of knowledge it belongs to.

The second design choice is to use a graph-based representation. As seen in Section 2.9, trees or more general graphs constitute the standard output for many analysis tools, including syntactic, semantic and discourse parsers. Graphs can also integrate additional types of analysis. Thus, many parsers and knowledge extraction systems produce graphs that incorporate named entities, word senses or coreference relations. It follows then that graphs constitute a sensible choice for a representation that can be instantiated from a wide range of alternative analysis tools. Adopting a graph-based representation also suits well our goal of exploiting similarities between graph-based ranking methods used for disambiguation and summarization tasks.

In order to support extractive summarization, our representation should be anchored to words in the input text, so that the results of relevance assessment on the graph can be transferred to the text for the extraction of fragments. In contrast, abstractive summarization requires that there is a semantic representation from which a summary can be generated anew. We attempt to satisfy both requirements by requiring that the vertices in our graph can be mapped both to words in the texts and to meanings in a lexical KB.

Taking all the above into account, we adopt as a representation for text planning a graph that incorporates vertices and relations coming from the analysis of the text, and two additional functions mapping vertices to words and meanings. More precisely, we adopt a simple directed graph where each vertex can be optionally associated with (i) a sequence of words in the source text and (ii) a lexical meaning expressed by those words. The vertex-to-word function is determined by the type of analysis performed before text planning and can be bijective, injective and non-surjective, partial or multivalued. Thus, for instance, a planning graph instantiated from dependency trees inherits the one-to-one bijective correspondence between nodes and words. If the starting representation are deep syntactic trees, the vertex function will be injective and non-surjective, as functional words have no corresponding node in the trees.

A partial or multivalued function is the result of starting from semantic graphs with a loose anchoring on the text and which may contain multiple vertices aligned to the same words or having no anchor in the text.

While vertices, edges and the vertex-to-word function are basically derived from the analysis phase, our approach to text planning proposes means to obtain the vertex-to-meaning function, as described in Chapter 5. This meaning function is likely to be undefined for some vertices in a planning graph, making it a partial function. This is due to limited coverage of existing lexical databases: it is not realistic to expect complete coverage of all possible lexical meanings and real-world entities. In addition to this coverage issue, the function will also be undefined for vertices aligned with function words or with no alignment.

Edges in the graph obtained from the analysis phase can have labels indicating specific types of relations, e.g. syntactic dependencies. Our approach is designed to be largely independent from the nature of these relations and, by extension, from the choice of analysis tools. Additional edges are added to the graph between vertices sharing the same lexical meaning. These additional edges express semantic relatedness and also help to obtain a connected graph in those cases where the analysis produces separate sentence-level analyses.

Henceforth, we will refer to our representation as *planning graph*, and to the functions mapping vertices to sequences of words in the text and to meanings as *alignment* and *meaning* functions, respectively. The rest of this chapter is structured as follows. In Section 4.1, we give a formal description of the planning graph. In Section 4.2, we describe a mechanism used to instantiate planning graphs from UD syntactic trees, a mechanism used in our experiments. Finally, Section 4.3 briefly discusses how to instantiate planning graphs using other analysis tools.

4.1 Formal Description of the Planning Graph

Instantiating a planning graph requires a number of elements. Given:

1. A sequence of sentences $S = (S_0, \ldots, S_M)$ belonging to one or several documents, where each sentence is a sequence of words $S_i = (w_0^i, \ldots, w_N^i)$,
2. a simple directed –and possibly unconnected– graph $G = (V, E)$ resulting from the analysis of S,
3. an alignment function $a : V \to L$ that maps vertices in V to an occurrence of a word or sequence of words, i.e. a MWE, in S:
$$a(v) = (w_0^i, \ldots, w_N^i) \land S_i \in S,$$
4. a multivalued dictionary $L_M : L \to M$ where L is a set of lexical units and M is a set of lexical meanings, mapping lexical units composed of one or more words to sets of possible meanings:
$$L_M(l) = \{m_0, \ldots, m_N\} \subseteq M, \text{ and}$$
5. a partial meaning function $m : V \nrightarrow M$ that associates each vertex for which it is defined with a single, disambiguated meaning in M:
$$m(v) = \begin{cases} m \in L_M(l) & \text{if } a(v) \text{ matches a lexical unit } l \text{ in } L \\ \mathit{undefined} & \text{otherwise} \end{cases},$$

a planning graph for S is a simple directed graph $G'_S = (V', E', a, m)$ such that:

1. $V' = V$,
2. $E' = E \bigcup E_m$ where $E_m = \{(v, v') : (v, v') \notin E \land m(v) = m(v')\}$

A planning graph G'_S extends the graph G_S obtained from the analysis of the sequence of sentences S by incorporating (i) the alignment function a that maps vertices to subsequences of S, (ii) the meaning function m that maps vertices to lexical meanings in L_M, and (iii) new edges E_m labeled “same meaning” to previously unconnected pairs of vertices whenever they share the same lexical meaning.

Note that the alignment function a maps vertices to single words or MWEs, hence it maps vertices to sequences of words rather than single words. The lexical dictionary L_M maps lexical units with sets of meanings reflecting the fact that lexical databases often provide multiple meanings for ambiguous lexical units. The function of the meaning m is to disambiguate between candidate meanings returned by L_M. “same meaning” edges use m to decide if two vertices share the same meaning. If coreference relations are available following the analysis phase, these can be used to establish additional “same meaning” edges between anaphoric relations and their antecedents.

4.2 Instantiating Planning Graphs from UD Trees

In the following, we describe one way of instantiating planning graphs, which corresponds to the mechanism adopted in the experiments described in Chapter 6 and which we use to exemplify and explain the process of obtaining planning graphs. We start from the following elements:

- $G_S = (V, E)$ is an unconnected graph composed of a set of trees, one per sentence in S, where V corresponds to single tokens in the input text and E to UD-labeled dependencies between pairs of words.
- The alignment function a is a bijective function mapping each node in V to a word in S.
- L_m is the BabelNet lexical KB.
- The meaning function m mapping nodes to meanings in L_m, implemented as described in Chapter 5.

The instantiation process is very simple and comprises only two steps:

1. MWEs in every UD tree [1] are merged into a single node if L_m has

[1] MWEs are indicated using “compound”, “mwe”, “name”, “foreign” or “goeswith” dependencies.

an entry for the MWE.

2. New edges are added between pairs of vertices across the graph for which m returns the same meaning.

The graph resulting from applying these steps is a UD-based planning graph.

The choice of UD parse trees as a starting point for our experiments, already advanced in Section 2.9, is motivated by the abundance of tools, models and corpora for multiple languages. Starting from dependency trees makes the alignment between nodes in the graph and the text trivial, as every node in a dependency tree corresponds to exactly one word –and nodes in the planning graph correspond to either one word or a MWE.

As mentioned before, the meaning function m is part of our own methods for planning, described in detail in Chapter 5. It maps nodes in the planning graph to lexical meanings indexed in a lexical KB. This mapping involves matching words or groups of consecutive words in the text with lexical units in the index of the KB, and then disambiguating the meanings available for each lexical unit to the most suitable meaning. m may assign meanings to MWEs if the index contains them. MWEs in the UD trees, for which L_m is defined, are represented in the planning graph by a single vertex. This results in a simpler, sparser graph, as "same meaning" relations between parts of a MWE are avoided.

Our experimental setup uses BabelNet as a repository of lexical meanings, a choice motivated by convenience; BabelNet aggregates lexical information from multiple resources under a unified API, most notably multilingual versions of WordNet and Wikipedia. Nevertheless, other lexical resources could be used without altering the meaning function or the overall planning approach.

In order to illustrate the instantiation of a planning graph from UD trees, we go back to our running example:

> John Major met Jacques Chirac in Paris to discuss nuclear energy two months after meeting in London. This, however, was not his

first encounter with the French president in the British capital.

Figure 4.1 shows the UD trees for the two sentences, while Figure 4.2 shows a planning graph created from them, assuming that a *perfect* meaning function is available. Comparing the two representations, it can be observed that MWEs are represented as a single vertex in the planning graph and that new edges, marked in bold, have been added between pairs of vertices sharing the same meaning.

The task of detecting vertices that share the same meaning is far from trivial and encompasses various NLP tasks. Thus, anaphora resolution and detection of named entities are needed to determine that the possessive pronoun *his* corefers with *John Major*. In order to elicit the same meaning relation between *encounter* and *meaning*, it is necessary to correctly disambiguate both words and be aware that their word senses are semantically close. *British capital* can be linked with *London* if the former is recognized as a MWE and both are known to be synonymous. Similarly, associating *French president* with *Jacques Chirac* requires recognizing them as names, choosing the right meaning for each and using world knowledge to infer that Jacques Chirac was the French president at the time the text was written.

UD parsing and the meaning function of our approach cover some of the required steps for adding the "same meaning" edges shown in Figure 4.2, namely the detection of MWEs and the disambiguation of lexical meanings. Additional tools could be added to provide additional information and move us closer to a perfect meaning function, e.g., coreference solvers or word similarity calculations. In our experimental set up, however, we approximate the task by establishing links between pairs of words or MWEs assigned exactly the same BabelNet synset. In our example, this would result in the link between *British capital* and *London* being added.

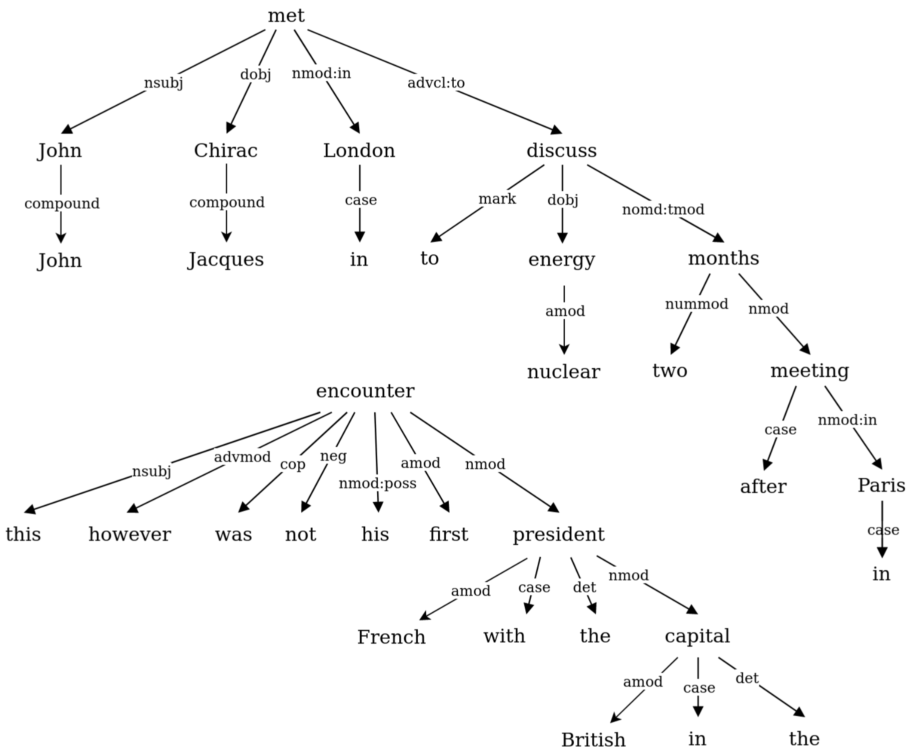


Figure 4.1: UD parse trees for the two sentences of our example.

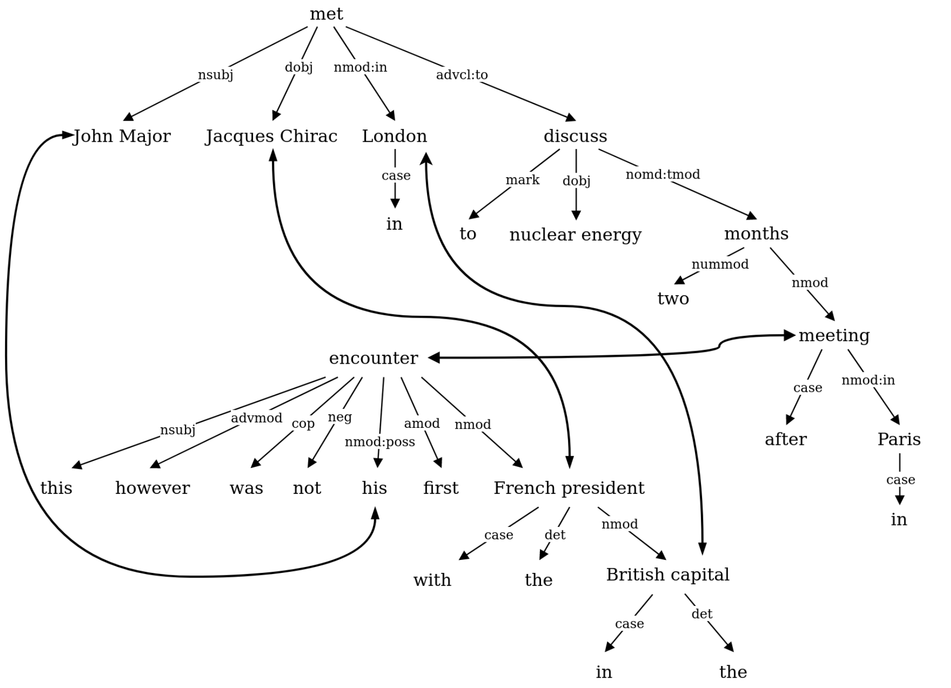


Figure 4.2: UD-based planning graph of our example. Arrows in bold correspond to “same meaning” edges.

4.3 Instantiating Planning Graphs from Other Representations

One of the stated goals behind our choice of representation is that it is sufficiently versatile to accommodate a wide range of analysis tools. The procedure described in the previous section for instantiating planning graphs from UD trees is also valid for any other dependency-based formalisms for syntactic or semantic parsing, such as Surface Syntactic Structure (SSyntS), DSyntS and EDS. In this section, we use AMR to illustrate how planning graphs can also be instantiated from representations that are not strictly based on word-to-word relations.

AMR is a formalism for describing the semantics of sentences. AMR structures are rooted, directed, acyclic graphs where leaf nodes indicate semantic types, referred in AMR literature as "concepts", and the all the other nodes correspond to "variables" instantiating these types. Two types of edges are present, those indicating that a variable instantiates a specific concept, and those that indicate roles adopted by the dependent node in relation to its governed node. AMR adopts PropBank frames as types for predicative variables and the roles filled by their arguments. All other variables instantiate ad hoc concepts labeled using words from the text.

Figure 4.3 shows the AMR structures for the two sentences of our example. The structures contain PropBank frames, i.e., 'meet-03', 'discuss-01', 'meet-03', 'contrast-01' and 'encounter-01', and generic concepts, i.e., 'energy', 'nuclear', 'after', 'year' and 'their'. Variables are linked to the concepts they instantiate through instance relations, e.g., "e1 / meet.03", marked in Figure 4.3 by dashed edges. Remaining edges correspond to either core roles associated with PropBank frames, i.e., ARG0, ARG1 or ARG2, and non-core roles taken from a closed list in the AMR specification, e.g., "location", "purpose", "mod", "time" and "quant".

AMR also dictates specific representations for certain semantic phenomena. NEs, for instance, are represented by a single variable and concept, and the variable is associated with a name using a special "name" relation. This is the case of the variables "p1" and "p2" in our example, which instantiate the concept "person" and are associated with the names *John Major* and *Jacques Chirac*. Quantities, ordinals and negation, amongst others, are also given a special treatment in AMR. In our example, the quantity *2 years* is represented using the concept 'temporal-quantity' and the relations "quant" and "unit". Similarly, the ordinal *first* in the second sentence is assigned the concept 'ordinal-entity' and represented using the relations "value" and "domain". The negation associated with the ordinal is represented using the relation "polarity".

While AMR is an unanchored semantic formalism without direct correspondence to words in the text, alignment tools have been developed by the research community that, given a sentence and its AMR graph, produce an alignment between instances in the graph and words in the text. We assume that one such aligner is used to implement the align function required to construct planning graphs.

The meaning function described in Chapter 5 can also be applied to map instances to lexical meanings on the base of their aligned words. Certain variables used to model specific semantic phenomena instantiate special AMR concepts and cannot be aligned as they cannot be related to any specific word in the text. This is the case, in our example, of the variable "t1" instantiating the concept "temporal quality".

As in the case of UD trees, turning a collection of AMR graphs into a planning graph is also really simple and requires only two steps:

1. Remove all name and concept nodes.
2. Add edges between pairs of vertices indicating instances for which m returns the same meaning.

Name and concept nodes are replaced in the planning graph by the align and meaning functions, which is the reason why they are removed in the first step. The second step is essentially the same as with UD trees, but constrained to nodes indicating variables. The planning graph for our example is shown in Figure 4.4, where "same meaning" relations are marked in bold.

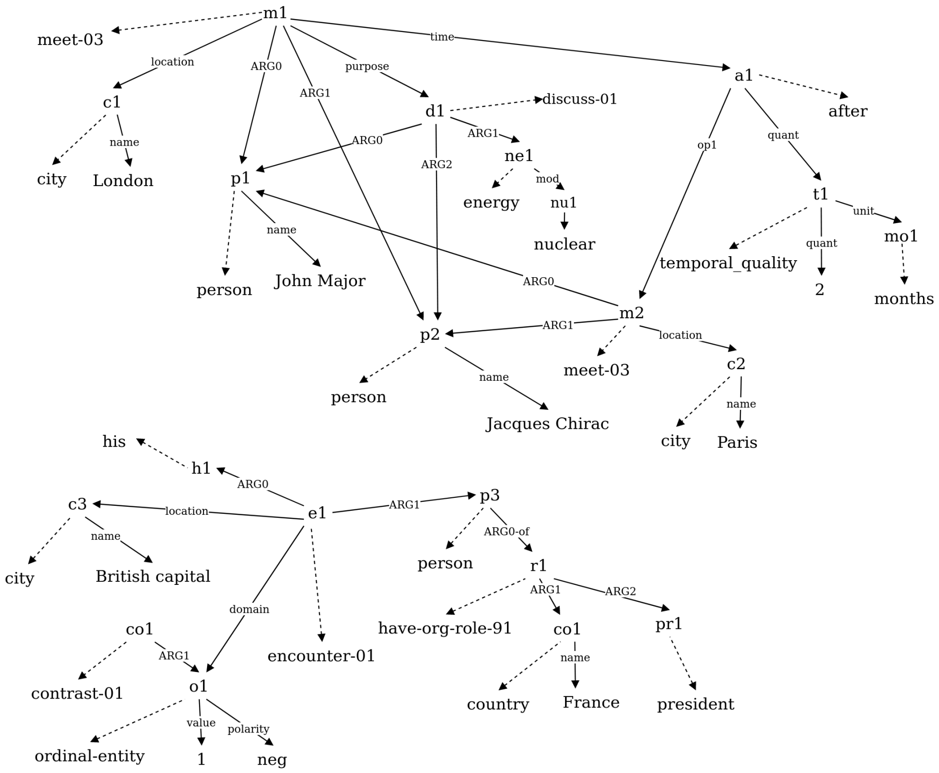


Figure 4.3: AMR graphs for the two sentences in our example.

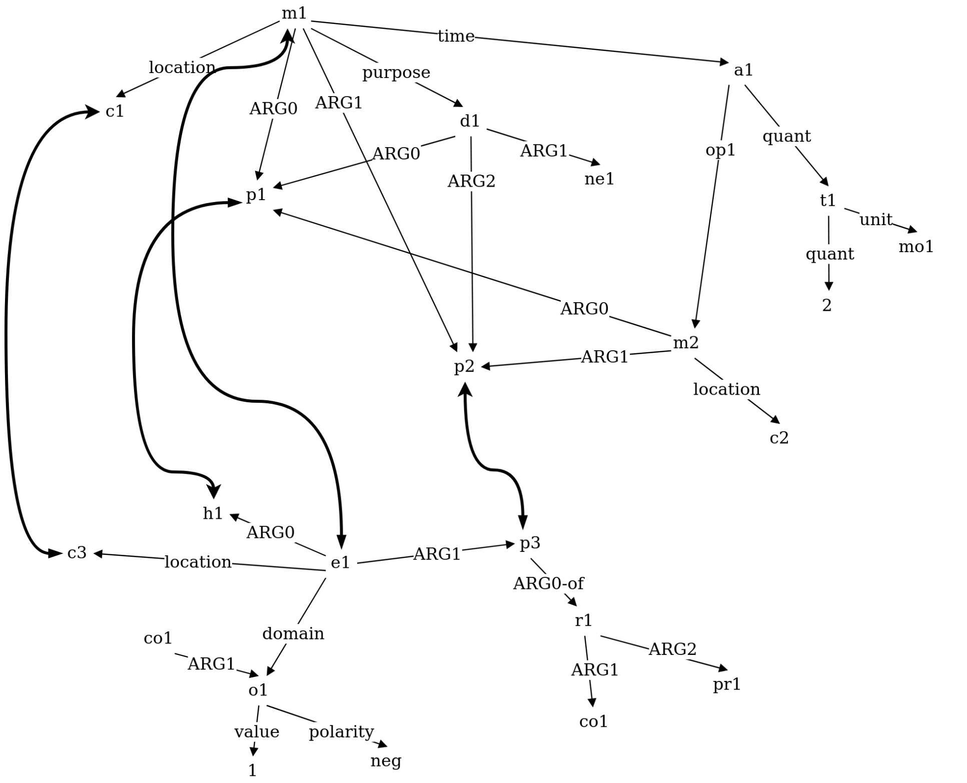


Figure 4.4: AMR-based planning graph for our example. Arrows in bold correspond to “same meaning” edges.

Chapter 5

APPROACH TO PLANNING SUMMARIES

This chapter presents a detailed description of a semantically-oriented and language-independent approach to planning summaries based on the planning graph introduced in the previous chapter. Our approach produces a text plan that consists of a complete ranking over all vertices in the graph and an ordered sequence of subgraphs extracted from it. The chapter starts with an outline of the approach in Section 5.1 followed by descriptions of each of the tasks that make up the approach, in Sections 5.2 to 5.5.

5.1 Outline of the Approach

In order to produce a text plan, we address a number of tasks sequentially, namely (i) assessment of contents, (ii) content selection, (iii) redundancy removal and (iv) content ordering. As discussed in Section 3.6, this division into tasks is common in graph-based approaches to summarization. The first three tasks correspond to the content selection part of the text planning step in traditional NLG architectures, while the ordering step can be seen as a form of discourse structuring. In the following we introduce each task.

Content assessment: A two-step ranking procedure is followed to assess contents. First, candidate meanings for words and MWEs in the input text are ranked. The resulting rank is used to create a planning graph $G = (V, E, a, m)$, as described in Chapter 4, where the meaning function m maps vertices to disambiguated meanings. Vertices in the graph are then ranked to determine the most salient contents. For the calculation of the two ranks of meanings and vertices, we use a biased graph-based centrality algorithm similar to the one applied to extractive summarization in (Otterbacher et al., 2009). This algorithm relies on a similarity function for pairs of items and a bias function that assesses items to be ranked on their own. For the ranking of candidate meanings, the similarity function compares pairs of meanings, while bias assesses candidate meanings in relation to the input text. For the ranking of vertices, similarity is a binary function indicating if two vertices are connected by an edge in G and bias is calculated with the meaning rank values produced in the first step.

Content selection: The ranks assigned to vertices in the previous task are used to produce a weighted version of the planning graph from which highly scored subgraphs are extracted. This problem is akin to the extraction of paths and subtrees from word and dependency graphs respectively, as seen in Chapter 3. Inspired by research on event detection in social networks (Rozenshtein et al., 2014; Letsios et al., 2016), we address the selection of contents from a weighted planning graph as a heavy and compact subgraph extraction problem, where *heavy* refers to nodes having a high average weight and *compact* refers to nodes in the subgraph having short distances to each other in the base graph. Both traits contribute towards producing concise and relevant summaries. Our extraction strategy is independent from language and level of text analysis, but it can be extended with mechanisms to ensure the selection of well-formed and semantically complete subgraphs.

Redundancy removal: The subgraphs extracted in the previous step may convey similar and potentially redundant information. As a mechanism for preventing redundancy, we introduce a measure of similarity between pairs of planning graphs that is based on an algorithm for calculating the edit distance between pairs of ordered trees (Zhang and Shasha, 1989). This edit distance also relies on a similarity function between pairs of meanings and serves as the basis for a simple pruning strategy to discard redundant subgraphs. The output of this redundancy removal task is a subset of the subgraphs extracted in the previous step.

Content ordering: All tasks so far address content selection from planning graphs, their output being an unordered set of subgraphs. The goal of this task is to transform this set into an ordered sequence –a text plan– where contents are organized into a coherent whole. We focus on improving local coherence and cohesion by establishing an order whereby semantically related subgraphs are positioned close to each other in the text plan. We formulate the task as a graph traversal problem where we balance the importance of each subgraph with its similarity to contents already included in the plan.

