# REPRESENTATION AND CONTEXTUALIZATION

# FOR DOCUMENT UNDERSTANDING

Von der Fakultät für Elektrotechnik und Informatik
der Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des Grades

DOKTOR DER NATURWISSENSCHAFTEN

**Dr. rer. nat.**

genehmigte Dissertation
von

**M.Sc. Nam Khanh Tran**

geboren am 02. September 1987, in Hai Duong, Vietnam

Hannover, Deutschland, 2019

**Referent: Prof. Dr. techn. Wolfgang Nejdl**
**Korreferent: Prof. Dr. Yannis Velegrakis**
**Korreferent: Prof. Dr. Kurt Schneider**
**Tag der Promotion: 04.02.2019**

# ABSTRACT

Document understanding requires discovery of meaningful patterns in text, which in turn involves analyzing documents and extracting useful information for a certain purpose. There is a multitude of problems that need to be dealt with to solve this task. With the goal of improving document understanding, we identify three main problems to study within the scope of this thesis. The first problem is about learning text representation, which is considered as starting point to gain understanding of documents. The representation enables us to build applications around the semantics or meaning of the documents, rather than just around the keywords presented in the texts. The second problem is about acquiring document context. A document cannot be fully understood in isolation since it may refer to knowledge that is not explicitly included in its textual content. To obtain a full understanding of the meaning of the document, that prior knowledge, therefore, has to be retrieved to supplement the text in the document. The last problem we address is about recommending related information to textual documents. When consuming text especially in applications such as e-readers and Web browsers, users often get attracted by the topics or entities appeared in the text. Gaining comprehension of these aspects, therefore, can help users not only further explore those topics but also better understand the text.

In this thesis, we tackle the aforementioned problems and propose automated approaches that improve document representation, and suggest relevant as well as missing information for supporting interpretations of documents. To this end, we make the following contributions as part of this thesis:

- *Representation learning* – the first contribution is to improve document representation which serves as input to document understanding algorithms. Firstly, we adopt probabilistic methods to represent documents as a mixture of topics and propose a generalizable framework for improving the quality of topics learned from small collections. The proposed method can be well adapted to different application domains. Secondly, we focus on learning the distributed representation of documents. We introduce multiplicative tree-structured Long Short-Term Memory (LSTM) networks which are capable of integrating syntactic and semantic information from text into the standard LSTM architecture for improved representation learning. Finally, we investigate the usefulness of attention mechanism for enhancing distributed representations. In particular, we propose Multihop Attention Networks which can learn effective representations and illustrate its usefulness in the application of question answering.
- *Time-aware contextualization* – the second contribution is to formalize the novel and challenging task of time-aware contextualization, where explicit context information is required for bridging the gap between the situation at the time of content creation and the situation at the time of content digestion. To solve this task, we propose a novel approach which automatically formulates queries for retrieving adequate contextualization candidates from an underlying knowledge source such as Wikipedia, and then ranks the candidates using learning-to-rank algorithms.
- *Context-aware entity recommendation* – the third contribution is to give assistance to document exploration by recommending related entities to the entities mentioned in the documents. For this purpose, we first introduce the idea of a contextual relatedness of entities and formalize the problem of context-aware entity recommendation. Then, we approach the problem by a statistically sound probabilistic model incorporating temporal and topical context via embedding methods.

**Keywords:** *document understanding, representation learning, time-aware contextualization, context-aware entity recommendation*

# ZUSAMMENFASSUNG

Es ist beim Dokumentverständnis erforderlich, sinnvolle Textbausteine im Dokument zu entdecken. Dies umfasst die Analyse des Dokuments und das Extrahieren von nützlichen Informationen für bestimmte Zwecke. Mit dem Ziel, das Dokumentverständnis zu verbessern, haben wir uns im Rahmen dieser Abschlussarbeit mit drei wesentlichen Aufgabenstellungen auseinandergesetzt. Die erste Aufgabenstellung bezieht sich auf das Lernen von Textrepräsentation, die als Startpunkt zum Gewinnen vom Dokumentverständnis gilt. Die Textrepräsentation ermöglicht uns, Anwendungen rund um die Semantik bzw. Bedeutung des Dokuments anstatt lediglich rund um die im Text enthaltenen Stichwörtern zu entwickeln. Die zweite Aufgabenstellung betrifft die Bereitstellung vom Dokumentkontext. Man kann ein Dokument bei isolierter Verarbeitung nicht vollständig nachvollziehen, denn es könnte sich auf (Vor-)Kenntnisse, die nicht explizit im Text enthalten sind, beziehen. Um das Dokument vollständig zu verstehen, müssen derartige Vorkenntnisse zur Ergänzung des Textes im Dokument abgerufen werden. Die dritte Aufgabenstellung geht auf die Empfehlung von relevanten Informationen zum Dokument ein. Bei Verarbeitung von Texten in Anwendungen wie E-readers und Webbrowsers lassen sich die Benutzer häufig von den im Text aufgetauchten Themen und Entities anziehen. Mithilfe der Verschaffung vom Verständnis dieser Aspekte werden die Benutzer in der Lage sein, nicht nur die erwähnten Themen weiter zu untersuchen, sondern auch den Text besser zu verstehen.

In dieser Abschlussarbeit befassen wir uns mit den obengenannten Aufgabenstellungen und schlagen automatisierte Ansätze zur Verbesserung der Textrepräsentation sowie zur Empfehlung fehlender und relevanter Kontexte, die die Interpretation von Dokumenten unterstützen, vor. Zu diesem Zweck leisten wir folgende Beiträge, die als Teil dieser Abschlussarbeit dargestellt werden:

- *Lernen von Textrepräsentation* – der erste Beitrag geht auf die Verbesserung der Textrepräsentation ein, die als Input für Dokumentenverständnis-Algorithmen dient. Zum Ersten wenden wir probabilistische Methoden an, um Dokumente als eine Mischung von Themen zu repräsentieren, und schlagen ein generalisierbares Framework zur Steigerung der Themenqualität beim Lernen auf kleinen Datensätzen vor. Die vorgeschlagene Methode kann gut geeignet für verschiedene Anwendungsdomäne sein. Zum Zweiten legen wir den Fokus auf das Lernen von der vektorisierten Repräsentation von Dokumenten. Wir stellen die multiplikativen baumstrukturierten Long Short-Term Memory (LSTM) Networks vor, die syntaktische und semantische Informationen aus dem Text in die LSTM-Standardarchitektur integrieren können, um das Lernen von Repräsentation verbessern. Zuletzt untersuchen wir die Nützlichkeit von Attention Mechanism, um die vektorisierte Dokumentrepräsentation zu verstärken. Wir stellen insbesondere die Multihop Attention Networks vor, die dazu fähig sind, effektive Repräsentationen zu lernen und die Effektivität in Question Answering-Anwendung nachzuweisen.

- *Zeitbewusste Kontextualisierung* – der zweite Beitrag fokussiert sich auf die Formalisierung der neuen und herausfordernden Aufgabe der Time-aware contextualization (zeitbewussten Kontextualisierung), wobei explizite Kontextinformationen erforderlich sind, um die Lücke zwischen der Situation im Zeitpunkt der Inhaltserstellung und der Situation im Zeitpunkt der Inhaltsverarbeitung zu überbrücken. Als Lösung zu dieser Aufgabe schlagen wir einen neuen Ansatz vor, der automatisch Abfragen nach angemessenen Kandidaten zur Kontextualisierung aus einer grundlegenden Wissensbasis, z.B. Wikipedia, generiert, und im Anschluss die Kandidaten anhand von learning-to-rank-Algorithmen einstuft.

- *Kontextbewusste Entitätsempfehlung* – der dritte Beitrag bezieht sich auf die Unterstützung von Dokumentuntersuchung durch Empfehlung von Entities, die relevant zu den im Doku-

ment enthaltenen Entities sind. Hierzu stellen wir die Idee eines kontextuellen Zusammenhangs zwischen Entities vor und formalisieren die Aufgabestellung der Context-aware entity recommendation (kontextbewussten Entitätsempfehlung). Als Lösungsvorschlag präsentieren wir ein statistisch fundiertes probabilistisches Modell, das sich zeitlicher und thematischer Kontexte anhand von Embedding methods (Einbettungsmethoden) bedient.

**Schlagwörter:** *Dokumentverständnis, Lernen von Textrepräsentation, zeitbewusste Kontextualisierung, kontextbewusste Entitätsempfehlung*

# ACKNOWLEDGMENTS

During my doctoral program, I have had the opportunities to work with and learn from many great mentors, colleagues, and friends.

First and foremost, I would like to thank my advisor Prof. Dr. techn. Wolfgang Nejdl. He provided the perfect environment and invaluable guidance throughout these years. I especially enjoyed the freedom he gave me to pursue my research interests, helping me shape as a researcher and successfully conduct the work published in this thesis. I also thank Prof. Dr. Yannis Velegrakis and Prof. Dr. Kurt Schneider for agreeing to consider and evaluate my PhD thesis.

Special thanks to Dr. Claudia Niederée, for her close collaboration, the countless discussions and invaluable suggestions which helped me learn and develop as a researcher. I am also very grateful to Prof. Dr. Nattiya Kanhabua and Dr. Sergej Zerr for their guidance and introducing me to many exciting topics, projects, and providing helpful feedback and discussions.

I am indebted to Andrea Ceroni, Tuan Tran, Dat Nguyen, Giang Tran and Tuan-Anh Hoang for their contribution to my work. A very special thank to them and all the exceptional researchers with whom I had chance to collaborate. Many thanks to my officemates and to all my colleagues and staff at L3S Research Center for making the workplace an exciting atmosphere.

I learned a lot during the internship I did at Amazon Core Machine Learning, Berlin. I want to thank everyone in the NLP team, especially Weiwei Cheng and Alexandre Klementiev for their very helpful feedback and discussions.

A special note of thanks to Cam Tu, for her unconditional support and being there for me during the most important part of my PhD. She was the safe haven and the escape from the hectic period of countless experiments, late working hours that came along with the PhD.

Last but not least, I would like to thank my family for their unconditional love, support and tremendous patience. This was all possible because of you, and I dedicate this to you all.