5.2 Content Assessment

The overall goal of this first task is to produce a complete assessment of the set of vertices in a planning graph in terms of their importance relative to the whole graph. We aim to conduct this assessment in a way that is independent from the language of the input text and the nature of the contents in the graph. We achieve this by ranking contents on the basis of lexical meanings taken from a large lexical knowledge base. The ranking of candidate meanings for words and MWEs in the input text is described in Section 5.2.1, while the disambiguation and the ranking of vertices of a plannign graph incorporating the disambiguated meanings are described in Section 5.2.2 and Section 5.2.3, respectively. The instantiation of the planning graph, happening after the disambiguation step and before the ranking of vertices, is already discussed in Chapter 4 and will therefore not be covered in this section.

5.2.1 Ranking Meanings

In Section 4.1, we enumerated a list of elements needed to instantiate a planning graph:

1. A sequence of sentences S belonging to one or several documents,
2. a simple directed –and possibly unconnected– analysis graph $G = (V, E)$ resulting from the analysis of text in S,
3. an alignment function a that maps vertices in V to an occurrence of a word or MWE in S,
4. a multivalued dictionary L_M that associates lexical units in L with sets of meanings in M, and
5. a partial meaning function m that associates each vertex for which it is defined with a meaning in M.

We already argued how the second, third and fourth elements could be obtained using existing tools and resources. In contrast, the meaning function was described as being part of our approach. We now proceed to describe its implementation.

Collecting candidates

Given a sequence of sentences S, their graph-based analysis G, an alignment function a and a dictionary L_M, the first step is to collect all candidate meanings in L_M for words and MWEs in S. The candidate collection procedure, shown in Algorithm 1, finds in S potential *mentions* of meanings in M. Henceforth we will use the term mention to refer to linguistic expressions consisting of a sequence of one or more consecutive words in a sentence and expression a lexical meaning found in some lexical database L_M. Our candidate collection meaning starts by looking for potential mentions of meanings in L_M with a maximum length k.

The algorithm distinguishes between single words and potential MWEs. In the first case, both word forms and lemmas of content words are added to the set of mentions. In the case of mentions formed by multiple consecutive words, they are only added to the set of mentions if the vertices that align to them in G form a connected subgraph. This requirement seeks to avoid merging unrelated nodes during the creation of the planning graph, as merging different branches in a dependency tree or semantic graph is likely to result in a planning graph where the original interpretation of the text is corrupted or lost.

Restricting the set of multi-word mentions to k consecutive word forms is useful to limit the number of lookups in L_m and avoid combinatorial explosion, but excludes any non-fixed MWEs. Given that G may contain non-fixed MWE detected during analysis, the set of mentions is extended with any additional MWEs present in G. This would include, for instance, groups of words indicated by "compound" dependencies in UD trees or by "name" and "op" edges in AMR graphs.

Each mention in the final set of mentions is then looked up in L_M and the resulting set of candidates C_S returned. Note that a candidate is a pair

(e, m) of a mention $e \in S$ and meaning $m \in M$.

Algorithm 1: Candidates collection

```
Input: Sentences $S$, analysis graph $G = (V, E)$, dictionary $L_M$
Output: Set of candidates $C_S$
begin
    $E_S \leftarrow \emptyset$                                   // mentions
    foreach $S_i \in S$ do
        $Q \leftarrow \{e : e$ is a subsequence of $S_i \ \wedge \ |e| <= k\}$
        foreach $e \in Q$ do
            if $e = \{w\}$ then                            // is a single word
                if $content(w)$ then                       // is a content word
                    $E_S \leftarrow \{e,$ lemma of $w\}$
                end
            else
                if $\exists w \in e : w$ is a noun then
                    // find induced subgraph
                    $V_e \leftarrow \{v : v \in V \ \wedge \ a(v)$ is a subsequence of $e\}$
                    if $G'[V_e]$ is connected then $E_S \leftarrow e$
                end
            end
        end
    end
    $E_S \leftarrow E_S \ \cup \ \{$MWEs in $G\}$
    $C_S \leftarrow \emptyset$                                   // candidates
    foreach $e \in E$ do
        $M \leftarrow L_M(e)$ // does mention $e$ have meanings in $L_M$?
        if $M \neq \emptyset$ then $C_S = C_S \cup (e, M)$
    end
    return $C_S$
end
```

Similarity and Context Functions

The object of our ranking task is the candidate set C_S. We base our ranking procedure on two functions, a similarity function for pairs of meanings $similarity : M \times M \rightarrow \mathbb{R}$ and a context function for candidates $context : C \rightarrow \mathbb{R}$.

$similarity$ estimates the degree of similarity between pairs of meanings and is used to assess candidates collectively, in relation to each other. The resulting scores contribute towards determining what meanings are more representative of the whole candidate set, based on the assumption that representative meanings form a subset of all candidates with high average similarity across them.

Different from $similarity$, the goal of $context$ is to assess each candidate meaning on its own. This assessment involves estimating the plausibility of a candidate meaning given the local context of its mention in the text, i.e., the words surrounding the mention. This candidate-to-context metric serves the purpose of assigning higher ranks to those candidates that bear a closer relation to the text. As we will see, it is also used to filter out meanings that are found to be unlikely candidates for their mentions, thus reducing the scale and computational cost of the ranking problem.

We integrate both functions by approaching the ranking task as a graph centrality problem. This approach involves creating a graph where vertices correspond to candidate meanings and weighted edges indicate semantic relatedness between pairs of meanings. Taking inspiration in the Biased LexRank extractive system (Otterbacher et al., 2009), weights are calculated from similarity scores produced by the $similarity$ function and biased according to the $context$ function.

In Section 6.2 we will present a number of alternative implementations for both functions that meet the requirements set in Section 1.3, i.e., that they can be applied to texts in multiple languages and cover a wide range of topics. As we will see, in most cases this is achieved by using text similarity methods to compare meaning glosses obtained from L_m to each other and to the context.

Problem Formulation

As already mentioned, we approach the ranking task as a graph centrality problem where the goal is to determine how central are its vertices with respect to the overall graph by analyzing their connections to other vertices. One popular measure of centrality is eigenvector centrality, which involves using the adjacency matrix $\boldsymbol{A}$ of a graph to derive a stochastic matrix $\boldsymbol{X}$ describing a random walk Markov chain (Markov, 1971). A random walk is a stochastic process where vertices in the graph are visited in sequential steps according to the probabilities defined in $\boldsymbol{X}$. Each position $\boldsymbol{X}_{ij}$ denotes the probability of the walk transitioning from vertex i to vertex j of the graph at the next step. Running a random walk over a large enough number of steps converges to a stationary distribution indicating the long-run probability of being in each state of the Markov chain. This distribution corresponds to the ranking of items in the graph.

Eigenvector centrality is described by the following equation:

$$\boldsymbol{R}_u = \sum_{(u,v)\in E} \boldsymbol{X}_{u,v} \cdot \boldsymbol{R}_v$$

where $\boldsymbol{R}$ is a vector expressing a distribution of probabilities over the states of the Markov Chain. This distribution is repeatedly updated using to the stochastic matrix $\boldsymbol{X}$. According to the *Perron–Frobenius Theorem*, discrete time-homogeneous Markov chains with transition matrices containing strictly positive entries are ergodic: they converge towards a unique stationary distribution regardless of the initial distribution. A common strategy to ensure that the iterative update process converges towards this unique stationary distribution is to assign a probability to jump from any given state in the Markov chain to every other state:

$$\boldsymbol{R}_u = \frac{d}{|R|} + (1-d) \cdot \sum_{(u,v)\in E} \boldsymbol{X}_{u,v} \cdot \boldsymbol{R}_v$$

The constant d in the equation above controls the added probability of the walk jumping from a state to any other state.

In order to rank meanings according to the above equation, we create an edge-weighted directed graph with candidate meanings as vertices and edges indicating pairwise similarity relations. Depending on the size of the source text and the average polysemy of its words and MWEs, the set of candidate meanings may grow very large. As we will see in Chapter 6, it is common for news articles in summarization datasets to have a number of candidate meanings in the order of thousands, which leads to ranking matrices with millions of entries. We adopt some strategies to reduce the complexity of the ranking task.

First, for polysemous mentions we only consider candidates with a context score over a threshold c_{min}, as very low similarity values indicate unlikely candidates [1]. This has a direct impact on the size of the ranking graph. Second, similarity values produced by $similarity$ are set to zero for all pairs of meanings that are candidates of exactly the same set of mentions. We assume that these candidates are mutually exclusive given that they compete as alternative interpretations of the same words, and should not be related to each other during the ranking. In addition, we also set to zero similarity values below a threshold s_{min}. Setting similarity values to zero leads to a sparser matrix and speeds up ranking calculations.

Given the set of candidates C'_S obtained by removing candidates of polysemous mentions with a low $context$ score from the original set C_S, and a modified similarity function $similarity'$ incorporating the criteria detailed above, the ranking graph $G_{meanings} = (V, E)$ has the set of meanings without repetitions in C'_S as its vertex set, and a symmetric adjacency matrix

[1] Mentions with a single candidate meaning, however, keep their single candidates regardless of the context score.

$\boldsymbol{A}$:

$$\begin{pmatrix} \mathit{similarity}'(v_0, v_0) & \mathit{similarity}'(v_0, v_1) & \cdots & \mathit{similarity}'(v_0, j) & \cdots \\ \mathit{similarity}'(v_1, v_0) & \mathit{similarity}'(v_1, v_1) & \cdots & \mathit{similarity}'(v_1, j) & \cdots \\ \vdots & \vdots & \ddots & \vdots & \ddots \\ \mathit{similarity}'(v_i, v_0) & \mathit{similarity}'(v_i, v_1) & \cdots & \mathit{similarity}'(v_i, v_j) & \cdots \\ \vdots & \vdots & \ddots & \vdots & \ddots \end{pmatrix}$$

We transform $\boldsymbol{A}$ into a row-stochastic matrix $\boldsymbol{X}$ by row-normalizing it, i.e. by replacing each non-zero value in $\boldsymbol{A}$ with a probability $p(i, j)$ obtained by dividing $\boldsymbol{A}_{i,j}$ by the corresponding row sum. We also create a row vector $\boldsymbol{L}$ from the $\mathit{context}$ scores for each meaning and dividing each score by the total sum.

Given $\boldsymbol{X}$ and $\boldsymbol{L}$, the biased eigenvector centrality score is described by the following equation:

$$\boldsymbol{MR}_u = d \cdot \boldsymbol{L}_u + (1 - d) \cdot \sum_{(u,v) \in E} \boldsymbol{X}_{u,v} \cdot \boldsymbol{MR}_v \qquad (5.1)$$

The first term of the equation depends on the context function $\mathit{context}$, while the second one depends on the modified similarity function $\mathit{similarity}'$, with the constant d balancing the contribution of each term of the equation. The stochastic matrix of a random walk based on Equation (5.1) is:

$$P_{n,n} = \begin{pmatrix} d \cdot \boldsymbol{L}_1 + (1 - d) \cdot \boldsymbol{X}_{1,1} & \cdots & d \cdot \boldsymbol{L}_1 + (1 - d) \cdot \boldsymbol{X}_{1,j} & \cdots \\ \vdots & \ddots & \vdots & \\ d \cdot \boldsymbol{L}_i + (1 - d) \cdot \boldsymbol{X}_{i,1} & \cdots & d \cdot \boldsymbol{L}_i + (1 - d) \cdot \boldsymbol{X}_{i,j} & \cdots \\ \vdots & & \vdots & \ddots \end{pmatrix}$$

The overall process followed to obtain $\boldsymbol{P}$ from a set of candidates C_S

following the steps described above is detailed in Algorithm 2.

Algorithm 2: Create meaning ranking matrix

Input: set of candidates C_S

Output: column-stochastic ranking matrix

```
begin
    $M' \leftarrow \{m : \exists(m,e) \in C_S \land context(m) \geq c_{min}\}$      // meanings
    $n \leftarrow |M'|$
    $\boldsymbol{A} \leftarrow 0_{n,n}$                                              // adjacency matrix
    for $i \leftarrow 1$ to $n$ do
        for $j \leftarrow 1$ to $n$ do
            // $m_i, m_j$ have different mentions?
            if $\{e\colon (e,m_i) \in C_S\} \neq \{e\colon (e,m_j) \in C_S\}$ then
                if $similarity(m_i, m_j) >= s_{min}$ then
                    $\boldsymbol{A}_{i,j} \leftarrow similarity(m_i, m_j)$
            end
        end
    end
    $\boldsymbol{X} \leftarrow 0_{n,n}$                                              // stochastic matrix
    for $i \leftarrow 1$ to $n$ do
        $s_i \leftarrow \sum_{j=0,n} \boldsymbol{A}_{i,j}$                            // row sum
        for $j \leftarrow 1$ to $n$ do
            $\boldsymbol{X}_{i,j} \leftarrow \frac{\boldsymbol{A}_{i,j}}{s_i}$
        end
    end
    $\boldsymbol{L} \leftarrow < context(m_0), \ldots, context(m_n) >$               // context vector
    $\boldsymbol{P} \leftarrow 0_{n,n}$                                              // biased ranking matrix
    for $i \leftarrow 1$ to $n$ do
        for $j \leftarrow 1$ to $n$ do
            $\boldsymbol{P}_{i,j} \leftarrow d \cdot \boldsymbol{L}_i + (1-d) \cdot \boldsymbol{X}_{i,j}$
        end
    end
    return $\boldsymbol{P}$
end
```

Computation of the Ranking

We address the computation of the centrality score in Equation (5.1) by applying a power iteration method. This method takes as input a row-stochastic matrix $\boldsymbol{P}$ produced by Algorithm 2 and a threshold l that acts as a stopping criterion. After initializing a row-vector $\boldsymbol{W}_0$ with an arbitrary initial distribution, the algorithm iteratively updates the distribution by right-multiplying it with $\boldsymbol{P}$. The algorithm stops when the magnitude of changes from $\boldsymbol{W}_i$ to $\boldsymbol{W}_{i+1}$ falls below l, under the assumption that the distribution has converged to the stationary distribution. This distribution of centrality scores is returned as the ranking $\boldsymbol{MR}$. Pseudo-code for the power method is shown in Algorithm 3.

Algorithm 3: Power iteration ranking

```
Input: column-stochastic $m \times m$ matrix $\boldsymbol{P}$, stopping threshold $l$
Output: stationary distribution
begin
    $\boldsymbol{W} \leftarrow 1.0/m$            // Initial guess for eigenvector
    do
        $\boldsymbol{W}' \leftarrow \boldsymbol{P} \cdot \boldsymbol{W}$                  // Right-multiply P with W
        $\boldsymbol{W}' \leftarrow \boldsymbol{W}' \cdot \sum_{i=0}^{n} |r_i|$        // Normalize W with $L_1$ norm
        $\delta \leftarrow \max(|\boldsymbol{W}' - \boldsymbol{W}|)$             // Magnitude of update
        $\boldsymbol{W} \leftarrow \boldsymbol{W}'$
    while $\delta > l$
    return $\boldsymbol{W}$
end
```

5.2.2 Disambiguation

Recall that the goal of our ranking of meanings was to come up with a function mapping text to disambiguated meanings, as this function is required to instantiate planning graphs. Given a set of candidate meanings for a word or MWE, the best candidate is likely to maximize similarity with its context and with other meanings across the text. As pointed out in

Section 2.2, these intuitions –similarity to context and to other meanings– are the driving force behind most approaches to WSD and EL. Taking inspiration from works that search for maximally coherent global assignments of senses, we use $\boldsymbol{MR}$ to implement the meaning function m.

Given a set of candidates C_S, a mention e and its set of candidate meanings M_e s.t. $\forall m \in M_e \Rightarrow (m, e) \in C_V$, we assign a score to each meaning based on $\boldsymbol{MR}$:

$$score(m) = \begin{cases} \boldsymbol{MR}(m) & \text{if } \textbf{MR} \text{ is defined for m} \\ \mathit{0} & \text{otherwise} \end{cases}$$

For single words not part of any MWE, the $score$ function can be used to select the highest ranked candidate meaning. In the presence of MWEs, however, disambiguation involves deciding not only between candidate meanings but also between overlapping mentions [2]. We adopt a simple strategy and choose a mention over other overlapping mentions whenever its best ranked meaning has a greater rank value than the best ranked meanings of overlapping mentions. Algorithm 4 shows how, starting from the set of candidates C_S and the $score$ function defined above, our strategy is applied to obtain a dictionary mapping mentions in S to meanings in M. The returned dictionary is undefined for mentions overlapping other mentions that are judged to provide a better interpretation of the overall text.

The meaning function $m : V \nrightarrow M$ required to instantiate a planning graph can be easily obtained from the dictionary produced in Algorithm 4 by passing the result of applying the alignment function on a vertex of the planning graph to the dictionary. m will be defined for a vertex v if the

[2] Consider, for instance, the MWE *renewable energy sources*. A dictionary may contain meanings for overlapping mentions *renewable energy* and *energy source*, and also for each individual word. A strategy is needed to decide what combination of meaning and mention offers the best interpretation.

dictionary is defined for its aligned words $a(v)$.

Algorithm 4: Disambiguation

```
Input: set of candidates $C_S$, function $score$
Output: partial function from mentions in $S$ to meanings
begin
    $M_S \leftarrow$ empty dictionary                    // disambiguated meanings
    $E \leftarrow \{e : \exists(m, e) \in C_S\}$                              // mentions
    foreach $e \in E$ do
        $M^e \leftarrow \{m : \exists(m, e) \in C_S\}$               // candidates for $e$
        if $|M^e| = 1$ then                              // only one meaning
            $M_S[e] \leftarrow m_0 \in M^e$
        else
            $s^e_{max} \leftarrow \max_{m \in M^e} score(m)$           // max candidate score
            $E' \leftarrow \{e' : e, e'$ overlap in $S\}$  // overlapping mentions
            $M' \leftarrow \bigcup_{e' \in E'} \{m : \exists(m, e') \in C_S\}$        // their meanings
            $s'_{max} \leftarrow \max_{m \in M'} score(m)$        // max overlapping score
            if $s^e_{max} \geq s'_{max}$ then
                $m^e \leftarrow \arg\max_{m \in M^e} score(m)$
                $M_S[e] \leftarrow m^e$
            end
        end
    end
    return $M_S$
end
```

5.2.3 Ranking Vertices

The meaning function m of a planning graph is partially defined for the set of vertices in a graph. This may be due to vertices that do not align with words in the text or because these words are not indexed in the dictionary L_M. Furthermore, ranking values produced by $\boldsymbol{MR}$ will be the same for all mentions of a meaning in the text, but the actual importance of multiple mentions of the same meaning depends on the textual context in which they occur. These issues are problematic for the selection of contents, as they hamper our ability to assess the relevance of specific parts of a planning graph and, by extension, of the input text.

Consider, for instance, the verb *meet* occurring in the first sentence of our running example. It can be argued that its importance is a function of the participants in the situation it denotes: the politicians *John Major* and *Charles Chirac*, the city of Paris, the event denoted by *discuss* and the temporal expression introduced by *after*. $\boldsymbol{MR}$ does not account for their contributions to the importance of *meet*. Even if all these expressions are disambiguated to a suitable meaning in L_M, the meaning-based $similarity$ function is not likely to assign high similarity scores between *meet* and the named entities *John Major* and *Charles Chirac*.

We wish to obtain an individual assessment of all vertices in a planning graph and aligned words in the text that takes into account not only the meanings associated with vertices but also the context of each vertex, i.e., how is it related with other words in the text. Recalling from Chapter 4 that edges in a planning graph indicate relations between words obtained from text analysis, these edges can be used to transfer relevance across vertices.

Taking all the above into consideration, we adopt the same ranking formulation used for the ranking of meanings. This time the a priori relevance of items used to bias the ranking is estimated from the ranking scores $\boldsymbol{MR}$ assigned to vertex meanings, while the similarity function is based on the set of edges in the graph. More precisely, we use an indicator function $adj_G : V \times V \to \{0, 1\}$ for pairs of vertices in a planning graph G that returns '1' if two vertices are adjacent in the graph and '0' otherwise:

$$adj_G(u,v) = \begin{cases} 1 & \text{if } (u,v) \in G \\ 0 & \text{otherwise} \end{cases}$$

We transform the adjacency matrix of G, based on adj and containing only zeros and ones, into a row-stochastic matrix $\boldsymbol{Y}$ by row-normalizing it and encode bias by creating a row vector $\boldsymbol{M}$ from the ranking scores of the previous step. More precisely, the value $\boldsymbol{M}_v$ for a node v corresponds to the $\boldsymbol{MR}$ value for the meaning of v. If either v has no meaning or its meaning has no rank value, then $\boldsymbol{M}_v$ is set to the arithmetic mean of all ranking values in $\boldsymbol{MR}$:

$$\boldsymbol{M}_v = \begin{cases} \boldsymbol{MR}_{m(v)} & \text{if } m(v) \text{ and } \boldsymbol{MR}_{m(v)} \text{ are defined} \\ \overline{\boldsymbol{MR}} & \text{otherwise} \end{cases}$$

$\boldsymbol{M}$ is normalized by dividing each value by the sum of all of its values. Given $\boldsymbol{Y}$ and $\boldsymbol{M}$, the biased eigenvector centrality score for the set of vertices in a planning graph G is described by the following equation:

$$\boldsymbol{VR}_u = d \cdot \boldsymbol{M}_u + (1-d) \cdot \sum_{(u,v)\in E} \boldsymbol{Y}_{u,v} \cdot \boldsymbol{VR}_v \tag{5.2}$$

As in Equation (5.1), the first term of the equation expresses bias, towards meaning ranks, while the second relates items to each other, via edges in the planning graph. Algorithm 5 describes the creation of a vertex ranking matrix $\boldsymbol{Q}$ based on Equation (5.2) and starting from a planning graph and a ranking of meanings. $\boldsymbol{Q}$ is applied to the computation of the ranking over vertices using the same power iteration method described in

Section 5.2.1.

Algorithm 5: Create a vertex ranking matrix

```
Input: planning graph $G_S = (V, E, a, m)$, ranking $\boldsymbol{MR}$
Output: column-stochastic ranking matrix for $V$
begin
    $n \leftarrow |V|$
    $\boldsymbol{A} \leftarrow 0_{n,n}$                                  // adjacency matrix
    for $i, j \leftarrow 1$ to $n$ do
        $\boldsymbol{A}_{i,j} \leftarrow adj_G(v_i, v_j)$              // are $v_i, v_j$ adjacent?
    end
    $\boldsymbol{Y} \leftarrow 0_{n,n}$                                  // stochastic matrix
    for $i, j \leftarrow 1$ to $n$ do
        $\boldsymbol{Y}_{i,j} \leftarrow \frac{\boldsymbol{A}_{i,j}}{\sum_{j=0,n} \boldsymbol{A}_{i,j}}$      // divide by row sum
    end
    $\boldsymbol{M} \leftarrow \emptyset$                                    // meanings vector
    foreach $v_i \in V$ do
        if *if* $m(v_i)$ *and* $\boldsymbol{MR}_{m(v_i)}$ *are defined* then
            $\boldsymbol{M}_i \leftarrow \boldsymbol{MR}_{m(v_i)}$
        else
            $\boldsymbol{M}_i \leftarrow \overline{\boldsymbol{MR}}$
        end
    end
    $\boldsymbol{Q} \leftarrow 0_{n,n}$                              // biased ranking matrix
    for $i, j \leftarrow 1$ to $n$ do
        $\boldsymbol{Q}_{i,j} \leftarrow d \cdot \boldsymbol{M}_i + (1 - d) \cdot \boldsymbol{Y}_{i,j}$
    end
    return $\boldsymbol{Q}$
end
```

5.3 Content Selection

Following the assessment of contents in the graph, we now turn our attention to identifying and selecting clusters of highly relevant contents from which a summary of the input texts can be produced. More precisely, we seek to extract subgraphs of the main planning graph that concentrate highly ranked vertices and extract them as basic units for the composition of a summary. In Section 5.3.1, we describe a simple greedy algorithm for the extraction of a heavy and compact subgraph. This greedy algorithm serves as the foundation for a sampling strategy for the extraction of multiple subgraphs, as described in Section 5.3.2. In Section 5.3.3, we briefly discuss mechanisms to enforce the extraction of well-formed subgraphs, an important concern when targeting abstractive summaries that require NLG.

5.3.1 Greedy Extraction of a Subgraph

The subgraphs targeted by our selection strategy must be reasonably small and closely connected to ensure that they include related contents that facilitate the extraction of short passages or the generation of semantically sound sentences. We formulate the selection of contents from a planning graph as a subgraph extraction problem where relevance and conciseness must be balanced to produce graphs that are both heavy, i.e., vertices accumulate high weights, and compact, i.e., subgraphs have few vertices with short distances between them. This formulation is expressed via a weighting function that assigns vertices with normalized ranking scores obtained from the assessment of contents and an edge weighting function that assigns a uniform cost to all edges.