# FOREWORD

The methods and algorithms presented in this thesis have been published at various conferences, as follows:

Chapter 3 addresses the problem of deriving semantic representation of documents by exploiting document content and structure, and describes the contributions included in:

- Nam Khanh Tran, Sergej Zerr, Kerstin Bischoff, Claudia Niederée, Ralf Krestel. *Topic Cropping: Leveraging Latent Topics for the Analysis of Small Corpora.* In Proceedings of the International Conference on Theory and Practice of Digital Libraries, TPDL 2013, volume 8092 of Lecture Notes in Computer Science, pages 297-308. [TZB+13b]

- Nam Khanh Tran, Weiwei Cheng. *Multiplicative Tree-Structured Long Short-Term Memory Networks for Semantic Representations.* In Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics, *SEM 2018, pages 276–286. [TC18]

- Nam Khanh Tran, Claudia Niederée. *Multihop Attention Networks for Question Answer Matching.* The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2018, pages 325–334. [TN18b]

Chapter 4 focuses on bridging temporal context gaps for supporting interpretations of documents and builds upon the work published in:

- Nam Khanh Tran, Andrea Ceroni, Nattiya Kanhabua, Claudia Niederée. *Back to the Past: Supporting Interpretations of Forgotten Stories by Time-aware Re-Contextualization.* In Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, WSDM 2015, pages 339-348. [TCKN15a]

- Nam Khanh Tran, Andrea Ceroni, Nattiya Kanhabua, Claudia Niederée. *Time-travel Translator: Automatically Contextualizing News Articles.* In Proceedings of the 24th International Conference on World Wide Web, WWW 2015 Companion, pages 247-250. [TCKN15b]

- Andrea Ceroni, Nam Khanh Tran, Nattiya Kanhabua, Claudia Niederée. *Bridging Temporal Context Gaps Using Time-aware Re-contextualization*. In Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval, SIGIR 2014, pages 1127-1130. [CTKN14]

Chapter 5 addresses the problem of supporting document exploration via contextual entity relatedness and entity recommendation and includes the contribution published in:

- Nam Khanh Tran, Tuan Tran, Claudia Niederée. *Beyond Time: Dynamic Context-Aware Entity Recommendation*. The Semantic Web - 14th International Conference, ESWC 2017, pages 353-368. [TTN17] (**Nomination for best paper award**)

During the course of the doctoral studies I have also published and co-authored a number of papers touching different aspects of content analytics, information retrieval and machine learning. Not all aspects are discussed in this thesis due to space limitation. The complete list of publications is as follows:

**Published journal articles**

- Elia Bruni, Nam Khanh Tran, Marco Baroni. *Multimodal Distributional Semantics*. In Journal of Artificial Intelligence Research, Volume 49 Issue 1, January 2014, pages 1-47. [BTB14] (**2017 IJCAI-JAIR best paper prize**)
- Dat Ba Nguyen, Abdalghani Abujabal, Nam Khanh Tran, Martin Theobald, Gerhard Weikum. *Query-Driven On-The-Fly Knowledge Base Construction*. In Proceedings of the VLDB Endowment, PVLDB 2017, pages 66-79 [NAT+17]

**Papers published in conference proceedings**

- Nam Khanh Tran, Claudia Niederée. *Multihop Attention Networks for Question Answer Matching*. The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2018, pages 325–334. [TN18b]
- Nam Khanh Tran, Weiwei Cheng. *Multiplicative Tree-Structured Long Short-Term Memory Networks for Semantic Representations.* In Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics, *SEM 2018, pages 276–286. [TC18]
- Nam Khanh Tran, Claudia Niederée. *A Neural Network-based Framework for Non-factoid Question Answering*. In Companion Proceedings of the The Web Conference, WWW 2018, pages 1979-1983. [TN18a]

- Nam Khanh Tran, Tuan Tran, Claudia Niederée. *Beyond Time: Dynamic Context-Aware Entity Recommendation*. The Semantic Web - 14th International Conference, ESWC 2017, pages 353-368. [TTN17] (**Nomination for best paper award**)

- Nattiya Kanhabua, Philipp Kemkes, Wolfgang Nejdl, Tu Ngoc Nguyen, Felipe Reis, Nam Khanh Tran. *How to Search the Internet Archive Without Indexing It*. In Proceeding of the 20th International Conference on Theory and Practice of Digital Libraries, TPDL 2016, pages 147-160. [KKN+16]

- Tuan Tran, Nam Khanh Tran, Asmelash Teka Hadgu, Robert Jäschke. *Semantic Annotation for Microblog Topics Using Wikipedia Temporal Information*. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, pages 97-106. [TTTHJ15]

- Nam Khanh Tran, Andrea Ceroni, Nattiya Kanhabua, Claudia Niederée. *Back to the Past: Supporting Interpretations of Forgotten Stories by Time-aware Re-Contextualization*. In Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, WSDM 2015, pages 339-348. [TCKN15a]

- Nam Khanh Tran, Andrea Ceroni, Nattiya Kanhabua, Claudia Niederée. *Time-travel Translator: Automatically Contextualizing News Articles*. In Proceedings of the 24th International Conference on World Wide Web, WWW 2015 Companion, pages 247-250. [TCKN15b]

- Andrea Ceroni, Nam Khanh Tran, Nattiya Kanhabua, Claudia Niederée. *Bridging Temporal Context Gaps Using Time-aware Re-contextualization*. In Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2014, pages 1127-1130. [CTKN14]

- Nam Khanh Tran, Sergej Zerr, Kerstin Bischoff, Claudia Niederée, Ralf Krestel. *Topic Cropping: Leveraging Latent Topics for the Analysis of Small Corpora.* In Proceedings of the International Conference on Theory and Practice of Digital Libraries, TPDL 2013, volume 8092 of Lecture Notes in Computer Science, pages 297-308. [TZB+13b]

- Nam Khanh Tran. *Time-aware Topic-based Contextualization*. In Proceedings of the 23rd International Conference on World Wide Web, WWW 2014 Companion, page 15-20. [Tra14]

- Kerstin Bischoff, Claudia Niederée, Nam Khanh Tran, Sergej Zerr, Peter Birke, Kerstin Brückweh, Wiebke Wiede. *Exploring Qualitative Data for Secondary Analysis: Challenges, Methods, and Technologies*. In Proceedings of the 2014 Digital Humanities Conference. [BNT+14]

- Khaled Hossain Ansary, Anh Tuan Tran, Nam Khanh Tran. *A pipeline tweet contextualization system at INEX 2013*. In Working Notes for CLEF 2013 Conference. [ATT13]

**Papers published in workshop proceedings**

- Giang Binh Tran, Tuan A. Tran, Nam Khanh Tran, Mohammad Alrifai, Nattiya Kanhabua. *Leveraging Learning To Rank in an Optimization Framework for Timeline Summarization* In SIGIR 2013 Workshop on Time-aware Information Access (TAIA 2013). [TTT+13]

- Sergej Zerr, Nam Khanh Tran, Kerstin Bischoff, Claudia Niederée. *Sentiment Analysis and Opinion Mining in Collections of Qualitative Data*. In Proceedings of the 1st International Workshop on Archiving Community Memories at iPRESS 2013. [ZTBN13]

- Nam Khanh Tran, Sergej Zerr, Kerstin Bischoff, Claudia Niederée, Ralf Krestel. *"Gute Arbeit": Topic Exploration and Analysis Challenges for the Corpora of German Qualitative Studies*. In Exploration, Navigation and Retrieval of Information in Cultural Heritage (ENRICH), Workshop at SIGIR 2013, pages 15-22. [TZB+13a]

# Contents

# List of Figures

# List of Tables

# 1 Introduction

*"One morning I shot an elephant in my pajamas. How he got in my pajamas, I don't know."*

*– Groucho Marx*

## 1.1 Motivation

Every day, a huge amount of data is produced in a variety of forms of text, such as books, news articles, social media posts and more. In fact, we are now overwhelmed with textual data, which keep increasing day by day. Between the birth of the Internet and 2003, year of birth of social networks such as Delicious, LinkedIn, and Facebook, just a few dozen exabytes of text were created on the Web. Today, this same amount of textual content is created weekly. It is estimated that the data volume will grow to 40 zettabytes by 2020 [GR12]. With the explosive growth in the number of such textual documents, it is an acute mission to assist users in exploring, analyzing and discovering knowledge from documents with automated text mining methods and systems. These methods require a deep understanding of natural languages by machines.

The field of text understanding, which studies automatic means of capturing the semantics of textual content, plays a central part in the long-term goal of artificial intelligence (AI) research. The task encompasses many subtasks, including text matching [HLLC14, YAGC16, WJ17], question answering [WBC+16, XMS16, CFWB17], document summarization [RCW15, PXS18], contextualization [CTKN14, TCKN15a], and machine translation [KOM03, BCB15, SVL14]. To solve these tasks, most approaches rely on some forms of text representation such as bag of words and distributed vector representation. Early approaches relied on the former representation of documents, i.e. word counts and human input in the form of heuristics and sometimes hand-made rules [MDM07, II08]. While these hand-crafted features are well motivated and carefully designed, they often require prior knowledge of the application domains. Moreover, their performance is limited by the incompleteness of the hand-crafted features. Recent text un-

derstanding algorithms advance towards capturing the semantics of textual content from scratch with more advanced text representations [HKG+15, XMS16]. Such representations have achieved some improvements in various tasks while not requiring much domain knowledge [CWB+11, BCV13]. Efficient methods for the representation learning have therefore become increasingly important for AI applications. Hence, the question central to the first part of this thesis is how to further improve representation learning for document understanding tasks.

Text understanding or comprehension, from a human perspective, is not just the by-product of accurate word recognition. Instead, text comprehension can be viewed as a complex process which requires active and intentional cognitive effort on the part of the reader [BRVB12]. It involves the incremental construction and updating of a mental representation of the situation described in the text [Kin98]. However, when the context under which the texts are constructed is missing, the reader might construct wrong or uncompleted interpretations. More specifically, many textual documents are generated in certain context and time periods, and can be best understood with the models of this information in mind. When the context and time changes, the content can be inconsistent if digested in isolation, making it hard for users to fully construct the meanings from the words as well as the whole documents. A good example of this is the word *"computer"*, which used to refer to a person employed to do computations, a meaning which many people today are unaware of. Another example is the advertisement poster of cigarette companies from the 1950s *"More Doctors Smokes CAMELS than any other cigarette!"*. From today's perspective, it is more than surprising that doctors would recommend smoking. It can, however, be understood with the context information of that time which has been extracted from the Wikipedia article on tobacco advertising *"Prior to 1964, many of the cigarette companies advertised their brand by claiming that their product did not have serious health risks. Such claims were made both to increase the sales of their product..."*. Therefore, the question we want to address in the second part of this thesis is about how we can retrieve the original context under which documents were created by automated methods to support interpretations of textual documents.

Furthermore, when consuming a textual document, in many cases users are attracted by specific concepts or entities mentioned in the document instead of the document in general. In consequence, they wish to see related information to those entities. For example, when users are reading an article about the movie *"World War Z"* starring Brad Pitt, they likely want to see either other movies acted by Brad Pitt or other co-starring actors in the movie. In order to accomplish this goal, we need to answer several questions such as how such related entities can be retrieved; whether or not they are dependent on the content of the document. In the last part of this thesis, we aim to tackle these questions by introducing the notion of contextual entity relatedness and proposing different approaches to context-aware entity recommendation, where a list of related entities is presented to the entity of interest under a given context. The related entities, as consequence, can not only provide increased user experience in document exploration, but also help users better understand the text in the document.

To sum up, despite the fact that computer science and computational linguistics scientists have been working on document understanding tasks for years, there is still a multitude of issues that need to be dealt with. Three main of them - which focus on representation learning, re-contextualization, and related entity recommendation - are addressed in this work.

## 1.2 Research Outline and Questions

In the following, we elaborate the three main problems addressed in this thesis for supporting the interpretations of documents: (i) document representation, (ii) document contextualization, and (iii) document exploration via entity recommendation.

**(I)** Text understanding starts with the challenge of learning machine-understandable representation that captures the semantics of texts. Bag-of-words (BoW) and its N-gram extensions are arguably the most commonly used document representations. Despite its simplicity, BoW works considerably well for many tasks [WM12]. However, by treating words and phrases as unique and discrete symbols, BoW often fails to capture the similarity between words or phrases and also suffers from sparsity and high dimensionality. Various dimension reduction techniques including Latent Semantic Indexing (LSI) [DDF+90] was proposed to tackle these problems. LSI represents the semantics of text documents through the linear combination of terms, which is computed by the Singular Value Decomposition (SVD) [KL80]. However, the high complexity of SVD [ABB00] makes LSI rarely used in real-world applications. In addition, LSI and other similar techniques also lose the innate interpretability of the bag-of-words approach. Moreover, such representations neglect potential semantic links between words. In order to overcome the limitations of the bag-of-words approach, many models have been proposed recently including Probabilistic Latent Semantic Indexing (PLSI) [Hof99] or Latent Dirichlet Allocation [BNJ03] and distributed representation learning approaches [LM14].

Motivated by the LSI, the Probabilistic Latent Semantic Indexing (PLSI) [Hof99] and its extension - Latent Dirichlet Allocation [BNJ03] are proposed for representing the semantics of text documents, in which documents are represented as a mixture of topics, where a topic is a probability distribution over words. In contrast to LSI, the latent dimensions in PLSI and LDA are topics which are much more interpretable. However, a key weakness of topic modeling is that it needs a large amount of data (e.g., thousands of documents) to provide reliable statistics to generate coherent topics. In practice, many document collections do not have so many documents. Given a small number of documents, classic topic modeling algorithms often generate very poor topics [CL14]. Hence, in this thesis, we want to address this problem for improving the topic quality for small collections of documents. In particular, we aim to study the following research question:

**RQ1.1.** *How to improve the topic quality in terms of coherence and diversity when applying topic modeling algorithms to small collections of documents?*

Recent works on using neural networks to learn distributed vector representations of words have gained great popularity. The well-known Word2Vec [MCCD13], by learning to predict the target word using its neighboring words, maps words of similar meanings to nearby points in the continuous vector space. To generalize the idea for learning vector representations for long spans of text such as sentences and documents, various approaches have been proposed recently [LM14, TSM15, KGB14]. In [LM14], Le and Mikolov proposed to learn paragraph vectors in which a target word is predicted by the word embeddings of its neighbors together with a unique document vector learned for each document. The approach outperforms established document representations such as BoW and LDA [BNJ03] on various text understanding tasks [DOL15]. In addition, there is another line of work for learning task-specific document representation with deep neural networks, which are typically based on Convolutional Neural Networks (CNN) [KGB14] or Long Short-Term Memory (LSTM) networks [HS97]. However, these approaches often ignore the linguistic knowledge such as syntactic information of text documents, which has been shown leading to better representations [TSM15]. We formalize the research question addressing this problem as follows:

**RQ1.2.** *How to improve representation learning by exploiting syntactic and semantic information in neural network models?*

The general idea of applying neural networks based approaches to text understanding tasks is that input sequences are first encoded into fixed-length internal representations by employing CNNs or LSTMs. These representations are then utilized as input features in the downstream tasks. Though LSTM or CNN based models outperform other representation learning approaches (e.g. LDA), they still suffer from an important issue. They are limited on the length of input sequences that can be reasonably learned and results in worse performance for very long input sequences [TdSXZ16]. Therefore, in this thesis we seek to overcome this limitation with the help of attention mechanism [BCB15] and investigate its effectiveness in the application of question answering. In particular, we aim to tackle the following research question:

**RQ1.3.** *How to improve distributed representation learning by using attention mechanism?*

**(II)** A broad model of text comprehension should not only simulate how information is extracted from the text itself, but also how this information is interpreted in light of the readers' knowledge [FKNV07]. The interpretation might require context knowledge from the time of document creation. Indeed, without context words have no meaning and the same is true for documents, in that often a wider context is required to fully interpret the information they contain. Hence, with the aim of supporting interpretations of text documents, in the second part of this thesis we introduce the problem of time-aware re-contextualization, where explicit context information is required for bridging the gap between the situation at the time of content creation and the situation at the time of content digestion. This includes changes in background knowledge, the societal and political situation, language, technology, and simply the forgetting of the original knowledge about the context. Text

contextualization differs from text expansion in that it aims at helping a human to understand a text rather than a system to better perform its tasks. For example, in the case of query expansion in information retrieval, the idea is to add terms to the initial query that will help the system to better select the documents to be retrieved. Text contextualization on the contrary can be viewed as a way to provide more information on the corresponding text to make it understandable and to relate this text to information that explains it. Specifically, we formalize the research question addressing this problem as follows:

**RQ2.** *How to bridge temporal context gaps for supporting interpretations of documents by time-aware re-contextualization?*

For this question, several subgoals of the information search process have to be combined with each other. First, the context information has to be relevant and complement the information already available in the document. Second, it has to consider the time of creation (or reference) of the document. Furthermore, the set of collected context information should be concise to avoid overloading the user.

**(III)** As we briefly discussed in the previous section, when consuming content in applications such as e-readers, word processors, and Web browsers, users often get attracted by the topics or concepts mentioned in the content. As an additional example, consider an user who is reading a news article on *President Obama's address to the nation on the Syrian crisis*. At some point, the user may highlight the entity Russia and ask the system for contextual insights. The notion of contextual insights is to provide users with additional information ("insights") that is contextually relevant to the content that they are consuming. In this example, good insights for the entity Russia are clearly dependent on the context of document that the user is reading. It is close to the problem in (**II**), however unlike previous approaches which aim to gain overall understanding of documents, here we focus on a fine-grained but important aspect of documents, i.e., entities. The goal is to recommend a list of related entities to the entity of interest when users are consuming texts. In particular, we aim to answer the following research question:

**RQ3.** *How to support document exploration and comprehension by recommending contextually related entities?*

For this question, several tasks have to be considered. The first task is to find an appropriate representation for context and to model the notion of contextual relatedness. Then, the next task will be to leverage this notion for suggesting related entities. Furthermore, how to effectively present and visualize suggested information to users is another challenging task to work on.

## 1.3 Main Contributions

In this thesis, we study the research questions formalized in the previous section and make three principal contributions to the field of document understanding. The first is to propose different approaches to enhance document representations, which then serve as inputs to document understanding algorithms. The second is to frame the novel and challenging problem of re-contextualization and propose a novel approach for retrieving contextualizing information to support the understanding of documents in presence of wide temporal and contextual gaps. The third contribution is to recommend contextual related entities to support document exploration. Figure 1.1 shows an outline of our contributions and the proposed solutions for the problems listed in Section 1.2.

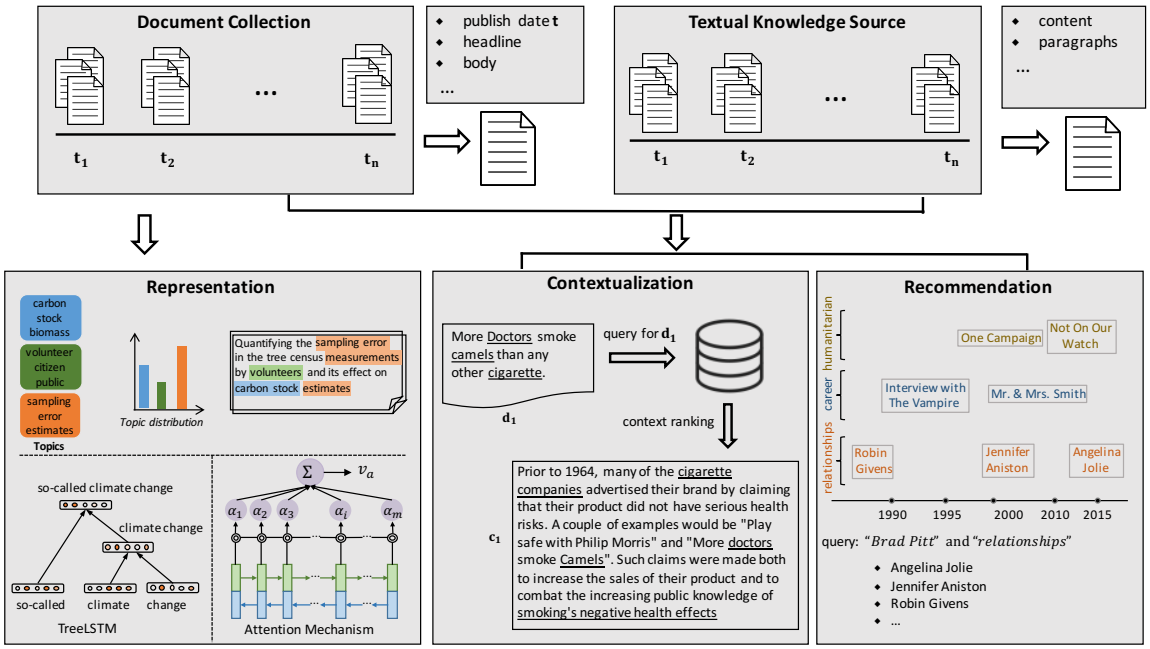



**Figure 1.1** Overview of the proposed approaches for supporting document understanding: Representation Learning, Re-Contextualization, and Entity Recommendation

**(I) Learning Representation for Document Understanding:** In the first part of this thesis, we propose approaches for improving document representations. In particular, we address the three research questions in problem **(I)**.