The extraction of compact subgraphs with short distances between nodes and a large sum of weights has been addressed in the context of network analysis. Rozenshtein et al. (2014) experimentally showed that, while the problem is NP-hard for general graphs, standard greedy algorithms performed as well as more sophisticated alternatives at finding approximated solutions to the problem with multiplicative performance guarantees. We follow their insights and adopt a greedy algorithm for the task of extracting a single subgraph.

Given a planning graph $G = (V, E, a, m)$, a vertex weight function $w : V \rightarrow \mathcal{R}$ s.t. $\forall v \in V : w(v) = \boldsymbol{VR}_v$, a cost function $c : E \rightarrow \mathcal{R}$, and a normalization coefficient λ, the extraction of heavy and compact subgraphs consists in finding a subset of vertices $S \subseteq V$ such that they form an induced subgraph $G'[S]$ that maximizes the following objective function:

$$Q(S) = \lambda W(S) - C(S) + C(V).$$

where:

$$W(S) = \sum_{v \in S} w(v)$$

$$C(S) = \sum_{(u,v) \in G[S]} c((u, v))$$

The two terms in the equation above, $W(S)$ and $C(S)$, reflect the tradeoff between increasing the weight of the subgraph and keeping it compact. The term $C(S)$ corresponds to the sum of costs of all edges in the induced subgraph G[S], while $C(V)$ is used to ensure that the function is non-negative.

Algorithm 6 shows the pseudo-code for extracting a single subgraph from a planning graph G by greedily optimizing the objective function Q. It starts from an initial solution or state S containing the highest ranked vertex in G. In each iteration of the algorithm, a candidate set C is generated by considering new vertices in the neighborhood of S. The algorithm chooses a candidate according to the Q-value of each candidate, and checks whether the new solution resulting from adding the chosen candidate to S has improved. When the Q-value of S stops improving, the algorithm stops and returns the subgraph of G induced by S.

Algorithm 6: Greedy extraction

```
Input: Planning graph $G = (V, E, a, m)$, weight and cost functions $w$ and $c$
Output: a subgraph of $G$
begin
    $S \leftarrow \emptyset, S' \leftarrow \{\arg\max_{v \in V}(w(v))\}$        // top ranked vertex
    $q \leftarrow 0\,, q' \leftarrow Q(S')$        // Use Q as objective function
    while $q' > q$ do
        $S \leftarrow S'$
        $C \leftarrow \emptyset$        // candidate set
        foreach $v \in S$ do
            foreach $v' \in N_G(v)\ \wedge\ v' \notin S$ do
                add $\{S\ \cup\ v'\}$ to $C$
            end
        end
        $S' \leftarrow \arg\max_{C_i \in C}(Q(C_i))$        // greedy selection
        $q \leftarrow q'\,, q' \leftarrow Q(S')$
    end
    return $G[S]$        // return induced subgraph
end
```

As we will discuss in Section 5.3.3, we constrain our greedy algorithm to extract subgraphs from a single sentence each by assigning a large fixed cost $C(V)$ to edges of type “same meaning”, effectively excluding them when assessing candidates with Q.

5.3.2 Sampling Multiple Subgraphs

The overall goal of the selection step is to obtain multiple subgraphs from which to extract text fragments or generate sentences of a summary. We extend the greedy algorithm presented in the previous section to sample multiple graphs stochastically based on the distribution of weights in the graph.

First, we change the deterministic policy for candidate selection to a probabilistic policy by replacing $argmax$ with a selection policy based on a softmax distribution over vertex weights. Given a weighted graph $G = \{V, E, w\}$, a softmax distribution $\sigma(V)$ is calculated as follows:

$$\sigma(V)_i = \frac{e^{w(v_i)}}{\displaystyle\sum_{v' \in S} e^{w(v')}} \text{ for } i = 1 \ldots |V|$$

A softmax distribution can also be calculated for sets of vertices:

$$\begin{aligned} \sigma(C)_i &= \frac{e^{W(C_i)}}{\displaystyle\sum_{C_j \in C} e^{W(C_j)}} \text{ for } i = 1 \ldots |C| \\ W(C) &= \sum_{v \in C} w(v) \end{aligned}$$

We extend the algorithm to perform the extraction procedure multiple times and, at every step, check if the extracted graph is a subgraph of any of the previously extracted graphs. This check is greatly simplified by the fact that all subgraphs are induced by a subset of the vertex set of the host graph. Consequently, given two subgraphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, G_1 is subgraph of G_2 iff $V_1 \subseteq V_2$, which can be implemented by comparing sets of vertices instead of addressing it as an NP-complete subgraph isomorphism problem. The new algorithm is shown in Algorithm 7.

Algorithm 7: Sampling subgraphs

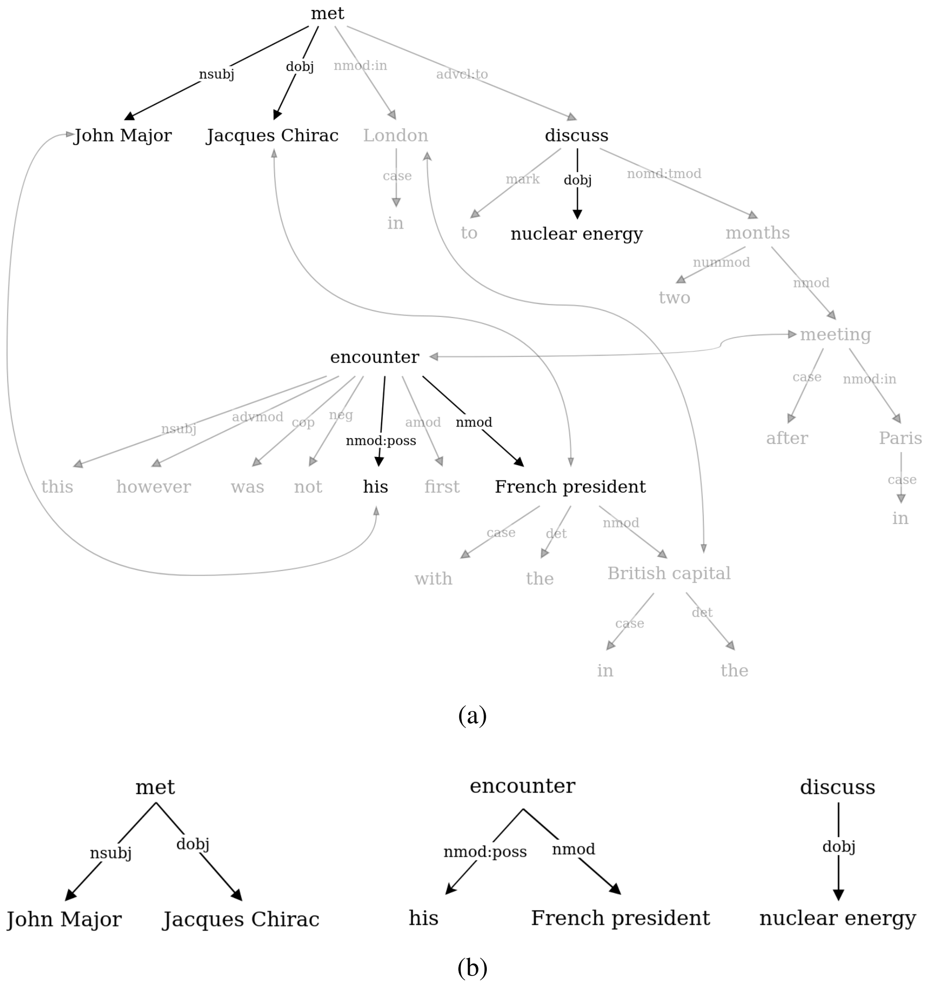
```
Input: Planning graph $G = (V, E, a, m)$, weight and cost functions $w$ and
       $c$, number of subgraphs to extract $m$
Output: Set of subgraphs of $G$
begin
    $I \leftarrow \emptyset$                                  // induced subgraphs
    for $i \leftarrow 0$ to $m$ do
        $S \leftarrow \emptyset$
        $S' \leftarrow$ select from $C$ according to $\sigma(V)$
        $q \leftarrow 0$ , $q' \leftarrow Q(S')$
        while $q' > q$ do
            $S \leftarrow S'$
            $C \leftarrow \emptyset$                          // determine candidate set
            foreach $v \in S$ do
                foreach $v' \in N_G(v) \ \wedge \ v' \notin S$ do
                    add $\{S \ \cup \ v'\}$ to $C$
                end
            end
            $S' \leftarrow$ select from $C$ according to $\sigma(C)$
            $q \leftarrow q'$ , $q' \leftarrow Q(S')$
        end
        // Is $S$ already in $I$?
        if $\neg\exists\, G_i = (V_i, E_i) : \ G_i \in I \ \wedge \ V_i' \supseteq S$ then
            $I \leftarrow I \cup G'[S]$                        // add induced subgraph
        end
    end
    return $I$
end
```

Extracting subgraphs from our running example

Figure 5.1 shows some hypothetical subgraphs extracted from the UD-based plan of our running example. Assuming that the vertex aligned with *John Major* is a highly ranked vertex according to w, it will be chosen with high probability by the softmax policy. If that is the case, then the initial state S becomes a set with *John Major* as its only element. The algorithm then creates a set of candidate states C to succeed S from the neighbours of *John Major* in the planning graph. The only neighbour is the vertex aligned with *met* ("same meaning" relation connecting to *his* is ignored) and therefore C contains a single candidate state with the two vertices $\{$*John Major*, *met*$\}$. At this point, the algorithm stops expanding S if the Q-value q' of the candidate is less or equal than the old value q, i.e. $Q(\{$*John Major*, *met*$\} <= Q(\{$*John Major*$\})$.

Assuming the opposite to be true, the candidate becomes the new state S. *met* has multiple neighbours that are not part of S, resulting in a new set of candidates $C = \{$ *John Major*, *met*, *Jacques Chirac*$\}$, $\{$*John Major*, *met*, *Paris*$\}$ and $\{$*John Major*, *met*, *discuss*$\}$. We assume that the softmax policy picks $\{$*John Major*, *met*, *Jacques Chirac*$\}$ as the successor state S. If in a subsequent iteration no candidate expansion of the subgraph has a greater Q-value then the subgraph of G induced by the selected vertices is added to the set I. This induced subgraph is shown at the bottom-left corner of Figure 5.1.

After completing a first subgraph, the sampling algorithm will pick a new initial vertex for another subgraph using the softmax policy. If the softmax policy chooses, for instance, *met* or *Jacques Chirac*, then the algorithm may end up extracting the same subgraph again, or part of it. The last check in Algorithm 7 ensures that subsets of $\{$*John Major*, *met*, *Jacques Chirac*$\}$ are never used to add a subgraph to I. Figure 5.1 also shows two more subgraphs that could be potentially extracted by the sampling algorithm. Note that the Q function is designed to promote small, compact subgraphs, but it does not prevent semantically equivalent and potentially redundant subgraphs from being extracted, as is the case of the two first subgraphs in the figure. Redundancy prevention is discussed in Section 5.4. In the following subsection we will discuss the extraction of well-formed subgraphs.



(a)

(b)

Figure 5.1: Subgraph extraction from the UD planning graph of our example

5.3.3 Well-formed Subgraphs

The selection methods presented so far are independent from the specific lexical meanings in the graph and from the nature of the underlying relations obtained from text analysis and used to establish edges in the planning graph. If the subgraphs are applied to extract fragments smaller than sentences or used as input to an NLG component, however, each subgraph should be well-formed and semantically complete – well-formed with respect to the representation from which the planning graph is instantiated and complete with respect to the set of lexical meanings conveyed by the subgraph.

In its current state, the algorithm shown in Algorithm 7 has no mechanism to enforce the extraction of subgraphs that express complete meanings. Looking at our example, our algorithm would extract the subgraph induced by the vertices $\{$*John Major*, *met*$\}$ if this subgraph had a greater Q-value than the first subgraph in Figure 5.1. This is despite the fact that the extracted subgraph is semantically incomplete and that a UD-to-text generation component would not be able to produce a grammatically correct text containing the transitive verb *meet*, as there are no vertices to realize both its mandatory arguments.

A subgraph that appears to be complete may nevertheless possess structural problems. Consider, for instance, adding the vertices *discuss* and *nuclear energy* to the first subgraph in Figure 5.1. The result may be semantically sound from the point of view of the lexical meanings involved, but the fact that it does not include the dependency “mark“ to the function word *to* means that it is not a well-formed tree according to the UD specification. A similar situation arises when considering formalisms other than UD. The AMR-based graph for our example shown in Figure 4.4 contains special AMR constructions used to represent quantities, ordinals or temporal expressions that involve multiple vertices and edges. These complex constructions should always be selected as a whole or the resulting subgraph will not constitute a valid AMR graph.

Even a well-formed subgraph containing a seemingly complete set of lexical meanings may convey a different meaning from that of the input text if certain words are missing. This is the case, for instance, of the negation in the second sentence of our example (*...this, however, was not his first encounter...*), which should always be included or else the original meaning is dramatically altered.

In order to avoid the problems described above, our subgraph extraction strategy should be extended with mechanisms to enforce structural well-formedness and promote semantic correctness and completeness. These mechanisms are likely to be specific to the formalism used to instantiate planning graphs. Statistical models such as dependency language models (Shen et al., 2008) may be a good choice for syntactic formalisms, as they can be used to evaluate dependency-based subgraphs and predict what additional dependencies are required to complete a subgraph during its extraction. They can be incorporated as part of the cost function c so that, when evaluating candidate expansions to a subgraph with Q, the best candidates are those that maximize the likelihood of the resulting subgraph being a correct dependency tree.

Rules and constraints filtering out incorrect subgraphs from the set of valid solutions can be particularly effective for planning graphs based on semantic representations. Constraints can be used to enforce that predicates have all their required arguments selected along them, that negations and modals are always selected, or to treat complex numerical or temporal expressions as atomic items during content selection. In addition to these mechanisms, simplification or language models can be applied to rank sentences in an overgenerate-and-rank approach (Langkilde and Knight, 1998; Walker et al., 2001) to text generation.

These mechanisms for enforcing correct subgraphs are dependent on the type of analysis performed prior to planning or the language of the summary. For this reason, and also because our summarization experiments are conducted for the production of extractive summaries where well-formedness issues are less important, we leave these considerations for future work.

Another important issue regarding the selection of well-formed subgraphs is the presence of “same meaning” relations. Due to their nature, these relations are likely to connect edges aligned with words in different sentences of the input text. Considering them during extraction may lead to subgraphs containing information from multiple sentences, which requires additional effort to ensure that the result is a well-formed and semantically sound subgraph. As already mentioned, we constrain content selection to produce subgraphs extracted from a single sentence each by assigning a large fixed cost $C(V)$ to edges of type “same meaning’.

5.4 Redundancy Removal

Identifying redundancy across contents selected for inclusion in a summary involves determining when separate pieces of contents are equivalent in terms of the meaning they communicate. In our setting, this involves comparing subgraphs extracted in the previous step to identify those that are equivalent and therefore most likely to be redundant.

Planning graphs are instantiated from word-to-word relations – possibly linguistic in nature – and include lexical meanings. This implies that multiple graph configurations can represent equivalent or nearly-equivalent meanings, as a planning graph is not likely to abstract away functional and lexical choices in the input text. Consequently, redundancy detection cannot be reduced to an isomorphism problem between subgraphs but requires a more flexible approach.

Generally speaking, to compare graphs involves finding a function $s : \mathcal{G} \times \mathcal{G} \rightarrow \mathbb{R}$ that quantifies the similarity or dissimilarity between pairs of graphs. The problem of graph similarity and dissimilarity is related to that of graph and subgraph isomorphism. However, rather than requiring an exact match between graphs, as is the case in isomorphism problems, algorithms for graph similarity are able to quantify differences between pairs of graphs, not only in terms of their structures but also by comparing their labels.

Edit distance algorithms are widely used methods for flexible comparison of graphs where the costs associated to edit operations can be used to tailor them to specific applications (Neuhaus and Bunke, 2007). While the time and space complexity of edit distance algorithms grows exponentially with the size of the graphs to be compared, there are well known algorithms that compute tree edit distances in polynomial time. In the following we describe how planning (sub)graphs can be converted into ordered trees and compared using the algorithm by Zhang and Shasha (1989) for calculating edit distances between pairs of vertex-labeled ordered trees.

5.4.1 Converting Planning Graphs into Ordered Trees

We cannot assume that the directed subgraphs extracted in the previous step are trees, as graph-based representations can be used to instantiate planning graphs. Unlike graphs, trees can have only one root and one ancestor per node. In addition, the trees compared by Zhang and Shasha (1989) have an order between siblings and have labels assigned only to their vertices; edges are unlabeled. Our conversion procedure involves replicating vertices in the graph with multiple parents and relabeling all vertices with their lexical meanings. While the former eliminates reentrancy from the graph, the latter allows us to establish a lexicographical order between siblings.

Before describing the procedure in detail, let us define the following graph operation:

Definition 5.1 (Vertex replication)**.** Let $G = (V, E, a, m)$ be a planning graph, w the vertex weight function based on the ranking of vertices $\boldsymbol{VR}$ and c the cost function described in Section 5.3.1. Given a vertex $v \in V$ and its set of direct ancestors of $A_v = \{a : a \in V \wedge (a, v) \in E\}$, a vertex replication consists in the following steps:

(1) for each ancestor $a \in A_v$, create a new vertex v_a s.t. $a(v_a) = a(v)$, $m(v_a) = m(v)$, $w(v_a) = w(v)$ and v_a has the same label as v,

(2) for each ancestor $a \in A_v$, create a new edge (a, v_a) s.t. $c((a, v'_a)) = c((a, v))$ and the new edge has the same label as (a, v),

(3) for each ancestor $a \in A_v$ and each outgoing edge $(v, d) \in E$ create a new edge (v_a, d) s.t. $c((v_a, d)) = c((v, d))$ and the new edge has the same label as (v, d), and

(4) remove v from G.

■

Definition 5.2 (Planning graph to tree conversion)**.** Let $G = (V, E, a, m)$

be a planning graph where:

- μ and ν are the vertex and edge labeling functions respectively and
- $\mathcal{R} = \{v : v \in V \wedge deg^{-}(v) = 0\}$ is the set of roots.

A conversion of G into a vertex-labeled ordered tree involves the following steps:

(1) iteratively perform a vertex replication of every vertex v s.t. $\{v : v \in V \wedge deg^{-}(v) > 1\}$, until all nodes have a single parent: $\forall v : v \in V \Rightarrow deg^{-}(v) \leqslant 1$.

(2) If the resulting graph is a forest, add a new root node r s.t. $\mu(r) = Root$, and connect it to all roots $r' \in \mathcal{R}$ via edges s.t. $\nu(r, r') = root$.

(3) For all $v \in V$ and their single incoming edge e_v, apply a new vertex labeling function μ':

$$\mu'(v) = \begin{cases} \nu(e_v) + m(v) & \text{if } m(v) \neq \emptyset \\ \nu(e_v) + \mu(v) & \text{if } m(v) = \emptyset \end{cases}$$

.

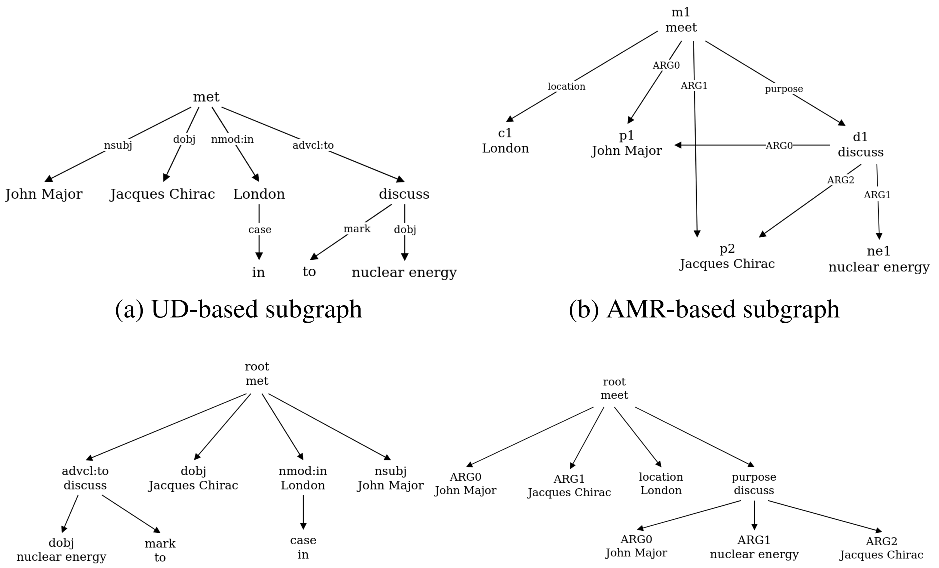
(4) Sort all sets of siblings lexicographically based on μ'.

■

Figure 5.2 illustrates this transformation by showing, in its top half, two equivalent subgraphs extracted from the UD-based and AMR-based planning graphs in Figure 4.2 and Figure 4.4 respectively, while the bottom half shows the trees resulting from applying the conversion described above. Compared to the UD-based tree in Figure 5.2a, vertices in Figure 5.2c have been renamed to include edge labels indicating UD dependencies, and siblings have been sorted in lexicographic order[3]. The conversion is more complex if the starting subgraph is not a tree, as is the case of the AMR-based subgraph in Figure 5.2b. In the corresponding

[3] Lexical meanings in vertex labels are indicated by words in the text.

tree in Figure 5.2d, vertices with multiple ancestors "p1" and "p2" have been replicated to avoid reentrancy.



(a) UD-based subgraph

(b) AMR-based subgraph

(c) UD-based tree

(d) AMR-based tree

Figure 5.2: Planning subgraphs and corresponding vertex-labeled ordered trees

Calculating Edit Distances

The distance returned by Zhang and Shasha (1989) algorithm for a pair of trees corresponds to the sum of costs associated with each of the operations required to transform one tree into the other. Three operation types are supported by the algorithm: to *rename* one vertex label to another label, to *delete* a vertex and to *insert* a vertex. Recalling that the vertices in our trees are labeled with a role r followed by a meaning m, we parameterize the algorithm with the following cost assignment:

$$C_{insert} = 1.0$$
$$C_{delete} = 1.0$$
$$C_{rename} = 1.0 - d_e \cdot \delta(r_1, r_2) - (1.0 - d_e) \cdot \mathit{similarity}_e(m_1, m_2)$$

A maximum cost 1.0 is assigned to both inserts and deletes. Renames are assigned a variable cost which is a function of the similarity between the two labels involved in the operation. More precisely, the cost depends on the roles r_1 and r_2 being equal, as indicated by the Kronecker delta function δ, and on the similarity value between the meanings m_1 and m_2, estimated by the similarity function $\mathit{similarity}_e$. The similarity function $\mathit{similarity}_e$ returns the value of the $\mathit{similarity}$ function described in Section 5.2.1 except in two cases where it returns zero: (i) if the similarity value is below a fixed threshold s_{min}[4] and (ii) if either m_1 or m_2 do not indicate actual meanings, i.e. vertices for which the meaning function m associated with the planning graph is undefined. The contribution of both functions to the cost of the operations is balanced by a constant d_e, the value of which is set experimentally as described in Chapter 6. This cost function is designed so that renaming one label to another with the same role and a similar meaning constitutes a cheap operation, with cost raising if roles differ or meanings are less similar.

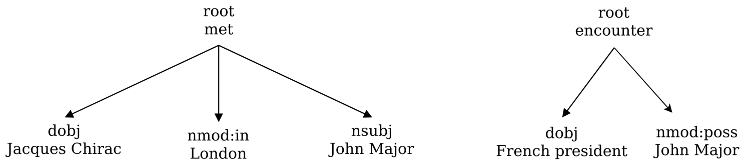


Figure 5.3: Two potentially redundant trees.

When applied to the two trees shown in Figure 5.3, one possible return

[4] This minimum similarity value is also the same used for the ranking of meanings in Section 5.2.1.

from Zhang and Shasha (1989) algorithm is the following sequence of operations to the left-hand tree:

1. Delete node labeled "nmod:in London".
2. Rename node labeled "root met" to "root encounter".
3. Rename node labeled "dobj Jacques Chirac" to "dobj French president".
4. Rename node labeled "dobj John Major" to "nmod::poss John Major".

The delete operation has a fixed cost of 1.0, while the rename operations have lower –and variable– costs. Renaming "root met" to "root encounter" is likely to have a small cost given the coincidence of roles and the high similarity between meanings. A similar situation can be expected in the second rename, while the third one requires changing the role but not the meaning. Note that rename operations can have the same cost as an insert or delete if the roles do not match and the meanings are sufficiently dissimilar according to the fixed similarity threshold k. Also note that, since the ordering of siblings has edge labels assigned to vertices as its main criteria, having the same meaning across two graphs but in different roles will result in higher cost operations and, consequently, a lower similarity value. To see why, consider what would happen if the right-hand tree in Figure 5.3 had the roles inverted for *John Major* and *Jacques Chirac*. This would invert their order in the tree, resulting in a more costly sequence of operations required to transform one tree into another.

5.4.2 Removing Redundant Subgraphs

The transformation of planning graphs to ordered trees, to which we will refer as $\mathcal{T}$, and the distance calculated by Zhang and Shasha (1989) algorithm parameterized with our cost assignment, henceforth referred to as δ, provide the basis for estimating the overall similarity between arbitrary pairs of planning graphs.

For the task of detecting pairs of potentially redundant subgraphs extracted from the same planning graph, we implement a simple pruning strategy where, given a set of subgraphs, we iteratively identify the pair with the highest similarity according to δ and discard the one with lowest average weight. The process stops once the desired number of subgraphs has been reached. The pseudo-code for the selection of subgraphs is shown in Algorithm 8.

Algorithm 8: Remove redundant subgraphs

```
Input: Set of subgraphs $I$ taken from a planning graph $G = (V, E, a, m)$,
       number of subgraphs to select $n$
Output: Subset $I' \subseteq I$
begin
    $I' \leftarrow I$
    while $|I'| \geq n$ do
        $A, B \leftarrow \arg\max_{A,B \in S'}(\delta(\mathcal{T}(A), \mathcal{T}(B)))$    // Most similar pair
        $w_A \leftarrow \sum_{v \in V_A} (w(v)) \;/\; |V_A|$    // Average weight
        $w_B \leftarrow \sum_{v \in V_B} (w(v)) \;/\; |V_B|$    // Average weight
        // Keep subgraph with highest weight
        if $w_A < w_B$ then $I' \setminus A$
        else $I' \setminus B$
    end
    return $I'$
end
```

5.5 Content Ordering

The output of the previous task is an unordered set of planning graphs. In order to produce natural language text from them, it is necessary to determine the order in which they will appear in the text. A straightforward criterion for sorting them would be to use some measure of importance, perhaps derived from the ranking scores $\boldsymbol{VR}$ associated with their vertices, as discussed in Section 5.2.3. However, this risks resulting in a text where consecutive statements have little or no relation to each other. Instead, we seek to establish an order that not only prioritizes relevant contents but also maximizes the semantic relatedness between pairs of consecutive graphs.

Our ordering strategy is based on the premise that a summary plan that groups similar information together facilitates the production of coherent summaries. This premise is supported by research on theoretical and computational linguistics that has shown how the flow and focus of information in texts contribute towards local coherence (Barzilay and Lapata, 2005). We argue that placing together in the text plan subgraphs with common lexical meanings is an effective mechanism towards producing a locally coherent and cohesive summary, i.e., a text where consecutive statements cover the same topic. This organization of contents also facilitates the generation of cohesive devices if NLG is used to produce an abstract from the plan, e.g., lexical chains, anaphoric expressions, conjunctions and ellipsis. In addition to local coherence and cohesive devices, placing multiple references to the same meaning in close vicinity in the text can be used to support strategies of information packaging during linguistic generation, e.g. thematic progressions.

Our sorting method consists in performing a traversal of a vertex-weighted, undirected graph where nodes correspond to planning subgraphs, vertex weights indicate the importance of each subgraph, and edges are established between pairs of subgraphs sharing at least one lexical meaning. Let $I = \{G_0, \ldots, G_k\}$ be a set of k subgraphs of a planning graph $G = (V, E, a, m)$ where each subgraph G_i has a set of nominal lexical meanings $nominal_i$ according to m, and w be the vertex weight function for G based on the ranking of vertices $\boldsymbol{VR}$. We create an undirected sort graph $G_{sort} = (V, E, \omega, \mu)$ where ω and μ are vertex and edge weighting functions, s.t.:

$$\begin{cases} V = I \\ E = \{(v_i, v_j) : v_i, v_j \in V \wedge G_i, G_j \in I \wedge nominal_i \cap nominal_j \neq \emptyset\} \\ \forall v_i \in V : \ \omega(v_i) = W(G_i(V_i, E_i)) = \frac{1}{|V_i|} \cdot \sum\limits_{v \in V_i} w_i(v) \\ \forall (v_i, v_j) \in E : \ \mu((v_i, v_j)) = \delta(\mathcal{T}(G_i), \mathcal{T}(G_j)) \end{cases}$$

Importance scores are assigned to graph-nodes that correspond to the average of the vertex weights of each planning graph, i.e., weights obtained from the ranking scores $\boldsymbol{VR}$ introduced in Section 5.2.3. Edges, on the other hand, are established between graph-nodes sharing at least a semantic type and are weighted according to the edit distance function δ introduced in Section 5.4.

We implement the traversal of G_{sort} using a greedy exploration algorithm, as shown in Algorithm 9. The algorithm starts from the highest weighted graph-node and incrementally adds graphs to a sequence of visited nodes. It ends once all vertices in the graph have been visited. Crucially, our problem formulation prioritizes adding graphs to the plan if they share at least one lexical meaning with those graphs already in the plan, as indicated by the edges of G_{sort}. At every step of the exploration, the algorithm chooses the graph-node with the highest combined weight – indicating its importance– and similarity relative to the graphs in the plan with which it shares at least one lexical meaning. If G_{sort} is not connected, the algorithm runs again from the highest ranked vertex which has not been visited yet.

Algorithm 9: Greedy exploration of a sort graph

```
Input: Graph $G_{sort} = (V, E, \omega, \mu)$
Output: Plan $P$
begin
    $v_{max} \leftarrow \arg\max_{v \in V}(\omega(v))$    // choose initial vertex
    $P \leftarrow (v_{max})$    // a plan is a sequence of visited nodes
    $A \leftarrow \{(v_{max}, v) : (v_{max}, v) \in E\}$    // candidate edges
    while $|P| < n$ do
        if $A = \emptyset$ then    // if graph is not connected
            $v_{max} \leftarrow \arg\max_{v \in V \wedge v \notin P}(\omega(v))$
        end
        $(v, v')_{max} \leftarrow \arg\max_{(v,v') \in A}(\omega(v') + \mu(v, v'))$    // choose edge
        add $v'$ to $P$    // extend plan
        // update candidate edges
        $A \setminus (v, v')_{max}$
        $A \leftarrow A \cup \{(v', v'') : (v', v'') \in E \ \wedge \ (v', v'') \notin A \ \wedge \ v'' \notin P\}$
    end
    return $P$
end
```

Ordering Applied to the Running Example

Figure 5.4 shows a sort graph obtained from three hypothetical subgraphs extracted from the planning graph of our example, which appear in the graph as vertices labeled v_1, v_2 and v_3. Undirected edges are established between subgraph-nodes sharing at least one nominal meaning. Thus, assuming that *his* and *John Major* corefer to the same meaning according to m, an edge is added between v_1 and v_2. Similarly, v_1 and v_3 are connected under the assumption that *British capital* and *London* are also assigned the same meaning.

Imagine that the first graph has the largest average weight $\omega(v_1)$. In that case, the traversal would start from v_1 and consider v_2 and v_3 as possible successors. If the combined weight of $\omega(v_2)$ and $\mu((v_1, v_2))$ is greater than that of $\omega(v_3)$ and $\mu((v_1, v_3))$, then the plan becomes the sequence (v_1, v_2). In a subsequent iteration of the main loop of Algorithm 9, the only remaining candidate edge connects v_1 to v_3, resulting in the latter being added to the plan (v_1, v_2, v_3). Note that according to this plan, the summary will start communicating contents related to *John Major* and will later shift the topic to *nuclear energy*.

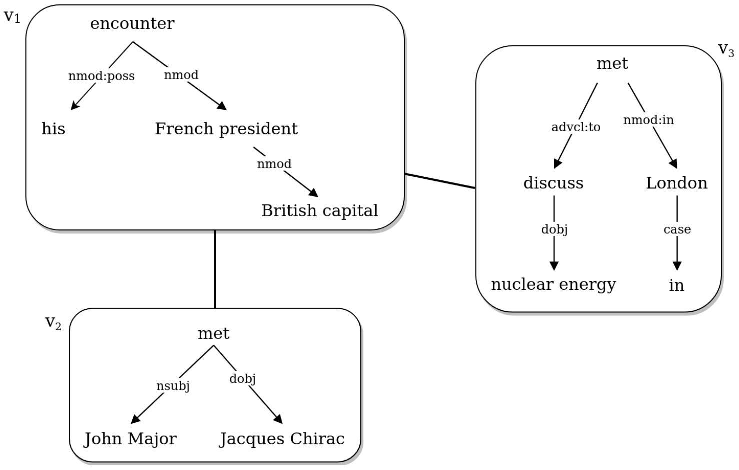


Figure 5.4: Sort graph with three subgraphs as vertices.

Chapter 6

EXPERIMENTAL EVALUATION

The approach to planning described in Chapter 5 involves addressing a number of tasks sequentially, each task starting from the output of the previous ones. In this chapter we present a system that adopts a pipeline architecture to implement this approach and a set of experiments designed to evaluate its performance at various stages of its execution.

The system being presented is based on the UD-based method for instantiating planning graphs described in Section 4.2. It implements each of the tasks introduced in Section 5.1, namely, content assessment, content selection, redundancy removal and content ordering. The pipeline architecture of the system, which comprises components for each one of the tasks listed above, is shown in Figure 6.1. The figure also indicates the representations produced by each component. Thus, the system starts from one or more documents containing natural language text and produces an analysis graph that is assessed and transformed into a planning graph. From this graph, content selection extracts a set of subgraphs that are filtered and sorted by the redundancy removal and content ordering components.

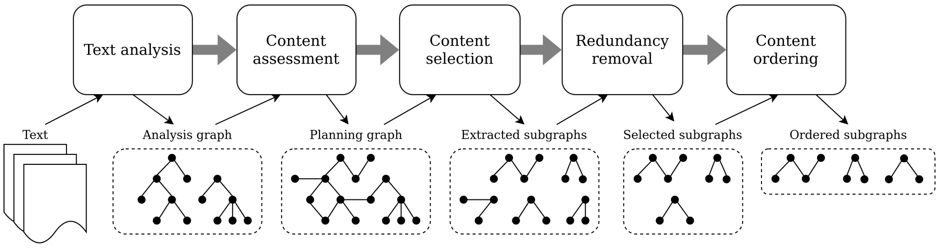


Figure 6.1: Pipeline architecture of our text planning system.

The evaluation described in this chapter comprises a mixture of intrinsic and extrinsic evaluation experiments on English materials. Where possible, we use existing evaluation datasets and compare with SoA systems. We emphasize the evaluation of content assessment and its subtasks, i.e., ranking of meanings, disambiguation and ranking of vertices, due to their importance to the overall approach. The following experiments will be presented:

1. A first set of experiments evaluates the ranking of meanings intrinsically by comparing the ranks produced by our component against those produced by a set of baselines. A small set of documents manually annotated with relevant meanings is adopted as a ground truth.

2. A second set evaluates the ranking and disambiguation of meanings applied to WSD and EL tasks separately, and then jointly, using existing datasets. The results of our system are compared with SoA systems and popular baselines.

3. A third set of experimets evaluates ranking intrinsically by comparing system ranks against words present in summaries belonging to existing summarization datasets, and comparing with baselines.

4. A fourth and final set evaluates extractive summaries produced with various versions of our system. This set of experiments, which constitutes an extrinsic evaluation of our approach, also uses existing

summarization datasets and compares with both SoA systems and popular baselines.

This chapter is organized as follows. Section 6.1 describes the experimental set up, while Section 6.2 shows some of variations of the system used in the experiments. Section 6.3 covers the first set of experiments evaluating the ranking of meanings while Section 6.4 covers the disambiguation experiments and their results. Finally, Section 6.5 presents the experiments for evaluating ranking of words and extractive summaries.

6.1 Experimental Setup

As mentioned above, we adopt the mechanism for instantiating planning graphs from sentence-level UD parses described in Section 4.2. More specifically, we use the following resources:

- the Stanford dependency parser (Chen et al., 2014) shipped with version 3.9.1 of the CoreNLP tools,
- version 4.0.1 of the BabelNet lexical database,
- English fastText pre-trained vectors (Grave et al., 2018),

While we only use English models and versions of the above resources, these cover a significant number of languages each. Thus, CoreNLP provides dependency models for six languages while corpora in other languages can be used to obain additional models. BabelNet supports 284 languages and pre-trained fastText vectors are available for 157 languages.

These resources are used in the implementation of the content assessment phase and applied to the creation of a weighted planning graph. Figure 6.2 depicts the internal architecture of the assessment component, which reflects the subtasks introduced in Chapters 4 and 5, namely, text analysis, ranking of meanings, disambiguation, instantiation of a planning graph and vertex ranking.

Thus, the Stanford parser implements the text analysis subtask and produces an unconnected analysis graph consisting of UD trees of sentences in the input. The algorithm for collecting candidate meanings described in Section 5.2 uses the BabelNet database as the L_M dictionary required to obtain a set of candidates C_S. These candidates constitute the input to the meaning ranking algorithm, and the resulting rank $\boldsymbol{MR}$ is passed to the disambiguation algorithm to produce a meaning function m for vertices in the analysis graph, which is in turn applied to the creation of a planning graph. Finally, both $\boldsymbol{MR}$ and m are used by the vertex ranking algorithm to produce a ranking of vertices $\boldsymbol{VR}$ and the corresponding weight function w.

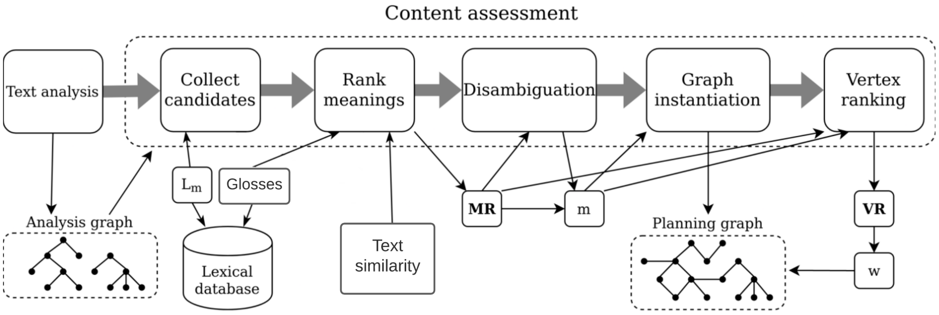


Figure 6.2: Architecture of the content assessment component.

In a similar fashion to content assessment, redundancy removal and content ordering also involve calculating pairwise similarities across meanings. Taking advantage of the fact that the set of meanings to compare is the same across all components, we cache the similarity values obtained from comparing pairs of meanings during the assessment phase and reuse them in the redundancy and ordering phases. This allows us to implement both phases without depending on any of the resources listed above.

The overall system takes a number of parameters that control the behavior of the algorithms described in Chapter 5. Table 6.1 lists these parameters along with the algorithms in which they are used, and the values they take in our experiments. The values have been set experimentally using two development sets. A first set comprising five documents drawn at random from the disambiguation dataset described in Section 6.4.1 is used to set the parameter values pertinent to the ranking and disambiguation of meanings, i.e., the first three parameters in Table 6.1. The second set has ten documents picked at random from the two summarization datasets described in Section 6.5, five from each, and is used to set the values for the following five parameters.

6.2 System variations

The $context$ and $similarity$ functions introduced back in Section 5.2.1 play an important role in our approach. The former compares pairs of a candidate meaning and the context of its mention in the text to assess how good of a match they are, while the latter compares pair of meanings in terms of semantic similarity. Due to their impact –they are used directly for content assessment and redundancy removal, and their values determine the results of all other steps– we have experimented with multiple formulations of both functions.

Most of these formulations involve obtaining meaning glosses from the dictionary L_m, BabelNet in our set up, and comparing them using textual similarity methods. In the case of $context$ applied to a candidate (m, e), the gloss of the meaning m is compared with the sentence containing the mention e, which constitutes the context of the candidate. This gloss-based approach is motivated by the requirements set in Section 1.3 and, more specifically, the goal of using methods, tools and resources that can be applied to a wide number of languages, and that are not tied to specific text genres or domains. On one hand, the textual similarity methods we use work with texts in multiple languages. On the other, using methods based on comparing glosses is crucial to make them applicable to meanings from a variety of lexical databases because such databases usually contain textual definitions of their meanings, thus guaranteeing that our approach can work with a wide range of topics.

Parameter	Description	Used in	Value
k	Maximum mention size	Algorithm 1	5
c_{min}	$context$ value minimum threshold	Algorithm 2	0.6
$d_{meaning}$	Damping factor of meaning ranking		0.1
l	Stopping threshold for ranking algorithm	Algorithm 3	$1 \cdot 10^{-5}$
$d_{vertices}$	Damping factor of vertex ranking	Algorithm 5	0.3
μ	Decay factor	-	3 *
λ	Normalization coefficient of subgraph extraction	Algorithm 7	3
c	Cost function for edges in subgraph extraction		**
m	Number of subgraphs to extract		1000
d_e	Balancing factor for rename operations in redundancy removal	Algorithm 8	0.9
n	Final number of subgraphs after redundancy removal		***

* This additional parameter is introduced in Section 6.5
** Set to the average word vertex rank
*** Takes different values depending on target summary length

Table 6.1: Parameters to the text planning system

Below is the list of alternative methods that can be applied to implement both *context* and *similarity* by comparing glosses with each other and with contexts. Whenever a method requires word embeddings, we use fastText.

- *BoW* calculates a Bag-of-Words (BoW) average of vectors of words in each sentence. The resulting averaged vectors are then compared using cosine distance. Calculating sentence vectors as the arithmetic mean of word vectors is a simple method to assess similarity between sentences, which has been shown to be effective (Perone et al., 2018).
- *SIF* calculates the Smooth Inverse Frequency (SIF) (Arora et al., 2017) average of embeddings of words in each sentence, where each vector is inversely weighted by the frequency in general corpora of the word it corresponds to. This results in more common words having a lower contribution to the final average while favoring less common and more informative words. In addition, common component removal is performed on the averaged vector as a means of denoising the averaged vectors, which are then compared using cosine distance. SIF has been shown to perform significantly better than BoW on textual similarity tasks.
- *WMD* is based on Word Mover’s Distance (WMD) (Kusner et al., 2015), a method that is also based on word embeddings but that, instead of averaging all word vectors, computes the sequence of edits with minum cost to transform one sentence into the other. The cost of each edit in the sequence is determined by the Euclidean distance between the word embeddings of the two words involved. Like BoW and SIF, WMD has also been shown to perform well in tasks involving comparison between texts.
- *SBERT* uses embeddings for whole sentences obtained from Bidirectional Encoder Representations from Transformers (BERT) models (Devlin et al., 2019). We employ the models and neural networks by Reimers and Gurevych (2019), which produce vector rep-

resentations that are close for semantically similar texts, can be inferred with reduced computational cost compared to regular BERT models, and can be compared using cosine distance. In addition, these models are available for over 100 languages. In our experimental set-up we use the *paraphrase-distilroberta-base-v1* model for English texts [1].

While looking for ways to improve the performance of our approach, we also experimented with additional implementations of $context$ and $similarity$. Unlike the implementations listed above, these additional implementations do not meet our requirements due to limited support for languages other than English or being constrained to specific lexical databases. Nevertheless, we include them in our experiments to be able to compare a wide range of SoA methods:

- *MLMS* is an implementation of $context$ that, given a candidate (m, e), the sentence s where e appears and the set of lexicalizations of m present in BabelNet, creates a set of sentences by replacing e in s by each of the alternative lexicalization of m, excluding e. The resulting set of sentences are evaluated with a language model and the maximum of all scores is used as the value of $context$. We use Masked Language Model Scoring (MLMS) by Salazar et al. (2020) and the OpenAI GPT-2 model (Radford et al., 2019) to score each sentence [2].

- *glossBERT* is another implementation of $context$ based on a binary classifier by Huang et al. (2019), who fine-tunned a BERT model

[1] We found out in preliminary tests that this model performed better than SBERT models fined tuned with semantic similarity datasets.

[2] While MLMS supports a number of other models including BERT, we found the performance to be much faster with GPT-2. This is arguably due to the fact that most transformer-based models such as BERT are not traditional language models while GPT-2 is. The latter is auto-regressive in the sense that each token in the sentence has the context of the previous word, while obtaining sentence scores with BERT involves masking each word in the sentence. Unfortunately, multilingual support in GPT-2 is reduced compared to BERT.

with pairs of gloss and sentence taken from the SemCor 3.0 corpus of English texts annotated with WordNet senses. We use the classification scores of their sentence-based model without weak supervision as values of $context$. Despite producing slightly lower results than other models presented by the authors, this model does not require that the word to be disambiguated is marked and is arguably better suited to generalize to words and meanings in other lexical databses [3].

- *SEW* is an implementation of $similarity$ that results from calculating the cosine similarity between pairs of SEW-Embed sense embeddings (Bovi and Raganato, 2017). SEW-Embed vectors are trained from a dump of the English Wikipedia annotated with BabelNet 3.0 senses. While these vectors are language-independent, they are only defined for lexical meanings in this specific version of BabelNet.

6.3 Evaluation of Ranking of Meanings

The goal of the evaluation described in this section is to assess the performance of the function $\boldsymbol{MR}$, introduced in Section 5.2.1, for the task of ranking meanings according to their importance in a text. This evaluation also serves as an assessment of the performance of the multiple implementations of the $context$ and $similarity$ functions applied to the ranking of candidate meanings. As a result of this assessment, we will select some implementations to be used in the rest of the experiments.

To the best of our knowledge, there is no publicly available dataset or gold standard with texts annotated with the most prominent meanings communicated in them. For this reason, we rely on a manually annotated corpus

[3] While GlossBERT is an English-model, it should be possible to fine-tune the classifier from a multilingual BERT model to obtain multilingual support. Similarly, while GlossBERT is trained for WordNet senses only, it could be extended with additional training data annotated with meanings from other lexical databases. The performance of such extensions remains an open question.

containing English texts and created specifically for this evaluation. We employ IE metrics to compare the ranks produced by our system against several baselines.

Data

Our corpus for evaluating meaning ranks comprises a subset of 5 document-pairs, drawn at random from the CNN/Daily Mail corpus (Hermann et al., 2015; Nallapati et al., 2016), where the summaries have been manually annotated with lexical meanings. The annotation was carried out by two annotators following a set of annotation guidelines, shown below, which provide guidance on how to annotate mentions [4] with meanings in a lexical database L_M. BabelNet 4.0.1 was used for the annotation.

The annotation guidelines provide annotators with criteria to choose what expressions in the text to look up in L_M and to decide the best meaning from all candidate meanings. In general, mentions can only be annotated with meanings returned by L_M when looking up their word forms or lemmas. This restriction is put in place to stop annotators from considering meanings inferred from the context or from world knowledge, as these inferred meanings are not only beyond the capacities of our disambiguation component but can also lead to a very large number of meanings being annotated. Anaphoric expressions are also excluded from the annpotation to focus the evaluation on the performance of the disambiguation component. The guidelines allow expressions to be annotated with multiple meanings if more than one candidate is deemed suitable, but overlapping annotations are forbidden.

The gold standard used in the evaluation is a consensus annotation created from the annotations of both annotators, who agreed on 79% of the annotations [5]. It contains 87 nouns annotated with their meanigns, 32 verbs,

[4] Recall from Section 5.2.1 that a mention is a sequence of one or more consecutive words in a sentence.

[5] It is worth considering that annotation task did not involve selecting prominent meanings, but only annotating the lexical meanings in BabelNet they considered cor-

12 adjectives, 8 adverbs and 16 MWEs.

Annotation guidelines

Given a mention e and a lexical database L_M:

1. Consider e for annotation if it belongs to any of the following categories:
    - Main verb
    - Adverb
    - Adjective
    - Quantifier or number
    - Symbol, e.g., €
    - Compound
2. Ignore e if it belongs to any of the following categories:
    - Determiner or pronoun except quantifier and number
    - Auxiliary verb
    - Conjunction
    - Preposition
    - Idiom or temporal expression
    - Non-consecutive words, e.g., *blue juice* appearing in the text as *blue iced juice*
3. Given $C_e = L_M(e)$, consider e for annotation if $C_e =\neq \emptyset$. When looking up e in L_M:
    - e can be looked up using the form in the text or lemmatized, e.g., *euros* vs *euro*, *green energies* vs *green energy*.
    - Do not distinguish meanings by grammatical category, even if L_M does, e.g., if e is *fast* then C_e includes both its adjectival and verbal meanings.

rect for words in the summaries

4. Annotate e with those candidate meanings in C_e that constitute either a match or a reasonable approximation to its meaning in the text.
    - e can be left unannotated if there are no suitable meanings in C_e.
    - e can be annotated with multiple meanings in C_e if more than one is judged to be suitable.
    - Do not attempt to annotate e with any meanings that are not found in the result C_e of looking up e in the dictionary, even if they can be unequivocally inferred from the context or from world knowledge, e.g., do not annotate *loosing side* with concept 'loser' or *Ford Fiesta* with concept 'Car'.
5. If two mentions e_1 and e_2 overlap and both have suitable meanings, prefer the longest mention, e.g., prefer *emergency break* over *emergency* or *break*.
6. If e is a MWE without a suitable meaning in L_M, do not annotate any parts of it whose meaning does not contribute compositionally to the MWE, e.g., if L_M has no entry for *Golden Gate*, do not annotate *Golden* nor *Gate*.