- **RQ1.1** Firstly, we propose a method to improve the probabilistic representation of documents where each document is represented by a mixture of topics learned by topic modeling algorithms. The topic modeling has gained a lot of popularity as a means of identifying and representing the topical structure of textual documents and whole corpora. There are, however, many document collections such as qualitative

studies in the digital humanities that cannot easily benefit from this technology. The limited size of those corpora leads to poor quality topic models. For solving this problem, we propose a fully automated adaptable process of topic cropping. For learning topics, this process automatically tailors a domain-specific Cropping corpus from a general corpus such as Wikipedia. The learned topic model is then mapped to the working corpus via topic inference. We analyze the learned topics with respect to coherence, diversity, and relevance, and show that they are of higher quality than those learned from the working corpus alone.

- **RQ1.2** Secondly, we propose multiplicative tree-structured Long Short-Term Memory networks to learn distributed vectors for document representations. The model is an extension of the TreeLSTM model [TSM15]. Unlike TreeLSTM, instead of using only word information, we also make use of relation information between words. Hence, the model is more expressive, as different combination functions can be applied for each word. Furthermore, in addition to syntactic trees, we investigate the use of Abstract Meaning Representation, a scheme for semantic knowledge representation, in tree-structured LSTM models, in order to incorporate both syntactic and semantic information for learning distributed representations.

- **RQ1.3** Finally, we present an approach to improve distributed representation learning with attention mechanism and investigate its usefulness in the application of question answering. More specifically, we propose Multihop Attention Networks (MAN) which aim to uncover the complex relations that can be observed between questions and answers for ranking question and answer pairs. Unlike previous models, we do not collapse the question into a single vector, instead we use multiple vectors which focus on different parts of the question for its overall semantic representation and apply multiple steps of attention to learn representations for the candidate answers. For each attention step, in addition to common attention mechanisms, we adopt sequential attention mechanism which utilizes context information for computing context-aware attention weights. We provide extensive experimental evidence of the effectiveness of our model on both factoid question answering and community-based question answering on different domains.

The contributions from this chapter are published in:

- Nam Khanh Tran, Sergej Zerr, Kerstin Bischoff, Claudia Niederée, Ralf Krestel. *Topic Cropping: Leveraging Latent Topics for the Analysis of Small Corpora.* In Proceedings of the International Conference on Theory and Practice of Digital Libraries, TPDL 2013, volume 8092 of Lecture Notes in Computer Science, pages 297-308. [TZB+13b]

- Nam Khanh Tran, Weiwei Cheng. *Multiplicative Tree-Structured Long Short-Term Memory Networks for Semantic Representations.* The Seventh Joint Conference on Lexical and Computational Semantics, *SEM 2018, pages 276–286. [TC18]

- Nam Khanh Tran, Claudia Niederée. *Multihop Attention Networks for Question Answer Matching.* The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2018, pages 325–334. [TN18b]

**(II) Bridging Temporal Context Gaps for Supporting Document Interpretations:** Fully understanding documents requires context knowledge from the time of document creation. Finding information about such context is a tedious and time-consuming task. In this case, just adding information, which is related to the entities and concepts mentioned in the text, as it is done in Wikification approaches, is not sufficient. The retrieved context information has to be time-aware, concise (not full Wikipedia pages) and focused on the coherence of the article topic. In the second part of this thesis, we first frame the novel problem of time-aware re-contextualization for supporting the interpretations of documents and then present an approach which takes those requirements into account in order to improve reading experience. For this purpose, we propose different query formulation methods for retrieving contextualization candidates and ranking methods taking into account topical and temporal relevance as well as complementarity with respect to the original document text.

The contributions in this chapter are published in:

- Nam Khanh Tran, Andrea Ceroni, Nattiya Kanhabua, Claudia Niederée. *Back to the Past: Supporting Interpretations of Forgotten Stories by Time-aware Re-Contextualization*. In Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, WSDM 2015, pages 339-348. [TCKN15a]

- Nam Khanh Tran, Andrea Ceroni, Nattiya Kanhabua, Claudia Niederée. *Time-travel Translator: Automatically Contextualizing News Articles*. In Proceedings of the 24th International Conference on World Wide Web, WWW 2015 Companion, pages 247-250. [TCKN15b]

- Andrea Ceroni, Nam Khanh Tran, Nattiya Kanhabua, Claudia Niederée. *Bridging Temporal Context Gaps Using Time-aware Re-contextualization*. In Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval, SIGIR 2014, pages 1127-1130. [CTKN14]

**(III) Dynamic Context-aware Entity Recommendation:** Entities and their relatedness are useful information in various tasks such as entity disambiguation, entity recommendation or exploratory search. In many cases, entity relatedness is highly affected by dynamic contexts, which can be reflected in the outcome of different applications. However, the role of context is largely unexplored in existing entity relatedness measures. In the last part of this thesis, we introduce the notion of contextual entity relatedness, and show its usefulness in the new yet important problem of context-aware entity recommendation. We propose a novel method of computing the contextual relatedness with integrated time and topic models. By exploiting an entity graph and enriching it with an entity embedding method, we show that our proposed relatedness can effectively recommend entities, taking contexts into account.

The contribution in this chapter has been published in:

- Nam Khanh Tran, Tuan Tran, Claudia Niederée. *Beyond Time: Dynamic Context-Aware Entity Recommendation*. The Semantic Web - 14th International Conference, ESWC 2017, pages 353-368. [TTN17] (**Nomination for best paper award**)

## 1.4 Thesis Structure

We organize the remainder of this thesis as follows. In Chapter 2, we discuss selected general background techniques and algorithms that build a basis to achieve the goals conducted in this thesis. In particular, we focus on selected techniques from the areas of Machine Learning, Natural Language Processing and Information Retrieval. Following that, in Chapter 3, we discuss the problem of learning representations of documents by exploiting document content and structure. We first study the probabilistic representation with topic modeling and then the distributed vector representation using neural network models. In addition, we illustrate the usefulness of representation learning with attention mechanism in the application of question answering. In Chapter 4, we introduce the task of time-aware contextualization and describe a novel approach to bridge temporal context gaps for supporting interpretations of documents. Subsequently, in Chapter 5, we introduce the notion of contextual entity relatedness and present a probabilistic approach to tackle the problem of dynamic context-aware related entity recommendation. Finally, we discuss the contributions of this thesis again and point out directions for future research in Chapter 6.

To aid readers of this thesis, each chapter has been written to serve as a self-contained reflection that highlights the challenges being tackled in the chapter, the related literature in that context, the proposed approach, experimental setup and methodology, our consequent findings and their implications.

# 2 Foundations and Technical Background

In this chapter, we discuss the technical background necessary to understand the work carried in this thesis. In particular, we first introduce the notion of word representation which then serves as a basic unit for learning document representation. Next, we provide a thorough analysis of information retrieval techniques. Finally, we describe machine learning algorithms, with a special focus on topic modeling and recurrent neural networks.

## 2.1 Semantic Representations

### 2.1.1 Word Representations

Words are typically the smallest units of representation, which can then be used to derive representations for larger units of information such as passages and documents. In a basic (*local*) representation, every word in a fixed size vocabulary $V$ is represented by a binary vector $v \in \{0,1\}^V$, where only one of the values in the vector is one and all the others are set to zero. Latent feature representations are another choice for the word representations, which have been widely used in many tasks in recent years [Man15, Got16]. Many methods have been proposed for learning such real-valued latent feature word vectors [MCCD13, Gol16]. The general hypothesis behind those methods is that words which occur in similar contexts share semantic relatedness or similarity [Har54]. Traditional count-based methods typically rely on word co-occurrence counts in a context window, e.g., methods, which are based on Pointwise Mutual Information or matrix factorization, use context windows of 5 or 10 words [TP10]. Recent prediction-based models maximize the probability of predicting contexts where a target word occurs, or vice versa, predicting the target word given its contexts [MCCD13, MSC+13].

In the following, we describe two recent widely used models for learning word vector representation. We utilize the pretrained word vectors produced by these models in Chapter 3 and Chapter 5.

**Word2Vec Skip-gram model.** Given a sequence of training words $D = \{w_1, w_2, ..., w_T\}$, the *Word2Vec* skip-gram model [MSC+13] minimizes the following negative log-likelihood objective function:

$$\mathcal{L} = -\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c; j\neq 0} \log p(w_{t+j}|w_t) \tag{2.1}$$

where $w_{t+j}$ is a context word given the target word $w_t$, with $c$ to be the context size. The basic skip-gram formulation defines $p(w_{t+j}|w_t)$ using the *softmax* function as follows:

$$p(w_O|w_I) = \frac{\exp\left({v'_{w_O}}^{\top} v_{w_I}\right)}{\sum_{i=1}^{V}\exp\left({v'_{w_i}}^{\top} v_{w_I}\right)} \tag{2.2}$$

where $v_w$ and $v'_w$ are the input and output vector representations of $w$, and $V$ is the number of words in the vocabulary $W$.

Computing $\log p(w_O|w_I)$ is expensive for each training target word, hence the Word2Vec skip-gram model approximates $\log p(w_O|w_I)$ with a negative-sampling objective:

$$\mathcal{O} = \log\sigma\left({v'_{w_O}}^{\top} v_{w_I}\right) + \sum_{i=1}^{k}\mathbb{E}_{w_i\sim P_n(w)}\left[\log\sigma\left(-{v'_{w_i}}^{\top} v_{w_I}\right)\right] \tag{2.3}$$

where $\sigma$ is the *sigmoid* function: $\sigma(x) = \frac{1}{1+e^{-x}}$ and words $w_i$ are randomly sampled from the vocabulary $W$ using a noise distribution $P_n(w)$, where there are $k$ negative samples for each data sample. The model is then trained to learn word vectors using vanilla stochastic gradient descent (SGD).

*In Chapter 5, we utilize the Word2Vec skip-gram model to learn entity and word vectors simultaneously and use these vector representations for the task of entity recommendation.*

**GloVe model.** The GloVe model [PSM14] is another widely used model for learning word vectors, by combining advantages of both count-based and prediction-based methods. Let $X$ be the word context co-occurrence matrix where $X_{ij}$ denotes the number of times the $i^{th}$ word type occurs near the $j^{th}$ word type in a corpus. The GloVe model learns word vectors from $X$ by minimizing the following objective function:

$$\mathcal{L} = \sum_{i,j=1}^{V} f(X_{ij})\left(w_i^T\tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2 \tag{2.4}$$

where $V$ is the size of the vocabulary, $b_i$ and $\tilde{b}_j$ are unknown target and context bias terms associated with the $i^{th}$ and $j^{th}$ word types, respectively. In addition, $f(X_{ij})$ is defined as the weighting function:

$$f(X_{ij}) = \begin{cases} \left(\frac{X_{ij}}{100}\right)^{3/4} & \text{if } X_{ij} < 100 \\ 1 & \text{otherwise} \end{cases}$$

The GloVe model is trained to learn word vectors using SGD with AdaGrad adaptive learning [DHS11].

*In Chapter 3, we make use of pretrained GloVe word vectors to derive the representations of documents, which are then used in various downstream applications.*

### 2.1.2 Document Representations

The most currently used method of document representation is Vector Space Model (VSM). In VSM, a document is represented by the terms occurring in the document, and for each term we can assign boolean indicator values or some forms of weight reflecting the importance in the document. The most widely used weighting scheme is based on the *tf-idf* [MRS08]. That is, the *term frequency* or $tf$ measures the frequency of a term $v \in W$ in a document $d \in D$, whereas the $idf$ or *inverse document frequency* counts the number of documents in which the term $v$ occurs. While $tf$ indicates the importance of term for a document, $idf$ measures how well such a term distinguishes a document from others. Their combination yields the trade-off between the two, and its simplest variation is defined by:

$$tfidf = tf(v,d) \cdot \underbrace{\frac{|D|}{df(v)}}_{\text{idf}} \tag{2.5}$$

where, $df(v) = |d \in D : v \in d|$, representing the number of documents in $D$ containing term $v$.

However, such representation neglects potential semantic links between words. To take them into account, several more recent models have been proposed in the literature, mostly based on a probabilistic approach. The n-grams statistical language models [MRS08] were proposed to capture term correlation within document. However, the exponentially increasing data dimension with the increase of $n$ limits the application of n-gram models.

The probabilistic topic models were also proposed for representing the semantics of text documents. They in general factor the joint or conditional probability of words and documents by assuming that the choice of a word during the generation of a document is independent of the document given some hidden variable, often called *topic* or *aspect*. Probabilistic Latent Semantic Indexing (PLSI) and Latent Dirichlet allocation (LDA) are the two well known topic modeling methods (see Section 2.3.2 for more details).

## 2.2 Information Retrieval

Information Retrieval (IR) deals with the means on accessing and satisfying user information needs through querying of large collections, mostly of unstructured documents. Though its foundations being on unstructured documents, IR has become a multi-modal field, providing techniques for access of multimedia objects. In addition, recent IR ap-

proaches have considered not only textual information but also taken into account other aspects such as temporal and geographical dimensions.

In this thesis, we mainly discuss relevant *query models*, which can be formally defined as follows:

*For a document collection $D$ which is projected into a vocabulary space of terms $V$, and a query $q \in V$, the task is to find relevant documents from $D$ such that they satisfy the information need in $q$.*

In the following sections, we first present two traditional IR models, i.e. Okapi BM25 and query-likelihood language model, and then describe several temporal IR models which take temporal dimension into consideration.

### 2.2.1 Traditional IR Models

**Okapi BM25.** One of the most widely used retrieval models is BM25 [RWJ+95]. In contrast to the *tfidf* model, which is based purely on the *tfidf* scores, BM25 requires parameter tuning that are dependent on the given document collection $D$. Furthermore, the document length is taken into account in the query-document scoring function. In particular, the BM25 scoring model is computed as follows:

$$r(q,d) = \sum_{v \in q} w_{tf}(v,d) \cdot w_{idf}(v) \tag{2.6}$$

where the term frequency score $w_{tf}(v,d)$ is defined by:

$$w_{tf}(v,d) = \frac{(k_1+1) \cdot tf(v,d)}{k_1 \cdot \left((1-b) + b \cdot \frac{|d|}{avgdl}\right) + tf(v,d)} \tag{2.7}$$

where the parameters $k_1$ ($k_1 \geq 1$) and $b$ ($0 \leq b \leq 1$) are tunable, and are usually set to values $k_1 = 1.2$ and $b = 0.75$, respectively. $|d|$ is the length of document $d$ in words whereas $avgdl$ stands for the average document length in $D$. Here, $b$ controls how much we normalize the term frequency scores according to the document length and its ratio to the average document length in $D$.

The inverse document frequency score $w_{idf}$ for a query term is computed as follows:

$$w_{idf}(v) = \log \frac{N - df(v) + 0.5}{df(v) + 0.5} \tag{2.8}$$

where $N$ represents the number of documents in $D$, and $df(v)$ represents the number of documents containing term $v$.

**Query-likelihood Language Model.** The query-likelihood language modeling approach was first introduced by Ponte and Croft [PC98]. The basic idea behind the approach is simple: first estimate a language model for each document, and then rank documents by

the likelihood of the query according to the estimated language model of each document. Formally, the goal is to rank documents by $P(d|q)$, where the probability of a document is interpreted as the likelihood that it is relevant to the query. Using Bayes rule, $P(d|q)$ is estimated as follows:

$$P(d|q) = \frac{P(q|d)P(d)}{P(q)} \tag{2.9}$$

$P(q)$ is the same for all documents, and so can be ignored. The prior probability of a document $P(d)$ is often treated as uniform across all $d$, and thus it can also be ignored, but we could implement a genuine prior which could include criteria like authority, length, genre and freshness. Given these simplifications, we return results ranked by simply $P(q|d)$, the probability of the query $q$ under the language model derived from $d$. With the assumption that query terms are independent, $P(q|d)$ can be estimated as:

$$P(q|d) = \prod_{w \in q} P(w|d)^{n(w,q)} \tag{2.10}$$

where $w$ is a query term in $q$, $n(w, q)$ is the term frequency of $w$ in $q$, and $P(w|d)$ is the probability of $w$ estimated using Dirichlet smoothing as follows:

$$P(w|d) = \frac{n(w, d) + \mu P(w)}{\mu + \sum_{w'} n(w', d)} \tag{2.11}$$

where $n(w, d)$ is the term frequency of $w$ in $d$, $\mu$ is the smoothing parameter and $P(w)$ is the probability of term $w$ in the collection.

### 2.2.2 Temporal IR Models

In practice, many information needs have a temporal dimension which is expressed by temporal phrases mentioned in the users' queries. To handle such temporal information needs, several temporal retrieval models have been proposed [BBAW10, KN10].

Formally, let $q_{text}$ and $q_{time}$ denote keywords and temporal expressions of a temporal query $q$. Let $d_{text}$ and $d_{time}$ be textual parts and temporal parts of a document $d$. In [KN10], Kanhabua et al. proposed a mixture model to combine textual similarity and temporal similarity for ranking time-sensitive queries, in which the similarity between question $q$ and document $d$ is defined by:

$$S(q, d) = (1 - \alpha) \cdot S'(q_{text}, d_{text}) + \alpha \cdot S''(q_{time}, d_{time}) \tag{2.12}$$

where $1 - \alpha$ and $\alpha$ indicates the importance of textual similarity and temporal similarity, respectively.

In [BBAW10], Berberich et al. proposed an alternative approach to combine these textual and temporal similarities:

$$S(q, d) = S'(q_{text}, d_{text}) \cdot S''(q_{time}, d_{time}) \tag{2.13}$$

In both Equation 2.12 and Equation 2.13, while $S'(q_{text}, d_{text})$ can be measured using any of existing text-based weighting functions, $S''(q_{time}, d_{time})$ is computed by assuming that a temporal expression $t_q \in q_{time}$ is generated independently from each other.

$$\begin{aligned} S''(q_{time}, d_{time}) &= \prod_{t_q \in q_{time}} P(t_q|d_{time}) \\ &= \prod_{t_q \in q_{time}} \left( \frac{1}{|d_{time}|} \sum_{t_d \in d_{time}} P(t_q|t_d) \right) \end{aligned} \tag{2.14}$$

In Equation 2.14, Jelinek-Mercer smoothing is applied to avoid the zero-probability problem and $P(t_q|t_d)$ can be estimated using different temporal ranking methods namely LMT and LMTU [BBAW10], TS and TSU [KN10].

*In Chapter 4, we demonstrate the usefulness of temporal IR models in the task of time-aware re-contextualization, where time is an important dimension.*

## 2.3 Machine Learning

In this section, we describe machine learning algorithms which are used in the thesis. We first introduce some *supervised learning* algorithms, and then concentrate on *probabilistic topic models*. Following that, we discuss *neural network* models, with a special focus on recurrent neural networks and attention mechanism.

### 2.3.1 Supervised Learning

In supervised learning, given a training dataset of inputs $X$ and outputs $Y$, the task is to learn an association function $f : X \rightarrow Y$ mapping each input $x \in X$ to an output $y \in Y$. The outputs $Y$ can be collected automatically but in some cases $Y$ must be provided by a human supervisor. In the following paragraphs, we briefly describe Logistic Regression and Support Vector Machines algorithms.