Metrics

Metrics used in evaluations of ranking tasks usually assess the quality of a rank in relation to some ground truth Liu (2011). Some metrics require ranks of a fixed length k, e.g., Discounted Cumulative Gain (DCG) and Normalized Discounted Cumulative Gain (NDCG), while others focus on the first relevant item in a rank, e.g., Mean Reciprocal Rank (MRP), or require a ground truth consisting of full ranks, e.g., Spearman's correlation. Considering that (i) our ranks can be over an arbitrary number of candidate meanings, (ii) our ground truth consists of sets of unranked items [6] and (iii) we want to evaluate all items in a rank, we adopt Average Precision (AP) and its generalization to multiple ranks Mean Average Precision (MAP) as the evaluation metrics for our task.

We calculate MAP as follows. Given an ordered sequence of n candidate meanings $R = (m_0, \ldots, m_n)$ for a document D, and a set of k meanings $G = m_0, \ldots, m_k$ annotated in a summary of D, then AP for D is:

$$AP_D = \frac{\sum_{i=0}^{n} P(i) \cdot rel(m_i)}{\sum_{i=0}^{n} rel(m_j)}$$

where precision P at position i is:

$$P(i) = \frac{\sum_{j=0}^{i} rel(j)}{i}$$

and the relevance function rel at position i is:

$$rel(i) = \begin{cases} 1 & \text{if } m_i \in G \\ 0 & \text{otherwise} \end{cases}$$

Once the AP value for each document in our ground truth is known, the value of the MAP metric for the whole dataset corresponds to the average of the document AP values.

Baselines and systems

We compare the ranks produced by a number of variations of our system and two baselines. Three versions of our system are evaluated:

- CTX ranks meanings based on the $context$ function alone,
- SIM ranks meanings based on the $similarity$ function alone and
- TP uses the full ranking method based on both functions.

[6] We do not rank the meanings annotated in each summary.

The purpose of evaluating these versions is to measure the contribution of the two terms of Equation (5.1). In addition, we also run variations using each of the functions listed in Section 6.2, namely *BoW*, *SIF*, *WMD*, *SBERT* and *MLMS* for both $context$ and $similarity$, *MLMS* and *GlossBERT* for $context$, and *SEW* for $similarity$.

Regarding the baselines, the first one produces a random ranking while the second ranks candidate meanings according to the number of mentions that have them as candidates. We refer to them as *random* and *frequency* respectively.

Experiments and results

All baselines and variations of our system are run on the set of five documents of our ground truth, resulting in five sets of five rankings. We calculate the AP score for each ranking, and then average the results for each five-ranking set to obtain MAP scores. In addition, we also calculate the average precision score after stopping at element k in the ranking, k being the number of meanings annotated in the corresponding gold summary. This MAP$_k$ score focuses on the top meanings of the rank, which are more likely to appear in a system summary.

In a first experiment, we evaluate the implementations of $context$ using two alterative notions of what context is, one where the it corresponds to the sentence containing a candidate’s mention –*local* context– and another where the context is the whole document –*global* context. Table 6.2 [7] shows the MAP and MAP$_k$ scores, with the left column containing local context scores and the right one global context scores. Note that the last two variations CTX-MLMS and CTX-GlossBERT do not have global context scores as they operate at sentence level by definition. They are also the best performing in terms of MAP scores. The limitations in terms of computational speed of CTX-MLMS and the better results of CTX-GlossBERT drive us to pick the latter as our implementation of

[7] We do not break down results by grammatical category due to the samll size of the corpus.

	Local		Global	
	MAP	MAP_k	MAP	MAP_k
Random	.02	.04	.05	.06
Frequency	.01	.02	.01	.03
CTX-BoW	.022	.082	.021	.104
CTX-SIF	.025	.114	.015	0
CTX-WMD	.021	.225	.02	.02
CTX-SBERT	.032	.063	.04	.066
CTX-MLMS	.076	.147	-	-
CTX-GlossBERT	**.099**	**.218**	-	-

Table 6.2: MAP values for various implementations of the *context* function

choice for successive experiments.

We follow this first experiment with a second one where we evaluate alternative methods for implementing *similarity*, both on their own [8] (with prefix SIM) and combined with a GlossBERT-based implementation of *context* (with prefix TP). The results, shown in Table 6.3, indicate that SBERT is the best performing implementation, both by itself (SIM-SBERT) and in combination with GlossBERT (TP-GlossBERT-SBERT). It it worth noting, however, that TP-GlossBERT-SEW does not lag far behind. For this reason, we will include both in our experiments. We will also include BoW as a baseline for more elaborate methods.

[8] When evaluating *similarity* on its own, we set c_{min} to 0 so that all candidates are ranked.

	MAP	MAP_k
SIM-BoW	.077	.28
SIM-SIF	.069	.164
SIM-WMD	.07	.166
SIM-SBERT	**.116**	**.266**
SIM-SEW	.088	.12
TP-GlossBERT-BOW	.117	.265
TP-GlossBERT-SIF	.129	.302
TP-GlossBERT-WMD	.11	.268
TP-GlossBERT-SBERT	**.152**	**.351**
TP-GlossBERT-SEW	.142	.309

Table 6.3: MAP values for for various implementations of the *similarity* function

6.4 Evaluation of Disambiguation

We conduct two separate evaluations to assess the performance of our approach at disambiguating mentions. While disambiguation is not the main focus of this thesis, we use this assessment as means to validate the first research goal stated in Section 1.3, i.e., that our text planning strategy can be effectively used both for summarization and for other tasks, and that this constitutes an advantage over seq2seq methods for summarization.

First, we evaluate disambiguation using the unified evaluation framework for WSD created by Navigli et al. (2017), which contains five datasets for evaluation of WSD systems. Then we use the SemEval-2015 Task 13 dataset (Moro and Navigli, 2015) to evaluate our system jointly for WSD and EL tasks. All experiments are based on texts in English and are evaluated using precision and recall metrics.

6.4.1 WSD evaluation

Recall from Section 2.2 that WSD is the task of choosing the right lexical meanings for content words from sets of candidate meanings obtained from a dictionary of word senses. We use the evaluation data, metrics and baselines of the evaluation framework created by Navigli et al. (2017). This framework is the result of merging multiple evaluation datasets based on different versions of WordNet into a single unified dataset where original annotations have been mapped to WordNet 3.0 using semiautomatic methods.

Data

The evaluation data comprises 7254 WordNet 3.0 sense annotations for nouns, verbs, adverbs and adjectives found in 26 documents belonging to multiple domains, ranging from news to fiction and biomedical. In total, the corpus contains 4353 different senses for 3663 lemmas. The annotated tokens have an average $6\dot{2}2$ candidate senses, and are also annotated with PoS tags and lemmas.

Metrics

Evaluation of WSD systems is conducted by comparing system senses to those in the evaluation data. System sense annotations that have a matching annotation in the gold are interpreted as true positives, while those that do not match are false positives. Words annotated with a sense in the gold but not in the system output correspond to false negatives, while words annotated by the system but not in the gold are ignored by the evaluation script and therefore do not count towards false positives. The resulting counts are used to obtain precision, recall and F1 values.

Baselines and systems

The evaluation data includes annotations produced by three baselines. The first one is a *random* strategy, the second selects the sense occurring the highest number of times in the evaluation data and the third selects the first candidate sense according to WordNet 3.0. We refer to the last two baselines as Most Frequent Sense (MFS) and WordNet First Sense (WFS) respectively. As done in the evaluation of meaning ranks, we also run *CTX*, *SIM* and *TP* variations of our system in combination with some of the functions listed in Section 6.2 –we pick combinations of BoW, SBERT, SEW and GlossBERT as explained in Section 6.3.

Experiments and results

Table 6.4 shows the F1 values for the three baselines and the variations of our system. Baseline results are shown in the top rows of the table, followed by two implementations of $context$, CTX-BoW and CTX-SBERT [9], three implementations of $similarity$, SIM-BoW, SIM-SBERT and SIM-SEW, and three versions of our ranking method combining GlossBERT with BoW, SBERT and SEW, respectively. At the bottom of the table are the results of the two best-performing systems as reported by Navigli et al. (2017), the supervised system *IMS*$_{-s}$*+emb* and the knowledge-based *UKB_gloss**. The last row corresponds to the GlossBERT(Sent-CLS) classifier. Results for MFS baseline and the best-performing systems are obtained from Navigli et al. (2017) and from the framework's website[10]. The results for the WFS come from implementing and running it ourselves.

Our approach to ranking meanings based on GlossBERT predictions manages to perform slightly better than the base GlossBERT classifier on four out of five datasets. While the improvements for this WSD are small, they are nevertheless a welcome by-product of our ranking of meanings as a first step towards summarization. SEW is the best textual similarity method when using ranking on its own (SIM-SBERT) and in combination with GlossBERT (GlossBERT-SEW), but the difference between SEW and SBERT (GlossBERT-SBERT) is marginal. The performance in the disambiguation evalaution comes fundamentally from the GlossBERT

[9] SEW cannot be used to compare meanings with mention's context, and Glo

[10] http://lcl.uniroma1.it/wsdeval/

	Nouns	Verbs	Adjectives	Adverbs	SE2	SE3	SE07	SE13	SE15	All
Random	33.5	22.6	44.5	52.7	40.6	35.9	26	40.8	42.3	33.2
MFS (SemCor)	-	-	-	-	65.6	66.0	54.5	63.8	67.1	62.9
WFS	66.6	50.0	69.6	69.5	66.7	66.0	55.5	62.1	67.7	63.3
IMS_s+emb	72.0	56.5	76.6	84.7	72.2	70.4	62.6	65.9	71.5	69.6
UKB_gloss*	-	-	-	-	68.8	66.1	53.0	68.8	70.3	67.3
GlossBERT					77.1	74.5	68.7 (*)	73.1	**79**	75.5 (*)
CTX-BoW					44.5	40.0	30.5	50	48.2	44.1
CTX-SBERT					53.4	47.3	38.2	53.7	58.1	51.5
SIM-BoW					51.2	44.9	36.6	52.3	52.4	51.7
SIM-SBERT					52.3	46.4	41.6	53.9	50.1	53.6
SIM-SEW					52.7	51.1	42.1	51.7	57.3	57.3
GlossBERT-BoW					70.9	74.5	68.9	72.7	76.5	74.5
GlossBERT-SBERT					**77.3**	**75.1**	**69.3**	74.1	78.1	75.6
GlossBERT-SEW					**77.3**	**75.1**	**69.3**	**74.6**	77.8	**75.7**

Table 6.4: F1 scores on the Unified WSD Evaluation dataset

model, with the improvements by alternative functions of *similarity* being very similar regardless of the method used to implement the latter function.

6.4.2 Joint WSD and EL evaluation

The SemEval-2015 Task 13 dataset (Moro and Navigli, 2015) was created to allow a joint evaluation of WSD and EL in multiple languages, and contains annotations of WorNet senses, Wikipedia pages and BabelNet synsets, the latter subsuming both word senses and named entities. We base our evaluation on the BabelNet annotations to obtain an estimation of how well does our disambiguation strategy perform when applied jointly to both tasks. As in our previous evaluation, we calculate precision and recall-based metrics, and compare our approach against existing SoA systems.

Data

The dataset consists of a parallel corpus of 4 documents in English, Italian and Spanish, and belonging to multiple domains: medical, scientific and social issues. All documents are manually annotated with WordNet 3.0 senses, Wikipedia pages and BabelNet 2.5.1 synsets. The English version of the documents are annotated with 1094 single-word and 81 multiword annotations, with an average number of meanings per annotation of 8.1. Two annotators participated in the annotation of each language, and agreed on an average of 68% of the annotations.

Metrics

We evaluate with the same precision, recall and F1 metrics used for the task. They are computed exactly in the same way as in the previous WSD evaluation but counting BabelNet annotations instead of WordNet.

Baselines and systems

Moro and Navigli (2015) introduced a BabelNet First Sense (BFS) baseline, similar to the WFS baseline but based on the first synset returned by BabelNet for a given mention. We report results of this baseline and also of the *random* baseline. In addition, we also run the evaluation with the CTX, SIM and TP variations of our system combined with BoW, SBERT, SEW and GlossBERT. In contrast to other experiments, our system uses version 2.5.1 of BabelNet instead of 4.0.1 to conform to the annotations found in the SemEval dataset.

Experiments and results

Table 6.5 shows the F1 scores for the baselines and variations of our system, as well as GlossBERT and the LIMSI system that obtained the best results for English in the SemEval task. The results are in line with those of the previous experiment with the Unified WSD Framework. GlossBERT is largely responsible for high performance but adding a ranking step results in some gains in performance, particularly when using SEW to implement $similarity$. Interestingly, the results for NEs are very high both GlossBERT and our system variations. This suggests that the GlossBERT model is capable of generalizing to meanings –and their glosses–

	All	NEs	Word senses				
			All	Nouns	Verbs	Adjectives	Adverbs
Random							
BFS	67.5 (66)	85.7	66.3/64.2 *	66.7	55.1	82.1	82.5
LIMSI	65.8	82.9	64.7	64.8	56.0	76.5	79.5
GlossBERT	73.9	**92**	72.6				
CTX-BoW	38.3	81	35.2				
CTX-SBERT	36	83.4	36.4				
SIM-BoW	43.8	87.1	40.7				
SIM-SBERT	49.9	88.3	46.3				
SIM-SEW	47.2	89.6	42.8				
TP-GlossBERT-BoW	73.2	**92**	71.8				
TP-GlossBERT-SBERT	73.7	**92**	72.3				
TP-GlossBERT-SEW	**74.2**	**92**	**73.0**				

* Running the WFS baseline we obtained a 63.3 precision score, lower than the 64.2 score reported in Navigli et al. (2017).

Table 6.5: F1 results on the English dataset of the SemEval-2015 Task 13 evaluation.

in datasets other than WordNet.

6.5 Evaluation of Extractive Summaries

In the following, we present an extrinsic evaluation of our approach applied to the task of obtaining extractive summaries. Our evaluation is based on datasets and evaluation metrics widely used in AS, which allows us to compare with popular baselines and existing systems. We evaluate versions of our system aimed at measuring the contribution of each of the main components: ranking of meanings, ranking of vertices, subgraph extraction and redundancy removal.

Data

We evaluate with two summarization datasets comprising newswire articles and human-crafted summaries. The first one is a large corpus of news stories originally created for QA (Hermann et al., 2015) and later adapted to AS (Nallapati et al., 2016), usually referred to as the CNN/Daily Mail dataset. This corpus comprises 311,672 documents of 766 words in average that are associated with single-document abstractive summaries (53 words in average), one each. The second dataset is the DUC 2002 corpus, which contains 60 sets of about 10 news articles each. Each one of the sets has multi-document extractive and abstractive summaries of various fixed lengths, and also single-document abstracts for each of the articles. Articles have 546 words in average, while single-document abstracts have a maximum length of 100 words.

Experiments are based on a subset of the CNN/Daily Mail dataset containing 500 document-abstract pairs drawn at random, and the full set of 579 document-abstract pairs from the DUC2002 dataset. As mentioned in Section 6.1, a development set of 5 pairs taken randomly from each of two datasets has been used to establish the parameter values for the system.

Metrics

Our evaluation uses ROUGE metrics (Lin, 2004b), popularized in the field of AS by the DUC and Text Analysis Conference (TAC) conferences. ROUGE metrics measure recall of n-grams in a system-generated summary with reference to a set of gold summaries. Recall metrics have long been favored by practitioners over precision-oriented metrics such as BLEU because the variability in the selection of sentences between human annotators makes precision overly strict (Nenkova and McKeown, 2011). We report ROUGE-1, ROUGE-2 variants since they have been shown experimentally to have the strongest correlation with human judgments (Lin, 2004a; Owczarzak and Dang, 2011). In addition, we also report ROUGE-L as it is often included in evaluations of extractive systems. Measuring n-grap overlap, as done by ROUGE metrics, gives an estimation of how informative a system summary is, but does not address other aspects such as readability, grammaticality or coherence.

Baselines and systems

We evaluate a number of strategies for composing extractive summaries based on our approach and designed to assess the individual contributions of our components:

1. *Meaning word rank (MWR)* ranks words based on the **MR** rank of their disambiguated meanings.
2. *Vertex word rank (VWR)* ranks words based on the **VR** rank of their aligned vertices.
3. *Subgraph word rank (SWR)* ranks words on the the **VR** rank of their aligned vertices if they are part of an extracted subgraph.
4. *Redundancy word rank (RWR)* ranks words on the the **VR** rank of their aligned vertices if they are part of a subgraph selected after redundancy removal.
5. *Meaning sentence rank (MSR)* ranks sentences based on the average **MR** rank of their disambiguated meanings.
6. *Vertex sentence rank (VSR)* ranks sentences based on the average **VR** rank of their aligned vertices.
7. *Subgraph sentence rank (SSR)* ranks sentences on the the average **VR** rank of their aligned vertices if they are part of an extracted subgraph.
8. *Redundancy sentence rank (RSR)* ranks sentences on the the **VR** rank of their aligned vertices if they are part of a subgraph selected after redundancy removal.

The first four systems compose a summary by concatenating the top k words according to the word-level ranks they produce, while systems in the second group compose a summary using the top k sentences. Given that the first group produces unconnected sequences of words, we evaluate them using unigram ROUGE only. Sentence-level extractive summaries are evaluated with ROUGE-1, ROUGE-2 and ROUGE-L. Please note the content ordering is not included, given that the ROUGE metrics used in this evalaution would not reflect the ordering.

MWR and MSR follow the approach described in Sections 5.2.1 and 5.2.2 for ranking and disambiguating meanings, while VWR and VSR follow the ranking of vertices described in Section 5.2.3. SWR and SSR the content selection method based on extracting subgraphs from a planning graph as detailed in Section 5.3, Finally, RWR and RSR follow the method for selecting non-redundant subraphs described in Section 5.4. All of them are based on the experimental set up described in Section 6.1 and use the parameter values listed in Table 6.1. Ranking of meanings $\boldsymbol{MR}$ is implemented with GlossBERT as the $context$ function and SBERT as the $similarity$ function, given that this combination produces good overall results in the disambiguation evaluations of the preceding sections [11].

A popular and hard to beat baseline for summarization systems is the *lead-k* baseline, which takes the first k sentences of a document as its summary. Its effectiveness stems from the fact that, in news articles and in many other domains, authors communicate the most important information at the beginning of a document. We use a *lead-3s* baseline to compare against sentence-based extractive summaries, and a *lead-kw* baseline taking the first k words of the original document to compare against word-based extractive summaries.

We also compare the strategies listed above with the results of two SoA graph-based extractive systems, URANK (Wan, 2010) and TGRAPH (Parveen et al., 2015), and three recent neural extractive systems, the summarizer by Cheng and Lapata (2016) (Cheng2016), the SuMMaRuNNer system (Nallapati et al., 2017) and the system by Liu and Lapata (2019) (LIU2019). All systems except for the last one report results for the DUC2002 dataset, while the three neural systems report results for the CNN/Daily Mail

[11] While SEW sense embeddings have a slight edge over SBERT in the evaluations, the latter can generalize to unseen meanings and for this reason we choose it over the former.

corpus. Cheng and Lapata (2016) and Nallapati et al. (2017) use the anonymized version of the CNN/Daily Mail dataset where NEs have been replaced with anonymous identifiers that are ignored at evaluation time. In contrast, we follow Liu and Lapata (2019) and evaluate using the non-anonymized –and more challenging– dataset. This means that results for our system reflect its ability to select relevant NEs.

Experiments and results

Recall fromSection 5.2.3 that the ranking **MR** uses the edges in the planning graph to transfer relevance across words in the text. In our experimental set up, edges in the planning graph correspond to UD dependencies obtained from the Stanford dependency parser. When running the evaluation of our system using the DUC2002 and CNN/Daily Mail datasets, we obtained very low scores for all strategies except meaning-based MWR and MSR, which suggested poor performance of the word ranking **VR** component. A closer look revealed that low ROUGE scores were influenced by errors in the parsing [12].

For this reason, we replace the indicator function adj used to calculate **VR** with a function that returns a value in the range $[0..1]$ and which reflects the distance in number of tokens between words aligned with two vertices. More precisely, given an offset function that maps a vertex in a graph v to the position in the text of the word or first word of the multiword aligned with it, we use an exponential decay function to implement the alternative adjaceny function:

$$adj\text{'}_G(x, y) = \frac{1}{e^{\frac{(|\,\mathit{offset}(x) - \mathit{offset}(y)|) - 1}{\mu}}}$$

[12] When the parser fails to correctly analyze a complex sentence, it tends to produce an analysis where a single word governs over a large number of other words. This situation, which arises when subtrees parsed from parts of long sentences have to be integrated into a single dependency parse, results in certain words of a document having a large number of edges in the graph. Consequently, they receive a higher rank, miguiding the selection process.

	DUC2002	CNN/Daily Mail	
	ROUGE-1 75 words	ROUGE-1 75 bytes	ROUGE-1 75 words
Lead	**39.03**	**21.42**	**46.33**
MWR	37.15	6.12	41.72
VWR	36.16	9.52	38.80
SWR	37.50	11.85	41.40
RWR	37.24	11.99	41.48
Chen et al’16 WE	27	15.7	-

Table 6.6: ROUGE scores for word extractive summaries of the DUC2002 and CNN/Daily Mail datasets.

Applying this function, where μ is a decay factor controling how fast adjacency values decrease with distance, yields improved results for the ranking of vertices and subsequent tasks compared to the former dependency-based adjacency function [13]. Note that syntactic dependencies produced by the parser are still used to add edges to the planning graph and are therefore also applied to extract subgraphs.

In a first experiment, we evaluate our word-extractive versions, MWR, VWR, SWR and RWR on both the DUC2002 and CNN/Daily Mail datasets, and compare with the lead baseline and with the word-extractive version of Chen et al. (2016). The results, shown in Table 6.6, are obtained from system summaries of a fixed length. For DUC2002, we set the length to 75 words following the DUC guidelines. For the Daily Mail/CNN, we use 75 bytes to be able to compare with Chen et al. (2016) and 75 words to compare with our results on the DUC2002 dataset. In all cases, we evaluate with ROUGE-1 recall with stemming and stop word removal [14].

The results indicate that our system scores below the lead baseline for both datasets. For the DUC2002 dataset, however, we manage to obtain

[13] We also tried using distances in the dependency trees obtained using a shortest path algorithm, but the effect of wrong parses persisted.

[14] The summaries are made up of extracted content words, so it would make little sense to evaluate comparing stop words or word sequences.

	ROUGE-1	ROUGE-2	ROUGE-L
lead	41.99	20.67	37.55
MSR	34.06	12.23	29.71
VSR	33.87	11.86	29.70
SSR	30.63	9.59	26.79
RSR	31.35	10.22	27.45
URank **	**48.5**	21.5	-
TGraph **	48.1	**24.3**	-
Chen et al’16 SE	47.4	23.0	**43.5**
SummaRuNNer	46.6	23.1	43.0

** Maximum length is 100 words.

Table 6.7: ROUGE scores for sentence extractive summaries of the DUC2002 dataset.

better score than Chen et al. (2016). Ranking of vertices (MWR) results in a slight drop in performance of the 75-word summaries, but actually improves ROUGE-1 scores in the much shorter 75-byte summaries. Extracting subgraphs and removing redundant contents (SWR and RWR) give the best results for the 75-byte CNN/Daily Mail summaries, while they improve results over ranking of vertices in the 75-word summaries. This suggests that, while ranking words may hurt ROUGE-1 scores, it also lays the ground for improvements in the following steps. Unfortunately the differences between scores are not significative enough to reach a firm conclusion.