**Logistic Regression - LR.** It is one of the most simplistic and widely used supervised learning algorithms [Bis06]. For a set of training examples $X = \{x^1, x^2, ..., x^k\}$ where each item $x^i$ is a $n$-dimensional feature vector, LR estimates the probability distribution $P(Y = y^i|x^i)$ by using maximum likelihood estimation to find the best parameter vector $\theta$ for a parametric family of distributions $P(y|x;\theta)$. If we have two classes, class 0 and class 1, we can use the logistic sigmoid function to squash the output of the linear function into the interval (0, 1) and interpret that value as a probability:

$$P(y = 1|x;\theta) = \sigma(\theta^\top x) \tag{2.15}$$

While logistic regression has found wide adaptation for many classification tasks, one main disadvantage is its linearity, that is, it can classify accurately instances that are only linearly separable.

**Support Vector Machines.** SVMs [CV95] are widely used supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis, especially in a high- or infinite-dimensional space. The basic idea is to find a *hyperplane* which linearly separates the $d$-dimensional data. An optimal hyperplane is constructed based on so called *support vectors*, which determines the maximal margin between support vectors of different classes. Figure 2.1 shows an example of support vectors for an optimal hyperplane.

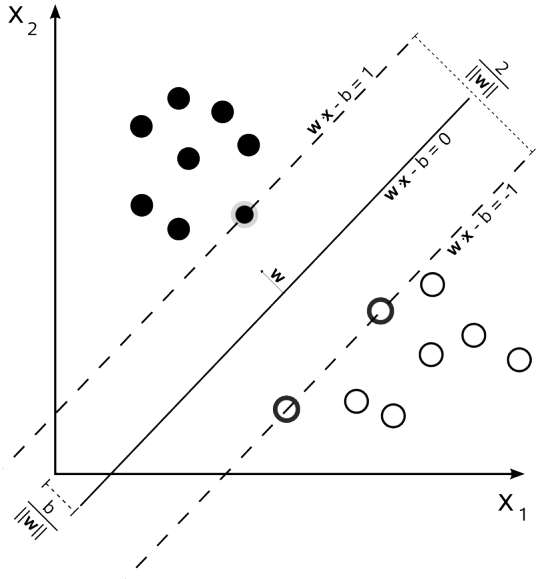



**Figure 2.1** Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. The support vectors are the ones which are on the margin.

The model is similar to logistic regression in that it is driven by a linear function $\theta^\top x+b$. Unlike logistic regression, the SVM does not provide probabilities, but only outputs a class identity. The SVM predicts that the positive class is present when $\theta^\top x + b$ is positive. Likewise, it predicts that the negative class is present when $\theta^\top x + b$ is negative. The optimal weights are estimated subject to the support vectors and are discussed in details in [CV95, Bis06]. *In Chapter 4, we make use of SVMs for approaching the problem of query performance prediction.*

### 2.3.2 Probabilistic Topic Models

Topic models [Hof99, BNJ03, GS04] are based upon the idea that documents are mixtures of topics, where a topic is a probability distribution over words. A topic model is a generative model for documents that specifies a simple probabilistic procedure by which document can be generated. Let $P(z)$ or $\theta^{(d)}$ denote the distribution over topics $z$ in a particular document $d$ and $P(w|z)$ denote the probability distribution over words $w$ given

topic $z$. Each word $w_i$ in a document is generated by first sampling a topic from the topic distribution, then choosing a word from the topic-word distribution. Let $P(z_i = j)$ be the probability that the $j$th topic was sampled for the $i$th word and $P(w_i|z_i = j)$ as the probability of word $w_i$ under topic $j$. The model specifies the following distribution over words within a document.

$$P(w_i) = \sum_{j=1}^{T} P(w_i|z_i = j)P(z_i = j) \tag{2.16}$$

where $T$ is the number of topics.

Hofmann [Hof99] introduced Probabilistic Latent Semantic Indexing method (pLSI) to document modeling. The pLSI model does not make any assumptions about how the mixture weights $\theta$ are generated, making it difficult to test the generalizability of the model to new documents. Blei et al. [BNJ03] extended this model by introducing a Dirichlet prior $\alpha$ on $\theta$, calling the resulting generative model Latent Dirichlet Allocation (LDA). As a conjugate prior for the multinomial, the Dirichlet distribution is a convenient choice as prior, simplifying the problem of statistical inference.

Griffiths and Steyvers [GS04] explored a variant of this model, discussed by Blei et al. [BNJ03], by placing a symmetric Dirichlet ($\beta$) prior on $P(w|z)$ as shown in Figure 2.2. The hyperparameter $\beta$ can be interpreted as the prior observation count on the number of times words are sampled from a topic before any word from the corpus is observed. This smoothes the word distribution in every topic, with the amount of smoothing determined by $\beta$ . Good choices for the hyperparameters $\alpha$ and $\beta$ will depend on number of topics and vocabulary size. Previous studies showed that $\alpha = 50/T$ and $\beta = 0.01$ often work well with many different text collections.

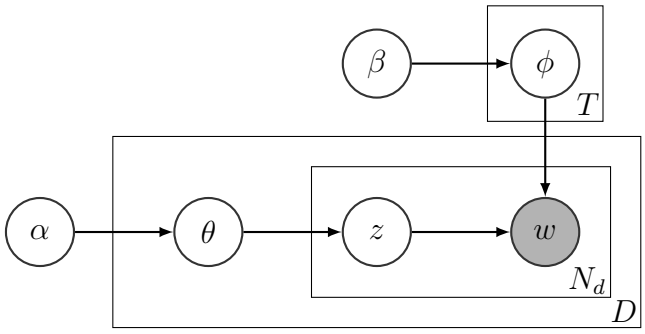



**Figure 2.2** The graphical model for topic model using plate notation.

Since topic modeling was developed in the context of large document collections such as scientific articles and news collections, it has obtained poor results in terms of topic coherence and diversity with small corpora [CL14].

*In Chapter 3, we discuss how to improve the quality of learned topics when applying topic modeling algorithms to small document collections through Topic Cropping.*

### 2.3.3 Neural Network Models

Neural network models consist of chains of tensor operations. The tensor operations can range from parameterized linear transformations (e.g., multiplication with a weight matrix, addition of a bias vector) to element-wise application of non-linear functions such as *tanh* or *rectified linear units* (ReLU). For example, given an input vector $x$, a simple feed-forward neural network with fully-connected layers produces the output $y$ as follows:

$$y = \tanh\left(W_2 \tanh\left(W_1\, x + b_1\right) + b_2\right) \tag{2.17}$$

The model training involves tuning the parameters $W_1, b_1, W_2$ and $b_2$ to minimize an expected loss.

Recently, people get more interested in neural networks with a lot of layers, i.e. *deep architectures* or *deep learning*, in which convolutional [KSH17, LKF10] and recurrent [Elm90, HS97, MKB+10] architectures are commonplace in most deep learning applications. In the scope of this thesis, we focus more on recurrent neural networks.

**Recurrent Neural Networks (RNNs).** RNNs [Elm90] are a family of neural networks for processing sequential data. RNNs are called *recurrent* because they perform the same task for every element of a sequence, with the output being depended on the previous computations. In theory RNNs can make use of information in arbitrarily long sequences, but in practice they are limited to looking back only a few steps.

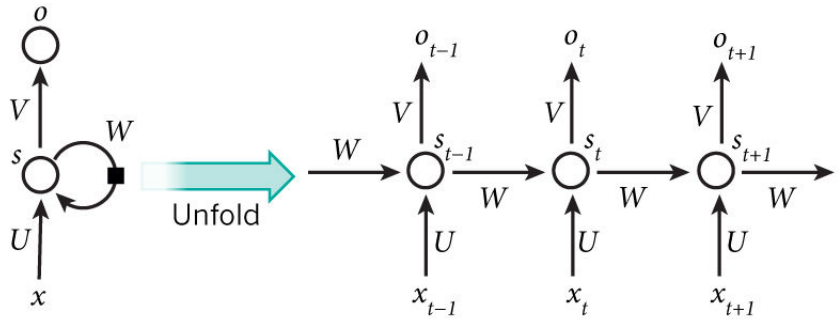



**Figure 2.3** A recurrent neural network and the unfolding in time of the computation involved in its forward computation. [LBH15]

Figure 2.3 shows a RNN being unrolled (or unfolded) into a full network. Given a sequence $x = (x_1, x_2, ..., x_T)$, the RNN updates its current hidden state $s_t$ by:

$$s_t = \begin{cases} 0 & t = 0 \\ \phi(s_{t-1}, x_t) & \text{otherwise} \end{cases} \tag{2.18}$$

where $\phi$ is a nonlinear function such as composition of a logistic sigmoid with an affine transformation. Traditionally, the update of the recurrent hidden state in Equation 2.18 is

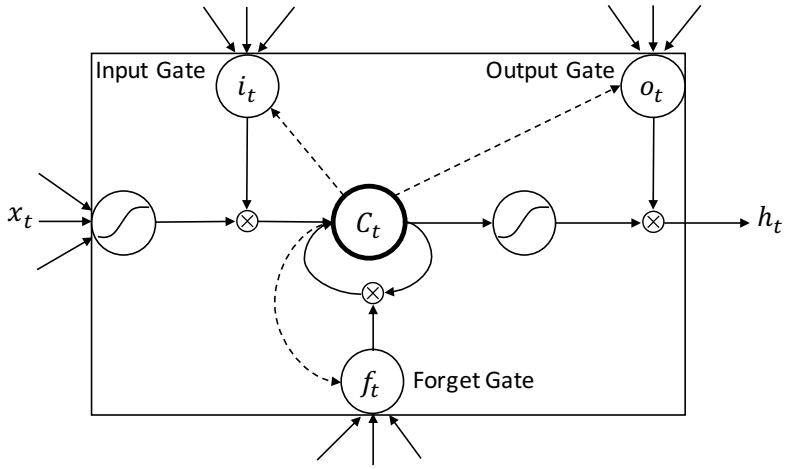



**Figure 2.4** LSTM memory block with one cell. [Gra12]

implemented as:

$$s_t = g(Ux_t + Ws_{t-1}) \tag{2.19}$$

where $g$ is a smooth, bounded function such as a logistic sigmoid function or a hyperbolic tangent function. In addition, the output $o_t$ at step $t$, for example if we want to predict the next word in a sentence, would be a vector of probabilities across our vocabulary, i.e. $o_t = \text{softmax}(Vs_t)$.

In practice, the range of context that can be assessed in standard RNN architectures is quite limited. This issue is often referred to as the *vanishing gradient problem* [HS97]. Long Short Term Memory networks, or LSTMs which are a special kind of RNN have been shown effectively in handling this problem. LSTMs were introduced by Hochreiter and Schmidhuber [HS97], and were refined and popularized by many people in the following work. LSTMs are capable of learning long-term dependencies and work tremendously well on a large variety of problems.

Figure 2.4 provides an illustration of an LSTM memory block with a single cell. An LSTM network is the same as a standard RNN, except that the summation units in the hidden layer are replaced by memory blocks. The same output layers can be used for LSTM networks as for standard RNNs. The multiplicative gates allow LSTM memory cells to store and access information over long periods of time, thereby mitigating the vanishing gradient problem.

The first step in LSTM is to decide what information is going to be thrown away from the cell state. This decision is made by a sigmoid layer called the *forget gate* layer. It looks at $h_{t-1}$ and $x_t$, and outputs a number between 0 and 1 for each number in the cell state $C_{t-1}$

$$f_t = \sigma(W_f h_{t-1} + U_f x_t + b_f) \tag{2.20}$$

The next step is to decide what new information is going to be stored in the cell state. A sigmoid layer called the *input gate* layer decides which values will be updated. A tanh

layer creates a vector of new candidate values $\tilde{C}_t$, that could be added to the state.

$$\begin{aligned} i_t &= \sigma(W_i h_{t-1} + U_i x_t + b_i) \\ \tilde{C}_t &= \tanh(W_C h_{t-1} + U_C x_t + b_C) \end{aligned} \tag{2.21}$$

These two values are combined to create an update for the state. The old state $C_{t-1}$ is multiplied by $f_t$, forgetting the things which are decided to forget earlier and then add $i_t \cdot \tilde{C}_t$.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \tag{2.22}$$

Finally, the output will be based on the cell state, but will be a filtered version. A sigmoid layer is first used to decide which parts of the cell state are going to output. Then, the cell state is put through tanh and multiply it by the output of the sigmoid gate.

$$\begin{aligned} o_t &= \sigma(W_o h_{t-1} + U_o x_t + b_o) \\ h_t &= o_t \cdot \tanh(C_t) \end{aligned} \tag{2.23}$$

Recently, tree-structured LSTMs [TSM15, ZSG15], TreeLSTMs for short, have been studied to extend the standard LSTM by exploiting syntactic information. The key idea of TreeLSTMs is to extend the LSTM structure from linear chains to trees. While the conventional LSTM forms its hidden state from the current input and the previous hidden state, TreeLSTM forms it with an input and the hidden states of arbitrarily many child units. It therefore includes the conventional LSTM as a special case and is not limited to sequential information propagation. Such extensions outperform competitive LSTM baselines on several tasks such as sentiment classification and semantic relatedness prediction [TSM15]. Li et al. [LLJH15] further investigated the effectiveness of TreeLSTMs on various tasks and discussed when tree structures are necessary.

*In Chapter 3, we propose multiplicative tree-structured LSTMs which further extend the TreeLSTM models by incorporating richer linguistic information.*

**Attention Mechanism.** Neural processes involving attention have been largely studied in Neuroscience and Computational Neuroscience [IKN98, DD95]. A particularly studied aspect is visual attention: many animals focus on specific parts of their visual inputs to compute the adequate responses. This principle has a large impact on neural computation as we need to select the most relevant piece of information, rather than using all available information, a large part of it being irrelevant to compute the neural response. A similar idea - focusing on specific parts of the input - has been applied in different tasks such as speech recognition, machine translation, and visual identification of objects.

In principle, an attention model is a method that takes $n$ arguments $\{y_1, ..., y_n\}$ and a context $c$. It returns a vector $z$ which is supposed to be the summary of the $y_i$, focusing on information linked to the context $c$. More specifically, it returns a weighted arithmetic mean of the $y_i$, and the weights are chosen according the relevance of each $y_i$ given the context $c$ as shown in Figure 2.5. One interesting feature of attention model is that the weights of the arithmetic means are accessible and can be plotted.

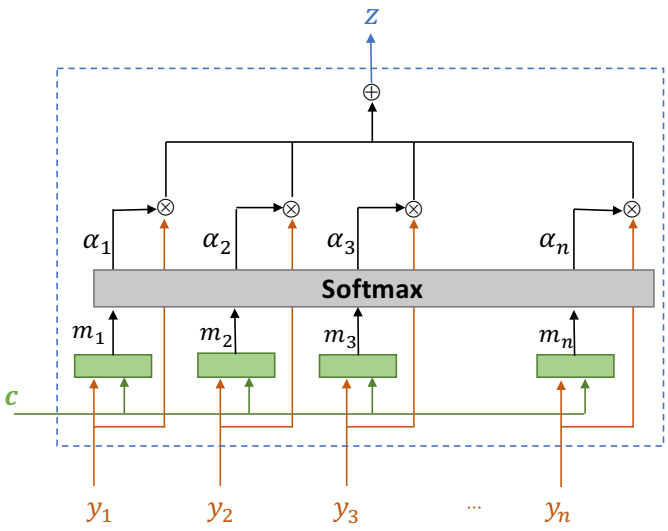



**Figure 2.5** Attention Mechanism.

Additive attention [BCB15] and multiplicative attention [RCW15] are the two most commonly used attention mechanisms. The additive attention mechanism uses a multi-layer perceptron network with *tanh* activation to compute attention weights as follows:

$$\begin{aligned} m_i &= \tanh\left(W_c y_i + U_c c\right) \\ \alpha_i &= \text{softmax}\left(W_m^T m_i\right) \\ z &= \sum_i \alpha_i y_i \end{aligned} \tag{2.24}$$

where $W_c, U_c$ and $W_m$ are attention parameters.

The multiplicative attention mechanism makes use of a billinear term instead of tanh layer for the weight estimations:

$$\begin{aligned} \alpha_i &= \text{softmax}_i c^T W_c y_i \\ z &= \sum_i \alpha_i y_i \end{aligned} \tag{2.25}$$

where $W_c$ is used in a bilinear term which allows us to compute a similarity between $c$ and $y_i$ more flexibly than with just a dot product. Some other attention mechanisms can be found in [LFdS+17, BYB17, XZS16].

*In Chapter 3, we investigate the usefulness of representation learning with attention mechanism in the application of question answering.*

# 3 Learning Representation for Document Understanding

In this chapter, we study the problem of learning document representation, which is becoming an important step in many document understanding tasks. Firstly, we describe a topic cropping approach which aims to improve the quality of learned topics using Latent Dirichlet Allocation on small collections. In this case, documents are represented by a mixture of topics where each topic is a distribution over words. Following that, we describe a neural network based approach aiming to learn the continuous low-dimensional vector representation for documents. Finally, we illustrate the usefulness of representation learning by focusing on the problem of question answering.

## 3.1 Introduction

As briefly discussed in the introductory chapter, most popular document representation methods have relied on the bag-of-words based approaches [MS99], through which a document is fundamentally represented by counts of word occurrences within the document. Yet, this approach can be problematic when a number of documents being represented are enormous. As the number of documents increase, a number of words in vocabulary will also increase. Consequently, not only will the generated document vectors be sparse, but also their dimensions will be huge. Though various dimension reduction techniques [DDF+90] do exist, these techniques lose the innate interpretability of the bag-of-words approach. Moreover, such representation neglects potential semantic links between words. To overcome such limitations of the bag-of-words approach, many models have been proposed in recent years, including the Probabilistic Latent Semantic Analysis (PLSA) [Hof99] or Latent Dirichlet allocation (LDA) [BNJ03] and distributed representation learning approaches [LM14].

The PLSA or LDA, factors the joint or conditional probability of words and documents by assuming that the choice of a word during the generation of a document is independent

of the document given some hidden variables called *topics* or *aspects*. Documents are then represented by a mixture of topics where each topic is a distribution over words. The topic-based representation has outperformed the bag-of-words approaches in many tasks [BNJ03, GS04]. However, a key weakness of topic modeling is that it needs a large amount of data (e.g, thousands of documents) to provide reliable statistics to generate coherent topics. In practice, many document collections such as qualitative studies do not have so many documents. Given a small number of documents, the classic topic model LDA generates very poor topics [CL14]. In the first part of this chapter, we fill this gap by proposing a topic cropping approach which automatically tailors a bigger related dataset to generate more coherent topics for a given document collection.

Learning the distributed representation for long spans of text (e.g, passages, documents) has recently obtained significant popularity [LM14, TSM15, KGB14]. The basic idea is to utilize contextual information of each word and document to embed document vectors with a manageable dimension into a continuous vector space. This has become an important step in various NLP tasks such as text classification, semantic matching and machine translation. Seminal work is based on recurrent neural networks (RNN) [Elm90], convolutional neural networks [KGB14], and tree-structured neural networks [SLNM11, TSM15]. Previous approaches, however, often ignore the linguistic knowledge such as syntactic information of textual documents. To address this issue, in the second part of this chapter, we present multiplicative tree-structured Long Short-Term Memory networks, which are able to integrate the linguistic knowledge into neural network models for enhancing the distributed semantic representation of documents, and show their effectiveness in various applications.

Though LSTM or CNN based models outperform other representation learning approaches (e.g. LDA), they still suffer from an important issue. They are limited on the length of input sequences that can be reasonably learned and results in worse performance for very long input sequences [TdSXZ16]. Therefore, in the last part of this chapter we seek to overcome this limitation with the help of attention mechanism. The basic idea of attention mechanism is that given an input sequence the segments with a stronger focus are treated more important and have more influence on the resulting representation. We illustrate the usefulness of representation learning with attention mechanism in the task of question answer selection.

## 3.2 Leveraging Latent Topics for the Analysis of Small Corpora

In this section, we study the problem of improving the quality of topics when applying topic modeling algorithms to small collections of documents such as qualitative studies. For social sciences, sharing qualitative primary data like interviews and re-using it for secondary analysis is very promising as data collection is very time consuming. Moreover, some qualitative data sources capture valuable information about attitudes, beliefs and so on, as people had them at other times – "realities" that cannot be captured anymore. En-

abling secondary analysis of data not collected by oneself, analyzing it with new research questions in mind, imposes a lot of challenges though. Here, we focus on the aspect of advanced techniques for facilitating exploration of such data and for improving findability in digital data archives.