A second experiment involves evaluating our sentence-extractive versions, MSR, VSR, SSR and RSR, on the same two datasets. To facilitate comparison with other systems, we truncate our DUC2002 summaries at 75 words and evaluate with ROUGE recall. CNN/Daily Mail summaries contain three whole sentences and, given that they do not have a fixed length, are evaluated with ROUGE-1 F1 instead of recall. In both cases,

	ROUGE-1	ROUGE-2	ROUGE-L
Lead-3s	40.04	17.48	36.31
MSR	32.85	12.13	29.62
VSR	32.98	11.80	29.66
SSR	29.10	9.17	26.14
RSR	29.30	9.30	26.36
SummaRuNNer *	39.6	16.2	35.3
Liu Lapata’19	**43.85**	**20.34**	**39.90**

* Results obtained from the anonymized version of the dataset

Table 6.8: ROUGE F1 scores for sentence extractive summaries of the CNN/Daily Mail dataset.

we use word stemming but no stop word removal. The results for the DUC2002 dataset are shown in Table 6.7. Our F1 scores fall short of those of the lead baseline and of SoA extractive systems. In contrast to the word-extractive evaluation, subgraph selection (SSR) and redundancy removal (RSR) do not improve results over meaning ranking (MSR).

Table 6.8 shows the results for the CNN/Daily Mail [15]. Again, our system falls behind the baseline and is well below the system we compare with. It is worth pointing out how competitive the lead baseline is for this dataset, as even dedicated extractive systems like SummaRuNNer score lower.

The results in Tables 6.7 and 6.8 show unequivocally that, when applied to sentence extraction, our summarization strategy is well behind the lead-3 baseline and SoA extractive systems. In other words, selecting sentences in terms of the centrality of their disambiguated lexical meanings does not match the performance, measured in n-gram overlap, of other methods based on ranking directly with words or on neural networks underpinned by language models.

[15] The results of the baseline and our system correspond to a subset of 500 documents, while Liu and Lapata (2019) use the whole dataset.

		SBERT			SEW	
	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-1	ROUGE-2	ROUGE-L
MWR	41.72	-	-	39.77	-	-
VWR	38.80	-	-	36.97	-	-
SWR	41.40	-	-	39.77	-	-
RWR	41.48	-	-	39.95	-	-
MSR	32.85	**12.13**	29.62	31.53	10.84	28.53
VSR	**32.98**	11.80	**29.66**	31.03	9.93	27.82
SSR	29.10	9.17	26.14	27.38	7.9	24.69
RSR	29.30	9.30	26.36	27.69	7.76	24.76

Table 6.9: Comparison of SEW and SBERT versions of our system on the CNN/Daily Mail dataset.

In general, steps beyond ranking and disambiguation of meanings (MWR and MSR variations) do not translate into a clear positive effect in the ROUGE scores of the summaries. This is, to a certain extent, to be expected, since the purpose of ranking vertices is to spread relevance to vertices that are connected in the planning graph with highly ranked meanings. This redistribution of relevance and the steps following it – extraction, selectiona and ordering– seek to produce subgraphs that group clusters of highly relevant contents and enable the generation of semantically complete and overall coherent summaries. As discussed in Section 5.3.3, these concerns are not as relevant to extractive summarization as they are to abstractive summarization.

In the tables it can be observed that summarization systems tend to score higher in the DUC2002 dataset. This could be due to the fact that it is a curated dataset with high-quality summaries. Another potential reason is that some texts belong to domains where most relevant information is not necessarily found in their leading section. This could explain why AS systems score higher relative to lead baselines.

For the sake of completeness, we also presents a comparison on the CNN/Daily Mail dataset of our system using SBERT and SEW versions of

	DUC2002	CNN/Daily Mail	
	ROUGE-1 75 words	ROUGE-1 75 bytes	ROUGE-1 75 words
MWR	37.15	6.21	41.72
VWR	**42.51**	**26.75**	**59.94**
SWR	39.68	25.84	51.12
RWR	37.24	25.60	33.46

Table 6.10: ROUGE scores for word extractive summaries with position bias.

the $similarity$ function. As seen in Table 6.9, SBERT performs better across all summary versions.

What about document structure and position?

Our approach to text planning does not take into account the position of contents in the input text. All steps of our approach and their implementation for the experiments presented in this chapter operate without a notion of where in the source document is a mening being mentioned. This is in stark contrast to many works in AS we have reviewed or compared to so far, which encode position explictly as a feature in their systems, e.g., Wan (2010); Nallapati et al. (2017); Liu and Lapata (2019), or implictly as part of a neural architecture, e.g., Cheng and Lapata (2016).

On account of this consideration, we re-evaluate our system with a new $\boldsymbol{VR}$ ranking resulting from adjusting the bias term of Equation (5.2) with a decay function based on the position of the word aligned with a vertex:

$$\boldsymbol{VR}_u = d \cdot \boldsymbol{M}_u \cdot \frac{1}{e^{\frac{pos(u)}{\mu}}} + (1-d) \cdot \sum_{(u,v)\in E} \boldsymbol{Y}_{u,v} \cdot \boldsymbol{VR}_v \tag{6.1}$$

where $pos(u)$ is the offset of the word aligned with the vertex. We set the decay factor μ to the same value used for adj’.

The new results for word extraction and sentence extraction are shown

	DUC2002			CNN/Daily Mail		
	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-1	ROUGE-2	ROUGE-L
MSR	34.06	12.23	29.71	32.85	12.13	29.62
VSR	**36.54**	**14.59**	**32.08**	**39.38**	**17.03**	**35.63**
SSR	33.57	12.47	29.75	35.40	13.92	32.15
RSR	29.88	9.08	25.21	30.78	10.47	27.86

Table 6.11: ROUGE scores for sentence extractive summaries with position bias.

in Table 6.10 and Table 6.11, respectively. Our versions of word-based and sentence-based extraction based on vertex ranking (VWR and VSR) outperform the older versions in ROUGE-1 metrics by nearly 3 points in the case of DUC2002 and by 6 points in the case of CNN/Daily Mail. These improvements carry on to ROUGE-2 and ROUGE-L, and also to SSR. Gains in RSR are more moderate, particularly for the DUC2002 dataset. For CNN/Daily Mail, these new results bring us very close to the lead baseline and the SummaRuNNer system. More generally, they hint at the importance of considering document structure and position of contents in AS systems.

Chapter 7

CONCLUSIONS AND FUTURE WORK

In this thesis document, we have presented a semantically-oriented approach to planning summaries that operates on an intermediate representation of the input text we call a planning graph. A planning graph is a semantic representation that describes the disambiguated lexical meanings of a text and the relative prominence and generic relations between contents in the text. Due to its semantic nature, planning graphs support language and domain-independent methods. We have shown that this representation can be obtained with a combination of our own methods and readily available tools and resources with support for many languages.

Contrary to sequence to sequence methos, our approach separates text planning from NLU and NLG, and comprises several tasks that expose useful intermediate results. Planning is addressed by assessing contents first and then selecting and ordering them. The assessment part, central to the research presented in this thesis, applies ranking methods that contribute towards the creation of planning graphs by disambiguating lexical meanings and estimating their relevance, and the relevance of their mentions in the text.

Based on an experimental set up that includes an implementation of our appoach, we have conducted two empirical evaluations, one where we evaluate on tasks related to the ranking and disambiguation of candidate meanings, and another where we evaluate on single-document extractive summarization. In both cases we have used English datasets and compared to SoA systems.

This chapter closes the thesis with some conclusions drawn from our evaluations and a look at prospective lines of work. They are discussed in Sections 7.1 and 7.2, respectively.

7.1 Conclusions

One of our research goals was to contribute methods that can be used both for planning summaries and for other tasks. In that respect, we consider that we have acheived our goal by showing that the presented ranking methods are capable of ranking and disambiguating candidate lexical meanings above the proposed baselines, and produce results when applied to extractive summaries that are no too far from SoA systems –despite not being specifically designed for extractive summarization.

On the disambiguation front, our results do not only surpass demanding baselines like selecting the first sense returned by WordNet or BabelNet, but also manage to improve on state of the art systems, albeit slightly. It is particularly positive that this could be done using text similarity and classification methods based on glosses, since this guarantees that the methods will work for any lexical meaning for which a definition is available and, consequently, keeps our approach open and adaptable to new domains and topics. The contribution of BERT models cannot be underestimated. Not only they provide a large increase in performance when comparing glosses and local contexts, they are also important in adhering to our goal of keeping methods language-independent.

Extractive summarization has been far more challenging, with the results for the various versions of our system often falling below both the baselines and selected SoA systems. There are various reasons for this. One is the choice of surface-oriented metrics like ROUGE –still required if one is to compare with published works in AS– that only capture informativeness and, to a certain extent, fluency, on the basis of matching words and sequences of words between system and model summaires. This is arguably not the best evaluation method for a text planning strategy capable of attaining a semantic abstraction over the input texts (Ng and Abrecht, 2015).

In addition, both datasets used in the evaluation comprise only one reference summary per source document despite the fact that ROUGE was designed to work with multiple reference summaires. This lack of multiple reference summaries biases the evaluation against abstractive systems even when adapted to perform extractive summarization, as it the case of our system. Indeed, our planning strategy may pick words or sentences semantically related to those in the reference summary but not the same. In contrast, both the lead baseline and the systems we compare in our evaluation are designed purposedly for extractive summarization.

The flexibility and scope of our approach to text planning put it at disadvantage over specialized methods for summarization. In that respect, additional evaluations would be needed to demonstrate that the overall strategy also works for abstractive summarization, using additional domains and languages. Unfortunately, abstractive summarization requires NLG methods applicable to our text plans, an area of research that falls outside the scope of this thesis. These difficulties and limitations in the evaluation of text planning methods have been long known to practitioners in NLG (Dale and Mellish, 1998; Gatt and Krahmer, 2018).

Nevetheless, we believe that we have convincingly asserted that domain and language-independent planning of summaries can be effectively be carried out, an argument that is supported by our proposed intermediate representation, our discussion on means to instantiate it from text and the specific set up used in the experiments. While difficult to evaluate numerically, we also believe that our experiments with ranking and disambiguation of meanings show that keeping text planning separate from

NLU and NLG can expose useful information for downstream tasks other than summarization.

7.2 Future Work

Our evaluation is far more limited that we would have liked it to be. In particular, it would have been really interesting to evaluate using multilingual datasets for WSD and AS, e.g., the summarization data released in the Multiling workshops (Giannakopoulos, 2013; Giannakopoulos et al., 2015, 2017), since great effort has been placed in choosing tools and resources with support for many languages and to keep our own methods language-independent.

Another important facet of our system is its performance at detecting, ranking and disambiguating NEs, given that they are likely to play a more prominent role in selecting important contents that most word senses. The BabelNet database used in our experiments has a large coverage of NEs due to mapping Wikipedia/DBPedia entities, but our evaluation of EL with the SemEval 20015 dataset is fairly limited. In the future, we would like to evaluate our methods with the GERBIL benchmarking system (Röder et al., 2018), which automates evaluations on multiple datasets annotated with entities. GERBIL has support for 15 datasets that can be used to evaluate EL tools and which cover a variety of genres and domains.

Despite presenting methods for a coherent presentation of contents in the summary, in particular in the ordering step of our approach, we do not conduct any evaluation of the quality of the text plans in terms of coherence and cohesiveness. NLG and AS systems usually evaluate aspects related to the quality of produced texts, such as coherence, by presenting human judges with the texts to be evaluated and then asking them to fill questionnaires. Similarly, we did not conduct an in-depth evaluation of our redundancy removal component, for which an interesting evaluation would have involved highly redundant texts and a baseline based on comparing n-grams of the text fragments aligned with the extracted subgraphs.

Our experimental set up uses a UD parser to instantiate planning graphs. Given that we present a semantically-oriented text planner, it would be very interesting to test our system with deeper parsers, e.g., semantic parsers based on AMR or UCCA, and discourse parsers based on RST or other discourse representations. These tests could shed light on the importance of relations between contents and even open up the resulting graphs to other applications and types of summary.

We chose not to use a UD generator to produce abstracts from our text plans, amongst other reasons, because it is not trivial to guarantee that the subgraphs extracted by our approach are well-formed dependency trees and form a semantically sound set of lexical meanings. Recall from our discussion in Section 5.3.3 that there are a number of mechanisms that could be used to improve the quality of the extracted subgraphs, e.g., rules, language models, simplication methods. Experimenting with this mechanisms in conjunction with different types of parsers and generators (syntactic, semantic, discourse) constitutes another alluring line of research.

We have mentioned the importance of BERT in the performance of the early stages of the system used in our experimental evaluation. It follows then that fine-tuning BERT specifically for the tasks we apply it to is a likely way to improve results. Specifically, we would like to adapt it to predict if two meanings are related by comparing their glosses, and to predict the right meaning not just for WordNet senses but also for entities in Wikipedia, DBpedia or BabelNet. The former could be done by collecting pairs of meanings connected via semantic relations in a knwoledge base or dictionary, and use them as positive training examples. Extending GlossBERT to other lexical resources would involve building a training set combining WordNet with Wikipedia pairs of sentence and gloss –the latter perhaps could be collected from Wikipedia itself by looking at links between pages. Finally, a multilingual evaluation of our disambiguation strategy could involve fine tuning models from a multilingual version of

BERT or similar pre-trained model.

Bibliography

Abend, O. and Rappoport, A. (2013). Universal conceptual cognitive annotation (UCCA). In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*, pages 228–238. The Association for Computer Linguistics.

Abend, O. and Rappoport, A. (2017). The state of the art in semantic representation. In Barzilay, R. and Kan, M., 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 77–89. Association for Computational Linguistics.

Agirre, E. and Soroa, A. (2009). Personalizing pagerank for word sense disambiguation. In Lascarides, A., Gardent, C., and Nivre, J., editors, *EACL 2009, 12th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, Athens, Greece, March 30 - April 3, 2009*, pages 33–41. The Association for Computer Linguistics.

Aksoy, C., Bugdayci, A., Gur, T., Uysal, I., and Can, F. (2009). Semantic argument frequency-based multi-document summarization. In *The 24th International Symposium on Computer and Information Sciences, ISCIS 2009, 14-16 September 2009, North Cyprus*, pages 460–464. IEEE.

Al-Rfou, R., Kulkarni, V., Perozzi, B., and Skiena, S. (2015). POLYGLOT-NER: massive multilingual named entity recognition.

In Venkatasubramanian, S. and Ye, J., editors, *Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, BC, Canada, April 30 - May 2, 2015*, pages 586–594. SIAM.

Alfonseca, E., Pighin, D., and Garrido, G. (2013). HEADY: news headline abstraction through event pattern clustering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*, pages 1243–1253. The Association for Computer Linguistics.

Ammar, W., Mulcaire, G., Ballesteros, M., Dyer, C., and Smith, N. A. (2016). Many languages, one parser. *TACL*, 4:431–444.

Amplayo, R. K., Lim, S., and Hwang, S. (2018). Entity commonsense representation for neural abstractive summarization. In Walker, M. A., Ji, H., and Stent, A., 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 697–707. Association for Computational Linguistics.

Angeli, G., Premkumar, M. J. J., and Manning, C. D. (2015). Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 344–354. The Association for Computer Linguistics.

Arora, S., Liang, Y., and Ma, T. (2017). A simple but tough-to-beat baseline for sentence embeddings. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Augenstein, I., Padó, S., and Rudolph, S. (2012). Lodifier: Generating linked data from unstructured text. In Simperl, E., Cimiano, P., Polleres, A., Corcho, Ó., and Presutti, V., editors, *The Semantic Web: Research and Applications - 9th Extended Semantic Web Conference, ESWC 2012, Heraklion, Crete, Greece, May 27-31, 2012. Proceedings*, volume 7295 of *Lecture Notes in Computer Science*, pages 210–224. Springer.

Ballesteros, M., Bohnet, B., Mille, S., and Wanner, L. (2016). Data-driven deep-syntactic dependency parsing. *Natural Language Engineering*, 22(6):939–974.

Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., and Schneider, N. (2013). Abstract meaning representation for sembanking. In Dipper, S., Liakata, M., and Pareja-Lora, A., editors, *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse, LAW-ID@ACL 2013, August 8-9, 2013, Sofia, Bulgaria*, pages 178–186. The Association for Computer Linguistics.

Banerjee, S., Mitra, P., and Sugiyama, K. (2016). Multi-document abstractive summarization using ILP based multi-sentence compression. *CoRR*, abs/1609.07034.

Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M., and Etzioni, O. (2007). Open information extraction from the web. In Veloso, M. M., editor, *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 2670–2676.

Baralis, E., Cagliero, L., Mahoto, N. A., and Fiori, A. (2013). Graphsum: Discovering correlations among multiple terms for graph-based summarization. *Inf. Sci.*, 249:96–109.

Barrière, C. (2016). *Natural Language Understanding in a Semantic Web Context*. Springer.

Barzilay, R. and Lapata, M. (2005). Modeling local coherence: An entity-based approach. In Knight, K., Ng, H. T., and Oflazer, K., editors, *ACL*

2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA, pages 141–148. The Association for Computer Linguistics.

Barzilay, R. and Lee, L. (2003). Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In Hearst, M. A. and Ostendorf, M., editors, *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, HLT-NAACL 2003, Edmonton, Canada, May 27 - June 1, 2003*. The Association for Computational Linguistics.

Barzilay, R. and McKeown, K. (2005). Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

Barzilay, R., McKeown, K., and Elhadad, M. (1999). Information fusion in the context of multi-document summarization. In Dale, R. and Church, K. W., editors, *27th Annual Meeting of the Association for Computational Linguistics, University of Maryland, College Park, Maryland, USA, 20-26 June 1999*. ACL.

Bengtson, E. and Roth, D. (2008). Understanding the value of features for coreference resolution. In *2008 Conference on Empirical Methods in Natural Language Processing, EMNLP 2008, Proceedings of the Conference, 25-27 October 2008, Honolulu, Hawaii, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 294–303. ACL.

Bing, L., Li, P., Liao, Y., Lam, W., Guo, W., and Passonneau, R. J. (2015). Abstractive multi-document summarization via phrase selection and merging. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1587–1597. The Association for Computer Linguistics.

Bovi, C. D. and Raganato, A. (2017). Sew-embed at semeval-2017 task 2: Language-independent concept representations from a semantically enriched wikipedia. In Bethard, S., Carpuat, M., Apidianaki, M., Mohammad, S. M., Cer, D. M., and Jurgens, D., editors, *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*, pages 261–266. Association for Computational Linguistics.

Candito, M. and Constant, M. (2014). Strategies for contiguous multiword expression analysis and 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 743–753. The Association for Computer Linguistics.

Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Jr., E. R. H., and Mitchell, T. M. (2010). Toward an architecture for never-ending language learning. In Fox, M. and Poole, D., editors, *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*. AAAI Press.

Carlson, L., Marcu, D., and Okurovsky, M. E. (2001). Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Proceedings of the SIGDIAL 2001 Workshop, The 2nd Annual Meeting of the Special Interest Group on Discourse and Dialogue, Saturday, September 1, 2001 to Sunday, September 2, 2001, Aalborg, Denmark*. The Association for Computer Linguistics.

Çelikyilmaz, A., Bosselut, A., He, X., and Choi, Y. (2018). Deep communicating agents for abstractive summarization. In Walker, M. A., Ji, H., and Stent, A., 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 1662–1675. Association for Computational Linguistics.

Chali, Y. and Joty, S. R. (2008). Improving the performance of the random walk model for answering complex questions. In *ACL 2008, Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, June 15-20, 2008, Columbus, Ohio, USA, Short Papers*, pages 9–12. The Association for Computer Linguistics.

Chan, S. W. K. (2006). Beyond keyword and cue-phrase matching: A sentence-based abstraction technique for information extraction. *Decision Support Systems*, 42(2):759–777.

Che, W., Liu, Y., Wang, Y., Zheng, B., and Liu, T. (2018). Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In Zeman, D. and Hajic, J., editors, *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, Brussels, Belgium, October 31 - November 1, 2018*, pages 55–64. Association for Computational Linguistics.

Chen, D., Schneider, N., Das, D., and Smith, N. A. (2010). SEMAFOR: frame argument resolution with log-linear models. In Erk, K. and Strapparava, C., editors, *Proceedings of the 5th International Workshop on Semantic Evaluation, SemEval@ACL 2010, Uppsala University, Uppsala, Sweden, July 15-16, 2010*, pages 264–267. The Association for Computer Linguistics.

Chen, Q., Zhu, X., Ling, Z., Wei, S., and Jiang, H. (2016). Distraction-based neural networks for document summarization. *CoRR*, abs/1610.08462.

Chen, X., Liu, Z., and Sun, M. (2014). A unified model for word sense representation and disambiguation. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1025–1035. ACL.

Cheng, J. and Lapata, M. (2016). Neural summarization by extracting sentences and words. 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.

Cheung, J. C. K. and Penn, G. (2014). Unsupervised sentence enhancement for automatic summarization. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 775–786. ACL.

Chiang, D. (2000). Statistical parsing with an automatically-extracted tree adjoining grammar. In *38th Annual Meeting of the Association for Computational Linguistics, Hong Kong, China, October 1-8, 2000.* ACL.

Christensen, J., Mausam, Soderland, S., and Etzioni, O. (2013). Towards coherent multi-document summarization. In Vanderwende, L., III, H. D., and Kirchhoff, K., editors, *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 1163–1173. The Association for Computational Linguistics.

Christensen, J., Soderland, S., Bansal, G., and Mausam (2014). Hierarchical summarization: Scaling up multi-document summarization. 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 902–912. The Association for Computer Linguistics.

Cimiano, P., Buitelaar, P., McCrae, J. P., and Sintek, M. (2011). Lexinfo: A declarative model for the lexicon-ontology interface. *J. Web Sem.*, 9(1):29–51.

Clark, K. and Manning, C. D. (2015). Entity-centric coreference resolution with model stacking. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1405–1415. The Association for Computer Linguistics.

Clark, S. and Curran, J. R. (2007). Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4):493–552.

Claro, D. B., Souza, M., Xavier, C. C., and de Oliveira, L. S. (2019). Multilingual open information extraction: Challenges and opportunities. *Information*, 10(7):228.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Corcoglioniti, F., Rospocher, M., and Aprosio, A. P. (2016a). Frame-based ontology population with PIKES. *IEEE Trans. Knowl. Data Eng.*, 28(12):3261–3275.

Corcoglioniti, F., Rospocher, M., Aprosio, A. P., and Tonelli, S. (2016b). Premon: a lemon extension for exposing predicate models as linked data. In Calzolari, N., Choukri, K., Declerck, T., Goggi, S., Grobelnik, M., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016.* European Language Resources Association (ELRA).

Da Cunha, I., Wanner, L., and Cabré, T. (2007). Summarization of specialized discourse: The case of medical articles in spanish. *Terminology. International Journal of Theoretical and Applied Issues in Specialized Communication*, 13(2):249–286.

Dale, R. and Mellish, C. (1998). Towards evaluation in natural language generation. In *In Proceedings of First International Conference on Language Resources and Evaluation*.

de Lacalle, M. L., Laparra, E., Aldabe, I., and Rigau, G. (2016). A multilingual predicate matrix. In Calzolari, N., Choukri, K., Declerck, T., Goggi, S., Grobelnik, M., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016*. European Language Resources Association (ELRA).

de Marneffe, M., Dozat, T., Silveira, N., Haverinen, K., Ginter, F., Nivre, J., and Manning, C. D. (2014). Universal stanford dependencies: A cross-linguistic typology. In Calzolari, N., Choukri, K., Declerck, T., Loftsson, H., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014, Reykjavik, Iceland, May 26-31, 2014*, pages 4585–4592. European Language Resources Association (ELRA).

de Rosis, F. and Grasso, F. (1999). Affective natural language generation. In Paiva, A., editor, *Affective Interactions, Towards a New Generation of Computer Interfaces.*, volume 1814 of *Lecture Notes in Computer Science*, pages 204–218. Springer.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In Burstein, J., Doran, C., and Solorio, T., 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.

Dohare, S. and Karnick, H. (2017). Text summarization using abstract meaning representation. *CoRR*, abs/1706.01678.

Droganova, K. and Zeman, D. (2019). Towards deep universal dependencies. In *Proceedings of the Fifth International Conference on Dependency Linguistics (Depling, SyntaxFest 2019)*, pages 144–152, Paris, France. Association for Computational Linguistics.

Dunietz, J. and Gillick, D. (2014). A new entity salience task with millions of training examples. In Bouma, G. and Parmentier, Y., editors, *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2014, April 26-30, 2014, Gothenburg, Sweden*, pages 205–209. The Association for Computer Linguistics.

Duong, L., Afshar, H., Estival, D., Pink, G., Cohen, P., and Johnson, M. (2017). Multilingual semantic parsing and code-switching. In Levy, R. and Specia, L., editors, *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), Vancouver, Canada, August 3-4, 2017*, pages 379–389. Association for Computational Linguistics.

Durrett, G., Berg-Kirkpatrick, T., and Klein, D. (2016). Learning-based single-document summarization with compression and anaphoricity constraints. 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.

Durrett, G. and Klein, D. (2014). A joint model for entity analysis: Coreference, typing, and linking. *TACL*, 2:477–490.

Dutta, A., Meilicke, C., Niepert, M., and Ponzetto, S. P. (2013). Integrating open and closed information extraction: Challenges and first steps. In Hellmann, S., Filipowska, A., Barrière, C., Mendes, P. N., and Kontokostas, D., editors, *Proceedings of the NLP & DBpedia workshop co-located with the 12th International Semantic Web Conference (ISWC*

2013), Sydney, Australia, October 22, 2013., volume 1064 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Elsner, M. and Santhanam, D. (2011). Learning to fuse disparate sentences. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, MTTG '11, pages 54–63, Stroudsburg, PA, USA. Association for Computational Linguistics.

Erk, K. and Pado, S. (2006). Shalmaneser - a flexible toolbox for semantic role assignment. In *Proceedings of LREC 2006*, Genoa, Italy.

Erkan, G. and Radev, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research (JAIR)*, 22:457–479.

Etzioni, O., Cafarella, M. J., Downey, D., Popescu, A., Shaked, T., Soderland, S., Weld, D. S., and Yates, A. (2005). Unsupervised named-entity extraction from the web: An experimental study. *Artif. Intell.*, 165(1):91–134.

Falke, T. and Gurevych, I. (2017). Bringing structure into summaries: Crowdsourcing a benchmark corpus of concept maps. In Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2951–2961. Association for Computational Linguistics.

Fang, Y. and Teufel, S. (2016). Improving argument overlap for proposition-based summarisation. 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.

Färber, M., Bartscherer, F., Menne, C., and Rettinger, A. (2018). Linked data quality of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web*, 9(1):77–129.

Färber, M., Ell, B., Menne, C., and Rettinger, A. (2015). A comparative survey of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web Journal, July*, pages 1–26.

Ferreira, R., de Souza Cabral, L., de Freitas, F. L. G., Lins, R. D., de França Pereira e Silva, G., Simske, S. J., and Favaro, L. (2014). A multi-document summarization system based on statistics and linguistic treatment. *Expert Syst. Appl.*, 41(13):5780–5787.

Ferreira, T. C., Calixto, I., Wubben, S., and Krahmer, E. (2017). Linguistic realisation as machine translation: Comparing different MT models for amr-to-text generation. In Alonso, J. M., Bugarín, A., and Reiter, E., editors, *Proceedings of the 10th International Conference on Natural Language Generation, INLG 2017, Santiago de Compostela, Spain, September 4-7, 2017*, pages 1–10. Association for Computational Linguistics.

Filippova, K. (2010). Multi-sentence compression: Finding shortest paths in word graphs. In Huang, C. and Jurafsky, D., editors, *COLING 2010, 23rd International Conference on Computational Linguistics, Proceedings of the Conference, 23-27 August 2010, Beijing, China*, pages 322–330. Tsinghua University Press.

Filippova, K. and Altun, Y. (2013). Overcoming the lack of parallel data in sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1481–1491. ACL.

Filippova, K. and Strube, M. (2008). Sentence fusion via dependency graph compression. In *2008 Conference on Empirical Methods in Natural Language Processing, EMNLP 2008, Proceedings of the Conference, 25-27 October 2008, Honolulu, Hawaii, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 177–185. ACL.

Fillmore, C. J. (1968). The case for case. In Bach, E. and Harms, R. T., editors, *Universals in Linguistic Theory*, pages 0–88. Holt, Rinehart and Winston, New York.

Fillmore, C. J. (1976). The case for Case reopened. In Cole, P. and Sadock, J. M., editors, *Grammatical Relations*, volume 8 of *Syntax and Semantics*. Academic Press, New York.

Flanigan, J., Dyer, C., Smith, N. A., and Carbonell, J. G. (2016). Generation from abstract meaning representation using tree transducers. In Knight, K., Nenkova, A., and Rambow, O., 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 731–739. The Association for Computational Linguistics.

Gabriel, S., Bosselut, A., Holtzman, A., Lo, K., Çelikyilmaz, A., and Choi, Y. (2019). Cooperative generator-discriminator networks for abstractive summarization with narrative flow. *CoRR*, abs/1907.01272.

Gambhir, M. and Gupta, V. (2017). Recent automatic text summarization techniques: a survey. *Artif. Intell. Rev.*, 47(1):1–66.

Ganesan, K. (2018). ROUGE 2.0: Updated and improved measures for evaluation of summarization tasks. *CoRR*, abs/1803.01937.

Ganesan, K., Zhai, C., and Han, J. (2010). Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. In Huang, C. and Jurafsky, D., editors, *COLING 2010, 23rd International Conference on Computational Linguistics, Proceedings of the Conference, 23-27 August 2010, Beijing, China*, pages 340–348. Tsinghua University Press.

Gangemi, A., Alam, M., Asprino, L., Presutti, V., and Recupero, D. R. (2016). Framester: A wide coverage linguistic linked data hub. In Blomqvist, E., Ciancarini, P., Poggi, F., and Vitali, F., editors, *Knowledge Engineering and Knowledge Management - 20th International*

Conference, EKAW 2016, Bologna, Italy, November 19-23, 2016, Proceedings, volume 10024 of *Lecture Notes in Computer Science*, pages 239–254.

Gangemi, A., Presutti, V., Recupero, D. R., Nuzzolese, A. G., Draicchio, F., and Mongiovì, M. (2017). Semantic web machine reading with FRED. *Semantic Web*, 8(6):873–893.

Gatt, A. and Krahmer, E. (2018). Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *J. Artif. Intell. Res.*, 61:65–170.

Gatt, A. and Reiter, E. (2009). Simplenlg: A realisation engine for practical applications. In Krahmer, E. and Theune, M., editors, *ENLG 2009 - Proceedings of the 12th European Workshop on Natural Language Generation, March 30-31, 2009, Athens, Greece*, pages 90–93. The Association for Computer Linguistics.

Gerani, S., Carenini, G., and Ng, R. T. (2016). Modeling content and structure for abstractive review summarization. *Computer Speech & Language*.

Gerani, S., Mehdad, Y., Carenini, G., Ng, R. T., and Nejat, B. (2014). Abstractive summarization of product reviews using discourse structure. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1602–1613.

Giannakopoulos, G. (2013). Multi-document multilingual summarization and evaluation tracks in ACL 2013 MultiLing workshop. In *Proceedings of the MultiLing 2013 Workshop on Multilingual Multi-document Summarization*, pages 20–28, Sofia, Bulgaria. Association for Computational Linguistics.

Giannakopoulos, G., Conroy, J. M., Kubina, J., Rankel, P. A., Lloret, E., Steinberger, J., Litvak, M., and Favre, B. (2017). Multiling 2017 overview. In Giannakopoulos, G., Lloret, E., Conroy, J. M., Steinberger, J., Litvak, M., Rankel, P. A., and Favre, B., editors, *Proceedings of the Workshop on Summarization and Summary Evaluation Across Source Types and Genres, MultiLing@EACL 2017, Valencia, Spain, April 3, 2017*, pages 1–6. Association for Computational Linguistics.

Giannakopoulos, G., Kubina, J., Conroy, J. M., Steinberger, J., Favre, B., Kabadjov, M. A., Kruschwitz, U., and Poesio, M. (2015). Multiling 2015: Multilingual summarization of single and multi-documents, on-line fora, and call-center conversations. In *Proceedings of the SIGDIAL 2015 Conference, The 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 2-4 September 2015, Prague, Czech Republic*, pages 270–274. The Association for Computer Linguistics.

Grave, E., Bojanowski, P., Gupta, P., Joulin, A., and Mikolov, T. (2018). Learning word vectors for 157 languages. In Calzolari, N., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., and Tokunaga, T., editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. European Language Resources Association (ELRA).

Guinaudeau, C. and Strube, M. (2013). Graph-based local coherence modeling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*, pages 93–103. The Association for Computer Linguistics.

Hajic, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M. A., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Stepánek, J., Stranák, P., Surdeanu, M., Xue, N., and Zhang, Y. (2009). The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In Hajic, J., editor, *Proceedings of the Thirteenth Conference on Com-*

putational Natural Language Learning: Shared Task, CoNLL 2009, Boulder, Colorado, USA, June 4, 2009, pages 1–18. ACL.

Han, X., Sun, L., and Zhao, J. (2011). Collective entity linking in web text: a graph-based method. In Ma, W., Nie, J., Baeza-Yates, R. A., Chua, T., and Croft, W. B., editors, *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, July 25-29, 2011*, pages 765–774. ACM.

Han, X., Zhu, H., Yu, P., Wang, Z., Yao, Y., Liu, Z., and Sun, M. (2018). Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4803–4809. Association for Computational Linguistics.

Hardy and Vlachos, A. (2018). Guided neural language generation for abstractive summarization using abstract meaning representation. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 768–773. Association for Computational Linguistics.

He, S., Li, Z., and Zhao, H. (2019). Syntax-aware multilingual semantic role labeling. In Inui, K., Jiang, J., Ng, V., and Wan, X., 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 5349–5358. Association for Computational Linguistics.

Hellmann, S., Filipowska, A., Barrière, C., Mendes, P. N., and Kontokostas, D. (2013a). NLP & dbpedia an upward knowledge acquisition spiral. In Hellmann, S., Filipowska, A., Barrière, C., Mendes,

P. N., and Kontokostas, D., editors, *Proceedings of the NLP & DBpedia workshop co-located with the 12th International Semantic Web Conference (ISWC 2013), Sydney, Australia, October 22, 2013.*, volume 1064 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Hellmann, S., Lehmann, J., Auer, S., and Brümmer, M. (2013b). Integrating NLP using linked data. In Alani, H., Kagal, L., Fokoue, A., Groth, P. T., Biemann, C., Parreira, J. X., Aroyo, L., Noy, N. F., Welty, C., and Janowicz, K., editors, *The Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part II*, volume 8219 of *Lecture Notes in Computer Science*, pages 98–113. Springer.

Hendrickx, I., Kim, S. N., Kozareva, Z., Nakov, P., Séaghdha, D. Ó., Padó, S., Pennacchiotti, M., Romano, L., and Szpakowicz, S. (2010). Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In Erk, K. and Strapparava, C., editors, *Proceedings of the 5th International Workshop on Semantic Evaluation, SemEval@ACL 2010, Uppsala University, Uppsala, Sweden, July 15-16, 2010*, pages 33–38. The Association for Computer Linguistics.

Hermann, K. M., Kociský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching machines to read and comprehend. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1693–1701.

Hernault, H., Prendinger, H., duVerle, D. A., and Ishizuka, M. (2010). HILDA: A discourse parser using support vector machine classification. *D&D*, 1(3):1–33.

Hershcovich, D., Aizenbud, Z., Choshen, L., Sulem, E., Rappoport, A., and Abend, O. (2019). Semeval-2019 task 1: Cross-lingual semantic parsing with UCCA. In May, J., Shutova, E., Herbelot, A., Zhu, X., Apidianaki, M., and Mohammad, S. M., editors, *Proceedings of the 13th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2019, Minneapolis, MN, USA, June 6-7, 2019*, pages 1–10. Association for Computational Linguistics.

Hirao, T., Nishino, M., Yoshida, Y., Suzuki, J., Yasuda, N., and Nagata, M. (2015). Summarizing a document by trimming the discourse tree. *IEEE/ACM Trans. Audio, Speech & Language Processing*, 23(11):2081–2092.

Hirao, T., Yoshida, Y., Nishino, M., Yasuda, N., and Nagata, M. (2013). Single-document summarization as a tree knapsack problem. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1515–1520. ACL.

Hoffart, J., Yosef, M. A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., and Weikum, G. (2011). Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 782–792. ACL.

Honnibal, M. and Montani, I. (2017). spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Hovy, E. and Marcu, D. (1998). Tutorial on automated text summarization. In *COLING/ACL*.

Huang, L., Sun, C., Qiu, X., and Huang, X. (2019). Glossbert: BERT for word sense disambiguation with gloss knowledge. In Inui, K., Jiang, J., Ng, V., and Wan, X., 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 3507–3512. Association for Computational Linguistics.

Jackendoff, R. (1972). *Semantic Interpretation in Generative Grammar*. MIT Press, Cambridge, MA.

Ji, Y. and Eisenstein, J. (2014). Representation learning for text-level discourse 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 13–24. The Association for Computer Linguistics.

Jing, H. and McKeown, K. (2000). Cut and paste based text summarization. In *6th Applied Natural Language Processing Conference, ANLP 2000, Seattle, Washington, USA, April 29 - May 4, 2000*, pages 178–185. ACL.

Johansson, R. and Nugues, P. (2007). LTH: semantic structure extraction using nonprojective dependency trees. In Agirre, E., i Villodre, L. M., and Wicentowski, R., editors, *Proceedings of the 4th International Workshop on Semantic Evaluations, SemEval@ACL 2007, Prague, Czech Republic, June 23-24, 2007*, pages 227–230. The Association for Computer Linguistics.

Johansson, R. and Nugues, P. (2008). Dependency-based semantic role labeling of propbank. In *2008 Conference on Empirical Methods in Natural Language Processing, EMNLP 2008, Proceedings of the Conference, 25-27 October 2008, Honolulu, Hawaii, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 69–78. ACL.

Jones, K. S. and Endres-Niggemeyer, B. (1995). Automatic summarizing. *Inf. Process. Manag.*, 31(5):625–630.

Joshi, A. K. and Schabes, Y. (1997). *Tree-Adjoining Grammars*, pages 69–123. Springer Berlin Heidelberg, Berlin, Heidelberg.

Joty, S. R., Carenini, G., and Ng, R. T. (2015). CODRA: A novel discriminative framework for rhetorical analysis. *Computational Linguistics*, 41(3):385–435.

Kågebäck, M., Mogren, O., Tahmasebi, N., and Dubhashi, D. (2014). Extractive summarization using continuous vector space models. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*, pages 31–39, Gothenburg, Sweden. Association for Computational Linguistics.

Kamp, H. and Reyle, U. (1993). *From Discourse to Logic - Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*, volume 42 of *Studies in linguistics and philosophy*. Springer.

Khan, A., Salim, N., Farman, H., Khan, M., Jan, B., Ahmad, A., Ahmed, I., and Paul, A. (2018). Abstractive text summarization based on improved semantic graph approach. *International Journal of Parallel Programming*, 46(5):992–1016.

Khan, A., Salim, N., and Kumar, Y. J. (2015). A framework for multi-document abstractive summarization based on semantic role labelling. *Appl. Soft Comput.*, 30:737–747.

Kleinberg, J. M. (1999). Hubs, authorities, and communities. *ACM Comput. Surv.*, 31(4es):5.

Knight, K., Baranescu, L., Bonial, C., Georgescu, M., Griffitt, K., Hermjakob, U., Marcu, D., Palmer, M., and Schneifer, N. (2014). Abstract meaning representation (amr) annotation release 1.0. *Web download.*

Koller, A., Oepen, S., and Sun, W. (2019). Graph-based meaning representations: Design and processing. In Nakov, P. and Palmer, A., editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics: Tutorial Abstracts, ACL 2019, Florence, Italy, July 28, 2019, Volume 4: Tutorial Abstracts*, pages 6–11. Association for Computational Linguistics.

Konstantinova, N. (2014). Review of relation extraction methods: What is new out there? In Ignatov, D. I., Khachay, M. Y., Panchenko, A., Konstantinova, N., and Yavorskiy, R., editors, *Analysis of Images, Social Networks and Texts - Third International Conference, AIST 2014, Yekaterinburg, Russia, April 10-12, 2014, Revised Selected Papers*, volume 436 of *Communications in Computer and Information Science*, pages 15–28. Springer.

Konstas, I., Iyer, S., Yatskar, M., Choi, Y., and Zettlemoyer, L. (2017). Neural AMR: sequence-to-sequence models for parsing and generation. In Barzilay, R. and Kan, M., 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 146–157. Association for Computational Linguistics.

Kübler, S. and Zhekova, D. (2016). Multilingual coreference resolution. *Language and Linguistics Compass*, 10(11):614–631.

Kuhlmann, M. and Oepen, S. (2016). Towards a catalogue of linguistic graph banks. *Computational Linguistics*, 42(4):819–827.

Kusner, M. J., Sun, Y., Kolkin, N. I., and Weinberger, K. Q. (2015). From word embeddings to document distances. In Bach, F. R. and Blei, D. M., editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 957–966. JMLR.org.

Langkilde, I. and Knight, K. (1998). Generation that exploits corpus-based statistical knowledge. In Boitet, C. and Whitelock, P., editors, *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, COLING-ACL '98, August 10-14, 1998, Université de Montréal, Montréal, Quebec, Canada. Proceedings of the Conference*, pages 704–710. Morgan Kaufmann Publishers / ACL.

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., and Bizer, C. (2015). Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Letsios, M., Balalau, O. D., Danisch, M., Orsini, E., and Sozio, M. (2016). Finding heaviest k-subgraphs and events in social media. In Domeniconi, C., Gullo, F., Bonchi, F., Domingo-Ferrer, J., Baeza-Yates, R. A., Zhou, Z., and Wu, X., editors, *IEEE International Conference on Data Mining Workshops, ICDM Workshops 2016, December 12-15, 2016, Barcelona, Spain.*, pages 113–120. IEEE.

Li, J., Li, R., and Hovy, E. H. (2014). Recursive deep models for discourse parsing. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 2061–2069. ACL.

Li, J., Sun, A., Han, J., and Li, C. (2018). A survey on deep learning for named entity recognition. *CoRR*, abs/1812.09449.

Li, P., Cai, T. W., and Huang, H. (2015). Weakly supervised natural language processing framework for abstractive multi-document summarization: Weakly supervised abstractive multi-document summarization. In Bailey, J., Moffat, A., Aggarwal, C. C., de Rijke, M., Kumar, R., Murdock, V., Sellis, T. K., and Yu, J. X., editors, *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 1401–1410. ACM.

Li, Y. and Li, S. (2014). Query-focused multi-document summarization: Combining a topic model with graph-based semi-supervised learning. In Hajic, J. and Tsujii, J., editors, *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Confer-*

ence: Technical Papers, August 23-29, 2014, Dublin, Ireland, pages 1197–1207. ACL.

Liang, P. (2013). Lambda dependency-based compositional semantics. *CoRR*, abs/1309.4408.

Lin, C. (2004a). Looking for a few good metrics: Automatic summarization evaluation - how many samples are enough? In Kando, N. and Ishikawa, H., editors, *Proceedings of the Fourth NTCIR Workshop on Research in Information Access Technologies Information Retrieval, Question Answering and Summarization, NTCIR-4, National Center of Sciences, Tokyo, Japan, June 2-4, 2004*. National Institute of Informatics (NII).

Lin, C.-Y. (2004b). Rouge: A package for automatic evaluation of summaries. In Marie-Francine Moens, S. S., editor, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Lin, D. (1997). Using syntactic dependency as local context to resolve word sense ambiguity. In Cohen, P. R. and Wahlster, W., editors, *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, 7-12 July 1997, Universidad Nacional de Educación a Distancia (UNED), Madrid, Spain.*, pages 64–71. Morgan Kaufmann Publishers / ACL.

Liu, F., Flanigan, J., Thomson, S., Sadeh, N. M., and Smith, N. A. (2015). Toward abstractive summarization using semantic representations. In Mihalcea, R., Chai, J. Y., and Sarkar, A., editors, *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 1077–1086. The Association for Computational Linguistics.

Liu, P. J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, L., and Shazeer, N. (2018). Generating wikipedia by summarizing long sequences. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Liu, T. (2011). *Learning to Rank for Information Retrieval*. Springer.

Liu, Y. and Lapata, M. (2019). Text summarization with pretrained encoders. In Inui, K., Jiang, J., Ng, V., and Wan, X., 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 3728–3738. Association for Computational Linguistics.

Lopez, P. (2000). Extended partial parsing for lexicalized tree grammars. In *In Proc. of the Sixth International Workshop on Parsing Technologies (IWPT 2000*, pages 159–170.

Louis, A., Joshi, A. K., and Nenkova, A. (2010). Discourse indicators for content selection in summarization. In Fernández, R., Katagiri, Y., Komatani, K., Lemon, O., and Nakano, M., editors, *Proceedings of the SIGDIAL 2010 Conference, The 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 24-15 September 2010, Tokyo, Japan*, pages 147–156. The Association for Computer Linguistics.

Lu, J. and Ng, V. (2018). Event coreference resolution: A survey of two decades of research. In Lang, J., editor, *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 5479–5486. ijcai.org.

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165.

Mahdisoltani, F., Biega, J., and Suchanek, F. M. (2015). YAGO3: A knowledge base from multilingual wikipedias. In *CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings*. www.cidrdb.org.

Mani, I. (2001). *Automatic summarization*, volume 3. John Benjamins Publishing.

Mani, I. and Maybury, M. T. (1999). *Advances in Automatic Text Summarization*. MIT Press, Cambridge, MA, USA.

Mann, W. C. and Thompson, S. A. (1988). Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281.

Marcu, D. (1998). To build text summaries of high quality, nuclearity is not sufficient. In *Working Notes of the AAAI-98 Spring Symposium on Intelligent Text Summarization*, pages 1–8.

Marcu, D. (2000). *The Theory and Practice of Discourse Parsing and Summarization*. Bradford Books. The MIT Press.

Markov, A. (1971). Extension of the Limit Theorems of Probability Theory to a Sum of Variables Connected in a Chain. In Howard, R., editor, *Dynamic Probabilistic Systems (Volume I: Markov Models)*, chapter Appendix B, pages 552–577. John Wiley & Sons, Inc., New York City.

Martí, M., Taulé, M., Márquez, L., and Bertran, M. (2007). Ancora: A multilingual and multilevel annotated corpus. *206865-Corpus Linguistics*.

May, J. (2016). Semeval-2016 task 8: Meaning representation parsing. In Bethard, S., Cer, D. M., Carpuat, M., Jurgens, D., Nakov, P., and Zesch, T., editors, *Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2016, San Diego, CA, USA, June 16-17, 2016*, pages 1063–1073. The Association for Computer Linguistics.

May, J. and Priyadarshi, J. (2017). Semeval-2017 task 9: Abstract meaning representation parsing and generation. In Bethard, S., Carpuat, M., Apidianaki, M., Mohammad, S. M., Cer, D. M., and Jurgens, D., editors, *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*, pages 536–545. Association for Computational Linguistics.

Mehdad, Y., Carenini, G., and Ng, R. T. (2014). Abstractive summarization of spoken and written conversations based on phrasal queries. 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 1220–1230. The Association for Computer Linguistics.

Mel'čuk, I. A. (1988). *Dependency syntax: theory and practice*. SUNY series in Linguistics. SUNY press.

Mendes, P. N., Jakob, M., García-Silva, A., and Bizer, C. (2011). Dbpedia spotlight: shedding light on the web of documents. In Ghidini, C., Ngomo, A. N., Lindstaedt, S. N., and Pellegrini, T., editors, *Proceedings the 7th International Conference on Semantic Systems, I-SEMANTICS 2011, Graz, Austria, September 7-9, 2011*, ACM International Conference Proceeding Series, pages 1–8. ACM.

Meyers, A., Reeves, R., Macleod, C., Szekely, R., Zielinska, V., Young, B., and Grishman, R. (2004). Annotating noun argument structure for nombank. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation, LREC 2004, May 26-28, 2004, Lisbon, Portugal*. European Language Resources Association.

Mihalcea, R. and Tarau, P. (2004). Textrank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing , EMNLP 2004, A meeting of SIGDAT, a Special Interest Group of the ACL, held in conjunction with ACL 2004, 25-26 July 2004, Barcelona, Spain*, pages 404–411. ACL.

Mille, S., Ballesteros, M., Burga, A., Casamayor, G., and Wanner, L. (2016). Multilingual natural language generation within abstractive summarization. In Vrochidis, S., Melero, M., Wanner, L., Grivolla, J., and Estève, Y., editors, *Proceedings of the 1st International Workshop on Multimodal Media Data Analytics co-located with the 22nd European Conference on Artificial Intelligence, MMDA@ECAI 2016, The Hague, Netherlands, August 30, 2016.*, volume 1801 of *CEUR Workshop Proceedings*, pages 33–38. CEUR-WS.org.

Mille, S. and Wanner, L. (2008). Multilingual summarization in practice: the case of patent claims. In *Proceedings of the 12th European association of machine translation conference*, pages 120–129.

Miller, G. A. (1995). Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Mitkov, R., Palmer, M., Pradhan, S., and Xue, N. (2016). Semantic role labelling.

Molina, M. P. (1995). Documentary abstracting: Toward a methodological model. *JASIS*, 46(3):225–234.

Moore, J. D. and Pollack, M. E. (1992). A problem for RST: The need for multi-level discourse analysis. *Computational Linguistics*, 18(4):537–544.

Morey, M., Muller, P., and Asher, N. (2017). How much progress have we made on RST discourse parsing? A replication study of recent results on the RST-DT. In Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 1319–1324. Association for Computational Linguistics.

Moro, A. and Navigli, R. (2015). Semeval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. In Cer, D. M., Jurgens, D., Nakov, P., and Zesch, T., editors, *Proceedings of the 9th*

International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015, pages 288–297. The Association for Computer Linguistics.

Moro, A., Raganato, A., and Navigli, R. (2014). Entity linking meets word sense disambiguation: a unified approach. *TACL*, 2:231–244.

Moser, M. and Moore, J. D. (1996). Toward a synthesis of two accounts of discourse structure. *Computational Linguistics*, 22(3):409–419.

Mulcaire, P., Swayamdipta, S., and Smith, N. A. (2018). Polyglot semantic role labeling. In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 667–672. Association for Computational Linguistics.

Muller, P., Braud, C., and Morey, M. (2019). ToNy: Contextual embeddings for accurate multilingual discourse segmentation of full documents. In *Proceedings of the Workshop on Discourse Relation Parsing and Treebanking 2019*, pages 115–124, Minneapolis, MN. Association for Computational Linguistics.

Nakashole, N., Weikum, G., and Suchanek, F. M. (2012). PATTY: A taxonomy of relational patterns with semantic types. In Tsujii, J., Henderson, J., and Pasca, M., editors, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*, pages 1135–1145. ACL.

Nallapati, R., Zhai, F., and Zhou, B. (2017). Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In Singh, S. P. and Markovitch, S., editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3075–3081. AAAI Press.

Nallapati, R., Zhou, B., dos Santos, C. N., Gülçehre, Ç., and Xiang, B. (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond. In Goldberg, Y. and Riezler, S., editors, *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 280–290. ACL.

Napoles, C., Gormley, M. R., and Durme, B. V. (2012). Annotated gigaword. In Fan, J., Hoffman, R., Kalyanpur, A., Riedel, S., Suchanek, F. M., and Talukdar, P. P., editors, *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction, AKBC-WEKEX@NAACL-HLT 2012, Montrèal, Canada, June 7-8, 2012*, pages 95–100. Association for Computational Linguistics.

Narayan, S., Cohen, S. B., and Lapata, M. (2018). Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1797–1807. Association for Computational Linguistics.

Navigli, R. (2009). Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41(2):10:1–10:69.

Navigli, R., Camacho-Collados, J., and Raganato, A. (2017). Word sense disambiguation: A unified evaluation framework and empirical comparison. In Lapata, M., Blunsom, P., and Koller, A., editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, pages 99–110. Association for Computational Linguistics.

Navigli, R. and Lapata, M. (2010). An experimental study of graph connectivity for unsupervised word sense disambiguation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(4):678–692.

Navigli, R. and Ponzetto, S. P. (2012). Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artif. Intell.*, 193:217–250.

Nenkova, A. and McKeown, K. (2011). Automatic summarization. *Foundations and Trends in Information Retrieval*, 5(2-3):103–233.

Neuhaus, M. and Bunke, H. (2007). *Bridging the Gap between Graph Edit Distance and Kernel Machines*, volume 68 of *Series in Machine Perception and Artificial Intelligence*. WorldScientific.

Ng, J. and Abrecht, V. (2015). Better summarization evaluation with word embeddings for ROUGE. In Màrquez, L., Callison-Burch, C., Su, J., Pighin, D., and Marton, Y., editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1925–1930. The Association for Computational Linguistics.

Niklaus, C., Cetto, M., Freitas, A., and Handschuh, S. (2018). A survey on open information extraction. In Bender, E. M., Derczynski, L., and Isabelle, P., editors, *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 3866–3878. Association for Computational Linguistics.

Nivre, J., de Marneffe, M., Ginter, F., Goldberg, Y., Hajic, J., Manning, C. D., McDonald, R. T., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., and Zeman, D. (2016). Universal dependencies v1: A multilingual treebank collection. In Calzolari, N., Choukri, K., Declerck, T., Goggi, S., Grobelnik, M., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016*. European Language Resources Association (ELRA).

Oepen, S., Abend, O., Hajic, J., Hershcovich, D., Kuhlmann, M., O'Gorman, T., Xue, N., Chun, J., Straka, M., and Uresova, Z. (2019). MRP 2019: Cross-Framework Meaning Representation Parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 1–27, Stroudsburg, PA, USA. Association for Computational Linguistics.

Oepen, S., Kuhlmann, M., Miyao, Y., Zeman, D., Cinková, S., Flickinger, D., Hajic, J., and Uresová, Z. (2015). Semeval 2015 task 18: Broad-coverage semantic dependency parsing. In Cer, D. M., Jurgens, D., Nakov, P., and Zesch, T., editors, *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015*, pages 915–926. The Association for Computer Linguistics.

Oepen, S., Kuhlmann, M., Miyao, Y., Zeman, D., Flickinger, D., Hajic, J., Ivanova, A., and Zhang, Y. (2014). Semeval 2014 task 8: Broad-coverage semantic dependency parsing. In Nakov, P. and Zesch, T., editors, *Proceedings of the 8th International Workshop on Semantic Evaluation, SemEval@COLING 2014, Dublin, Ireland, August 23-24, 2014*, pages 63–72. The Association for Computer Linguistics.

Oepen, S. and Lønning, J. T. (2006). Discriminant-based mrs banking. In Calzolari, N., Choukri, K., Gangemi, A., Maegaard, B., Mariani, J., Odijk, J., and Tapias, D., editors, *Proceedings of the Fifth International Conference on Language Resources and Evaluation, LREC 2006, Genoa, Italy, May 22-28, 2006*, pages 1250–1255. European Language Resources Association (ELRA).

Oliveira, H., Ferreira, R., Lima, R., Lins, R. D., Freitas, F., Riss, M., and Simske, S. J. (2016). Assessing shallow sentence scoring techniques and combinations for single and multi-document summarization. *Expert Syst. Appl.*, 65:68–86.

Ono, K., Sumita, K., and Miike, S. (1994). Abstract generation based on rhetorical structure extraction. In *15th International Conference on Computational Linguistics, COLING 1994, Kyoto, Japan, August 5-9, 1994*, pages 344–348.

Otterbacher, J., Erkan, G., and Radev, D. R. (2009). Biased lexrank: Passage retrieval using random walks with question-based priors. *Information Processing and Management*, 45(1):42–54.

Over, P., Dang, H., and Harman, D. (2007). DUC in context. *Inf. Process. Manage.*, 43(6):1506–1520.

Owczarzak, K. and Dang, H. T. (2011). Overview of the tac 2011 summarization track: Guided task and aesop task. In *Proceedings of the Text Analysis Conference (TAC 2011), Gaithersburg, Maryland, USA, November*.

Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab. Previous number = SIDL-WP-1999-0120.

Paice, C. D. (1980). The automatic generation of literature abstracts: An approach based on the identification of self-indicating phrases. In Oddy, R. N., Robertson, S. E., van Rijsbergen, C. J., and Williams, P. W., editors, *Information Retrieval Research, Proc. Joint ACM/BCS Symposium in Information Storage and Retrieval, Cambridge, UK, June 1980*, pages 172–191. Butterworths.

Palmer, M. (2009). Semlink: Linking propbank, verbnet and framenet. In *Proceedings of the generative lexicon conference*, pages 9–15. Pisa Italy.

Palmer, M., Kingsbury, P., and Gildea, D. (2005). The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Parveen, D., Ramsl, H., and Strube, M. (2015). Topical coherence for graph-based extractive summarization. In Màrquez, L., Callison-Burch, C., Su, J., Pighin, D., and Marton, Y., editors, *Proceedings* *of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1949–1954. The Association for Computational Linguistics.

Pasupat, P. and Liang, P. (2015). Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1470–1480. The Association for Computer Linguistics.

Paulus, R., Xiong, C., and Socher, R. (2018). A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*, volume abs/1705.04304.

Peng, H., Chang, K., and Roth, D. (2015). A joint framework for coreference resolution and mention head detection. In Alishahi, A. and Moschitti, A., editors, *Proceedings of the 19th Conference on Computational Natural Language Learning, CoNLL 2015, Beijing, China, July 30-31, 2015*, pages 12–21. ACL.

Perone, C. S., Silveira, R., and Paula, T. S. (2018). Evaluation of sentence embeddings in downstream and linguistic probing tasks. *CoRR*, abs/1806.06259.

Peroni, S. and Vitali, F. (2009). Annotations with EARMARK for arbitrary, overlapping and out-of order markup. In Borghoff, U. M. and Chidlovskii, B., editors, *Proceedings of the 2009 ACM Symposium on Document Engineering, Munich, Germany, September 16-18, 2009*, pages 171–180. ACM.

Pighin, D., Cornolti, M., Alfonseca, E., and Filippova, K. (2014). Modelling events through memory-based, open-ie patterns for abstractive summarization. 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 892–901. The Association for Computer Linguistics.

Poesio, M., Ponzetto, S., and Versley, Y. (2011). Computational models of anaphora resolution: A survey.

Polanyi, L. (1988). A formal model of the structure of discourse. *Journal of Pragmatics*, 12(5):601 – 638.

Poon, H. (2013). Grounded unsupervised semantic parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*, pages 933–943. The Association for Computer Linguistics.

Pourdamghani, N., Knight, K., and Hermjakob, U. (2016). Generating english from abstract meaning representations. In Isard, A., Rieser, V., and Gkatzia, D., editors, *INLG 2016 - Proceedings of the Ninth International Natural Language Generation Conference, September 5-8, 2016, Edinburgh, UK*, pages 21–25. The Association for Computer Linguistics.

Pradhan, S., Moschitti, A., Xue, N., Uryupina, O., and Zhang, Y. (2012). Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *EMNLP-CoNLL Shared Task*, pages 1–40. ACL.

Pradhan, S., Ramshaw, L. A., Marcus, M. P., Palmer, M., Weischedel, R. M., and Xue, N. (2011). Conll-2011 shared task: Modeling unrestricted coreference in ontonotes. In *CoNLL Shared Task*, pages 1–27. ACL.

Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A. K., and Webber, B. L. (2008). The penn discourse treebank 2.0. In *Proceedings of the International Conference on Language Resources and Evaluation, LREC 2008, 26 May - 1 June 2008, Marrakech, Morocco*. European Language Resources Association.

Prasad, R., Webber, B. L., and Joshi, A. K. (2014). Reflections on the penn discourse treebank, comparable corpora, and complementary annotation. *Computational Linguistics*, 40(4):921–950.

Proudian, D. and Pollard, C. (1985). Parsing head-driven phrase structure grammar. In Mann, W. C., editor, *23rd Annual Meeting of the Association for Computational Linguistics, 8-12 July 1985, University of Chicago, Chicago, Illinois, USA, Proceedings.*, pages 167–171. ACL.

Punyakanok, V., Roth, D., and Yih, W. (2008). The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.

Qi, P., Dozat, T., Zhang, Y., and Manning, C. D. (2018). Universal dependency parsing from scratch. In Zeman, D. and Hajic, J., editors, *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, Brussels, Belgium, October 31 - November 1, 2018*, pages 160–170. Association for Computational Linguistics.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Reddy, S., Täckström, O., Collins, M., Kwiatkowski, T., Das, D., Steedman, M., and Lapata, M. (2016). Transforming dependency structures to logical forms for semantic parsing. *TACL*, 4:127–140.

Reddy, S., Täckström, O., Petrov, S., Steedman, M., and Lapata, M. (2017). Universal semantic parsing. In Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 89–101. Association for Computational Linguistics.

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. In Inui, K., Jiang, J., Ng, V., and

Wan, X., 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 3980–3990. Association for Computational Linguistics.

Reiter, E. and Dale, R. (1997). Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87.

Riedel, S., Yao, L., and McCallum, A. (2010). Modeling relations and their mentions without labeled text. In Balcázar, J. L., Bonchi, F., Gionis, A., and Sebag, M., editors, *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part III*, volume 6323 of *Lecture Notes in Computer Science*, pages 148–163. Springer.

Röder, M., Usbeck, R., and Ngomo, A. N. (2018). GERBIL - benchmarking named entity recognition and linking consistently. *Semantic Web*, 9(5):605–625.

Roth, M. and Lapata, M. (2015). Context-aware frame-semantic role labeling. *TACL*, 3:449–460.

Roth, M. and Woodsend, K. (2014). Composition of word representations improves semantic role labelling. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 407–413. ACL.

Rouces, J., de Melo, G., and Hose, K. (2017). Framebase: Enabling integration of heterogeneous knowledge. *Semantic Web*, 8(6):817–850.

Rozenshtein, P., Anagnostopoulos, A., Gionis, A., and Tatti, N. (2014). Event detection in activity networks. In Macskassy, S. A., Perlich, C., Leskovec, J., Wang, W., and Ghani, R., editors, *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 1176–1185. ACM.

Ruder, S., Peters, M. E., Swayamdipta, S., and Wolf, T. (2019). Transfer learning in natural language processing. In Sarkar, A. and Strube, M., 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, 2019, Tutorial Abstracts*, pages 15–18. Association for Computational Linguistics.

Ruppenhofer, J., Ellsworth, M., Petruck, M. R., Johnson, C. R., and Scheffczyk, J. (2006). *FrameNet II: Extended Theory and Practice*. International Computer Science Institute, Berkeley, California. Distributed with the FrameNet data.

Rush, A. M., Chopra, S., and Weston, J. (2015). A neural attention model for abstractive sentence summarization. In Màrquez, L., Callison-Burch, C., Su, J., Pighin, D., and Marton, Y., editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 379–389. The Association for Computational Linguistics.

Sagae, K., Miyao, Y., and Tsujii, J. (2007). HPSG parsing with shallow dependency constraints. In Carroll, J. A., van den Bosch, A., and Zaenen, A., editors, *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic*. The Association for Computational Linguistics.

Salazar, J., Liang, D., Nguyen, T. Q., and Kirchhoff, K. (2020). Masked language model scoring. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. R., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July*

5-10, 2020, pages 2699–2712. Association for Computational Linguistics.

Sandhaus, E. (2008). The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

Sarawagi, S. (2008). Information extraction. *Foundations and Trends in Databases*, 1(3):261–377.

Schick, T. (2017). Transition-based generation from abstract meaning representations. *CoRR*, abs/1707.07591.

Schuler, K. K. (2005). *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon*. PhD thesis, University of Pennsylvania. Actualizar - 200501; Última actualización - 2017-01-04; Primera página - 3241.

Schuster, T., Ram, O., Barzilay, R., and Globerson, A. (2019). Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In Burstein, J., Doran, C., and Solorio, T., 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 1599–1613. Association for Computational Linguistics.

See, A., Liu, P. J., and Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. In Barzilay, R. and Kan, M., 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 1073–1083. Association for Computational Linguistics.

Shen, L., Xu, J., and Weischedel, R. M. (2008). A new string-to-dependency machine translation algorithm with a target dependency language model. In McKeown, K. R., Moore, J. D., Teufel, S., Allan, J., and Furui, S., editors, *ACL 2008, Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, June*

15-20, 2008, Columbus, Ohio, USA, pages 577–585. The Association for Computer Linguistics.

Smith, A., Bohnet, B., de Lhoneux, M., Nivre, J., Shao, Y., and Stymne, S. (2018). 82 treebanks, 34 models: Universal dependency parsing with multi-treebank models. In Zeman, D. and Hajic, J., editors, *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, Brussels, Belgium, October 31 - November 1, 2018*, pages 113–123. Association for Computational Linguistics.

Soderland, S., Gilmer, J., Bart, R., Etzioni, O., and Weld, D. S. (2013). Open information extraction to KBP relations in 3 hours. In *Proceedings of the Sixth Text Analysis Conference, TAC 2013, Gaithersburg, Maryland, USA, November 18-19, 2013*. NIST.

Søgaard, A., Braud, C., and Coavoux, M. (2017). Cross-lingual rst discourse parsing. In Lapata, M., Blunsom, P., and Koller, A., editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, pages 292–304. Association for Computational Linguistics.

Song, L., Zhang, Y., Peng, X., Wang, Z., and Gildea, D. (2016). Amr-to-text generation as a traveling salesman problem. In Su, J., Carreras, X., and Duh, K., editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2084–2089. The Association for Computational Linguistics.

Soricut, R. and Marcu, D. (2003). Sentence level discourse parsing using syntactic and lexical information. In Hearst, M. A. and Ostendorf, M., editors, *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, HLT-NAACL 2003, Edmonton, Canada, May 27 - June 1, 2003*. The Association for Computational Linguistics.

Sporleder, C. and Lapata, M. (2005). Discourse chunking and its application to sentence compression. In *HLT/EMNLP 2005, Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, 6-8 October 2005, Vancouver, British Columbia, Canada*, pages 257–264. The Association for Computational Linguistics.

Steedman, M. (2000). *The Syntactic Process*. MIT Press, Cambridge, MA, USA.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Takase, S., Suzuki, J., Okazaki, N., Hirao, T., and Nagata, M. (2016). Neural headline generation on abstract meaning representation. In Su, J., Carreras, X., and Duh, K., editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1054–1059. The Association for Computational Linguistics.

Tan, J., Wan, X., and Xiao, J. (2017). Abstractive document summarization with a graph-based attentional neural model. In Barzilay, R. and Kan, M., 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 1171–1181. Association for Computational Linguistics.

Taylor, A., Marcus, M., and Santorini, B. (2003). The penn treebank: an overview. In *Treebanks*, pages 5–22. Springer.

Thadani, K. and McKeown, K. (2013). Supervised sentence fusion with single-stage inference. In *Sixth International Joint Conference on Nat-*

ural Language Processing, IJCNLP 2013, Nagoya, Japan, October 14-18, 2013, pages 1410–1418. Asian Federation of Natural Language Processing / ACL.

Torres-Moreno, J. (2014). *Automatic Text Summarization*. Wiley.

Trani, S., Ceccarelli, D., Lucchese, C., Orlando, S., and Perego, R. (2016). Sel: A unified algorithm for entity linking and saliency detection. In Sablatnig, R. and Hassan, T., editors, *Proceedings of the 2016 ACM Symposium on Document Engineering, DocEng 2016, Vienna, Austria, September 13 - 16, 2016*, pages 85–94. ACM.

Usbeck, R., Ngomo, A. N., Röder, M., Gerber, D., Coelho, S. A., Auer, S., and Both, A. (2014). AGDISTIS - agnostic disambiguation of named entities using linked open data. In Schaub, T., Friedrich, G., and O'Sullivan, B., editors, *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*, volume 263 of *Frontiers in Artificial Intelligence and Applications*, pages 1113–1114. IOS Press.

Vanderwende, L., Menezes, A., and Quirk, C. (2015). An AMR parser for english, french, german, spanish and japanese and a new amr-annotated corpus. In Mihalcea, R., Chai, J. Y., and Sarkar, A., editors, *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 26–30. The Association for Computational Linguistics.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Vilca, G. C. V. and Cabezudo, M. A. S. (2017). A study of abstractive summarization using semantic representations and discourse level information. In Ekstein, K. and Matousek, V., editors, *Text, Speech, and Dialogue - 20th International Conference, TSD 2017, Prague, Czech Republic, August 27-31, 2017, Proceedings*, volume 10415 of *Lecture Notes in Computer Science*, pages 482–490. Springer.

Walker, M. A., Rambow, O., and Rogati, M. (2001). Spot: A trainable sentence planner. In *Language Technologies 2001: The Second Meeting of the North American Chapter of the Association for Computational Linguistics, NAACL 2001, Pittsburgh, PA, USA, June 2-7, 2001*. The Association for Computational Linguistics.

Wan, X. (2010). Towards a unified approach to simultaneous single-document and multi-document summarizations. In Huang, C. and Jurafsky, D., editors, *COLING 2010, 23rd International Conference on Computational Linguistics, Proceedings of the Conference, 23-27 August 2010, Beijing, China*, pages 1137–1145. Tsinghua University Press.

Wan, X. and Yang, J. (2006). Improved affinity graph based multi-document summarization. In Moore, R. C., Bilmes, J. A., Chu-Carroll, J., and Sanderson, M., editors, *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 4-9, 2006, New York, New York, USA*. The Association for Computational Linguistics.

Wang, C., Xue, N., and Pradhan, S. (2015). A transition-based algorithm for AMR parsing. In Mihalcea, R., Chai, J. Y., and Sarkar, A., editors, *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 366–375. The Association for Computational Linguistics.

Wang, D., Li, T., Zhu, S., and Ding, C. H. Q. (2008). Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization. In Myaeng, S., Oard, D. W., Sebastiani, F., Chua, T., and Leong, M., editors, *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, Singapore, July 20-24, 2008*, pages 307–314. ACM.

Wang, Y., Li, S., and Wang, H. (2017). A two-stage parsing method for text-level discourse analysis. In Barzilay, R. and Kan, M., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*, pages 184–188. Association for Computational Linguistics.

Wang, Y., Ren, Z., Theobald, M., Dylla, M., and de Melo, G. (2016). Summary generation for temporal extractions. In Hartmann, S. and Ma, H., editors, *Database and Expert Systems Applications - 27th International Conference, DEXA 2016, Porto, Portugal, September 5-8, 2016, Proceedings, Part I*, volume 9827 of *Lecture Notes in Computer Science*, pages 370–386. Springer.

Weissenborn, D., Hennig, L., Xu, F., and Uszkoreit, H. (2015). Multi-objective optimization for the joint disambiguation of nouns and named entities. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 596–605. The Association for Computer Linguistics.

Wities, R., Shwartz, V., Stanowsky, G., Adler, M., Shapira, O., Upadhyay, S., Roth, D., Martínez Cámara, E., Gurevych, I., and Dagan, I. (2017). A consolidated open knowledge representation for multiple texts. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, pages 12–24. EACL, Association for Computational Linguistics.

Xue, N., Ng, H. T., Pradhan, S., Prasad, R., Bryant, C., and Rutherford, A. (2015). The conll-2015 shared task on shallow discourse parsing. In Xue, N., Ng, H. T., Pradhan, S., Prasad, R., Bryant, C., and Rutherford, A., editors, *Proceedings of the 19th Conference on Computational Natural Language Learning: Shared Task, CoNLL 2015, Beijing, China, July 30-31, 2015*, pages 1–16. ACL.

Xue, N., Ng, H. T., Pradhan, S., Rutherford, A., Webber, B. L., Wang, C., and Wang, H. (2016). Conll 2016 shared task on multilingual shallow discourse parsing. In Xue, N., Ng, H. T., Pradhan, S., Rutherford, A., Webber, B. L., Wang, C., and Wang, H., editors, *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning: Shared Task, CoNLL 2016, Berlin, Germany, August 7-12, 2016*, pages 1–19. ACL.

Yan, S. and Wan, X. (2014). Srrank: leveraging semantic roles for extractive multi-document summarization. *IEEE/ACM Trans. Audio, Speech & Language Processing*, 22(12):2048–2058.

Yan, S. and Wan, X. (2015). Deep dependency substructure-based learning for multidocument summarization. *ACM Trans. Inf. Syst.*, 34(1):3:1–3:24.

Yang, Q., Passonneau, R. J., and de Melo, G. (2016). PEAK: pyramid evaluation via automated knowledge extraction. In Schuurmans, D. and Wellman, M. P., editors, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, pages 2673–2680. AAAI Press.

Yao, J., Wan, X., and Xiao, J. (2017). Recent advances in document summarization. *Knowl. Inf. Syst.*, 53(2):297–336.

Yates, A., Banko, M., Broadhead, M., Cafarella, M. J., Etzioni, O., and Soderland, S. (2007). Textrunner: Open information extraction on the web. In Sidner, C. L., Schultz, T., Stone, M., and Zhai, C., editors, *Human Language Technology Conference of the North American Chapter* *of the Association of Computational Linguistics, Proceedings, April 22-27, 2007, Rochester, New York, USA*, pages 25–26. The Association for Computational Linguistics.

Yin, W. and Pei, Y. (2015). Optimizing sentence modeling and selection for document summarization. In Yang, Q. and Wooldridge, M. J., editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 1383–1389. AAAI Press.

Yoshida, Y., Suzuki, J., Hirao, T., and Nagata, M. (2014). Dependency-based discourse parser for single-document summarization. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1834–1839. ACL.

Zhang, C. and Weld, D. S. (2013). Harvesting parallel news streams to generate paraphrases of event relations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1776–1786. ACL.

Zhang, F., Yao, J., and Yan, R. (2018). On the abstractiveness of neural document summarization. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 785–790. Association for Computational Linguistics.

Zhang, K. and Shasha, D. E. (1989). Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing*, 18(6):1245–1262.

Zhang, Y., Zhong, V., Chen, D., Angeli, G., and Manning, C. D. (2017). Position-aware attention and supervised data improve slot filling. In

Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 35–45. Association for Computational Linguistics.

Zhou, Y. and Xue, N. (2012). Pdtb-style discourse annotation of chinese text. In *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, July 8-14, 2012, Jeju Island, Korea - Volume 1: Long Papers*, pages 69–77. The Association for Computer Linguistics.