By exploiting information retrieval and topic modeling techniques we can mine additional knowledge about themes discussed in primary qualitative data. This way, interview contents can be visualized by means of extracted topics to give a quick overview. For example, topics extracted from a collection of studies, or samples show the commonalities of themes while comparing topics of individual studies, or samples sheds light on the specifics. Interview topics as well aid an enhanced (automatic) content analysis and retrieval of similar documents. This is especially interesting as qualitative documents are often long, and thus it is hard to grasp their thematic coverage – let alone to manually analyze them.

Due to the enormous resources required for conducting qualitative research by means of interviews (holding the interview, transcription, document coding/analysis), the primary data resulting from such qualitative studies is usually limited to a small number of around 20 to 50, rarely around 100 interviews per study case or sample. Topic models, however, are based on statistics and thus perform better on big data sets [NBB11].

In this work, we present a generalizable framework for using topic modeling given such corpora restrictions as they occur in qualitative social science research. Our fully automated adaptable process tailors a domain-specific Cropping corpus by collecting relevant documents from a general corpus or knowledge base, here Wikipedia. The topic model learned on this substitute corpus is then applied to the original collection. Hence, we exploit state-of-the-art IT-methods adapting and integrating them for usage as research tools for the digital humanities. In detail, the contributions of this work are presented as follows:

- We propose a process for *topic cropping* and proof its improved performance for small corpora by analyzing diversity, coherence, and relevance.
- By integrating the automatic evaluation of topic quality we take a first step towards a self-optimizing process of selecting parameters for topic cropping in different settings.

### 3.2.1 Related Literature

Topic modeling is a generative process that introduces latent variables to explain co-occurrence of data points. Latent Dirichlet allocation (LDA) [BNJ03] is a further development of probabilistic latent semantic analysis (PLSA) [Hof99]. LDA was developed in the context of large document collections, such as scientific articles, news collections, etc. The success of LDA led to the application in other domains, such as image processing, as well as other types of documents, e.g. tweets [HD10] or tags [KFN09]. Some work applies topic modeling to transcribed text. In [PGKT06], the standard LDA model is extended to identify

not only topics but also topic boundaries within longer meeting transcripts. The authors show that topic modeling can be used to detect segments in heterogeneous text. Howes et al. [HPM13] investigate the use of topic models for therapy dialog analysis. More specifically, LDA is applied to 138 transcribed therapy sessions to then predict patient symptoms, satisfaction, and future adherence to treatment using latent topics detected vs. hand coded topics. The authors find only the manually assigned topics to be indicative. Human assessment of the interpretability of the automatically learned topics showed high variance of topic coherence.

Using topic models where there is only limited data, e.g., very short documents or very few documents, has been studied as well. Micro-blogging services, such as Twitter, limit single documents to 140 tokens. Hong and Davison [HD10] study different ways to overcome this limitation when training topic models by aggregating these short messages based on users or terms. The resulting longer documents yield better topic models compared to training on short, individual messages. Unfortunately, this method only works if the number of short texts is sufficiently large. Using additional long documents to improve topics used for classification was proposed in various approaches: Learning a topic model from long texts and then applying it to short text [PNH08] improves significantly over learning and applying it on short texts only. Learning it on both [XDYY08] and applying it on short texts improves performance further. Jin et al. [JLZ+11] present their Dual LDA model to model short texts and additional long text explicitly, which outperforms standard LDA on long and short texts for classification. Our focus is not on classification of short documents but we use topic modeling to analyze (long) individual documents and focus more on a careful selection of the corresponding training corpus. Incorporating domain knowledge for topic transition detection using LDA as described in [ZHMP08] addresses this problem using manual selection of training corpora. A topic model is trained using auxiliary textbook chapters and is used to compare slide content and transcripts of lectures. Because of sparse text on slides and possible speech recognition errors in the transcripts training a topic model on long, related documents improves alignment of slides and transcript significantly. In contrast, our method does not rely on a manual selection of a training set as cropping is performed as an automated process.

### 3.2.2 A General Approach for Topic Cropping

The goal of our approach is to enable the exploitation of the advantages of topic models, e.g., with respect to capturing latent semantics, even if the considered corpus is too small for their direct application. Smaller corpora such as qualitative studies in the humanities result in topic models of restricted quality. The approach we are following in this work is to use another larger corpus (the Cropping corpus) for learning the topic model. Subsequently, the learned topic model is applied to the study under consideration via topic inference. Qualitative studies are often very focused, which makes finding a good Cropping corpus a difficult task. Since we are looking for an approach, which is applicable in different settings (i.e., for studies in different application domains), there are two requirements to

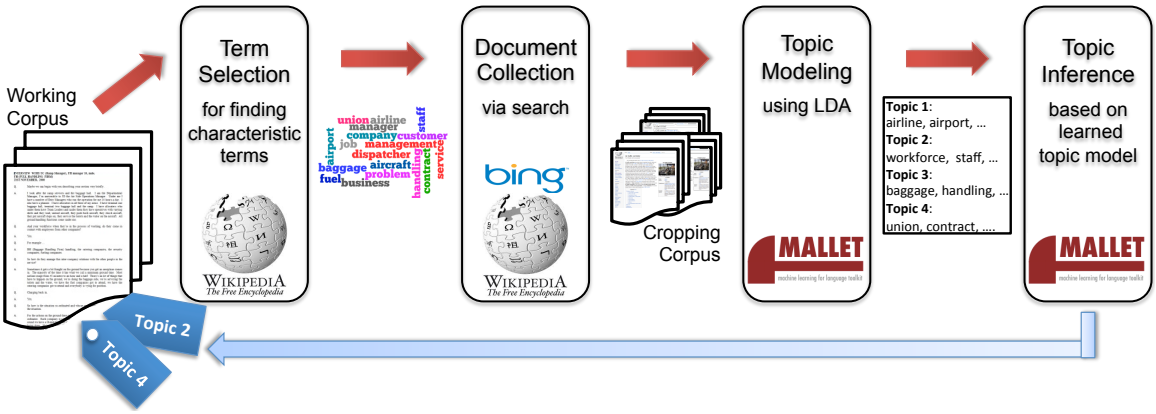



**Figure 3.1** Workflow for Topic Modeling on a Cropping corpus

be satisfied: (1) having a Cropping corpus that is specific enough to produce a good and useful coverage of the topics in the study under consideration (2) while avoiding the effort of searching for an adequate Cropping corpus whenever working with studies in a new application domain.

For this purpose, we decided to include into the automated process of topic cropping a phase for analyzing the working corpus coverage and a phase of automatic corpus tailoring. The tailoring phase creates a tailored domain-specific corpus from a large corpus with a very wide coverage such as Wikipedia. This implies a four step process for topic cropping (see also Figure 3.1):

1. Analyzing working corpus coverage by selecting characteristic terms
2. Tailoring a Cropping corpus by collecting relevant documents
3. Learning a topic model from the Cropping corpus
4. Applying topic inference to the working corpus

This process is embedded into a generalizable framework, which can be adapted to different settings via parameters. The final aim is to learn those parameters of the process steps in a self-optimizing loop.

**Analyzing Working Corpus Coverage**: For tailoring the Cropping corpus, we first have to understand the topical coverage of the corpus under consideration. At first glance, this might look like a hen-egg problem: we need to know the main topics of the corpus for building a corpus for learning those topics. For overcoming this, we relied on a method for determining the most relevant terms by using a counter corpus. Starting from a particular case in the study under consideration and a random subset of pages selected from Wikipedia, we used the metric of Mutual Information (MI) [MRS08], which measures how much the joint distribution of terms deviates from a hypothetical distribution in which features and categories (working corpus and Wikipedia corpus in our case) are independent of

each other. The measure ranks higher terms which are frequent in the working corpus but not in general. They are used as representative terms for corpus coverage.

**Tailoring a Cropping Corpus**: The top-ranked subset of those terms is used for tailoring the Cropping corpus. In our approach, we used a general Web search engine to identify the set of highest ranked Wikipedia pages for each of the terms. The Cropping corpus is created from the set union of all those pages. Wikipedia has been selected as the starting point for Cropping corpus creation because of its broad coverage providing information on seemingly every possible topic. Of course it is also possible to use large domain specific corpora or combinations of several corpora.

**Learning the Topic Model**: For learning the topic model, we made use of the Mallet topic modeling toolkit [McC02], namely the class ParallelTopicModel. This class offers a simple parallel threaded implementation of LDA (see [NASW09]) together with SparseLDA sampling scheme and data structure from [YMM09]. LDA models documents as probabilistic combinations of topics $P(z|d)$, with each topic described by terms following another probability distribution i.e. $P(w|z)$.

$$P(w_i) = \sum_{j=1}^{T} P(w_i|z_i = j)P(z_i = j) \tag{3.1}$$

where $P(w_i)$ is the probability of the $i$th word for a given document and $z_i$ is the latent topic. $P(w_i|z_i = j)$ is the probability of $w_i$ within topic $j$. $P(z_i = j)$ is the probability of picking a word from topic $j$ in the document. These probability distributions are specified by LDA using Dirichlet distributions. The number of latent topics $T$ has to be defined in advance and allows to adjust the degree of specialization of the latent topics. For inference and parameter estimation, Gibbs sampling iterates multiple times over each word $w_i$ in document $d_i$, and samples a new topic $j$ for the word based on the probability $P(z_i = j|w_i, d_i, z_{-i})$ until the LDA model parameters converge.

**Applying the Topic Model**: In this step, the topic model learned from the Cropping corpus is applied to the working corpus using topic inference as offered by the Mallet toolkit. It is not expected that the set of topics learned from the Cropping corpus is exactly the set of topics inherently included in the working corpus. Rather, the set of topics learned from the Cropping corpus is roughly a superset of the working corpus topics. Learned topics that are not available in the working corpus will however have no major impact on the topic inference process as long as the "real" working corpus topics are also in the learned topic model. Topic inference will assign to each of the topics in the topic model a probability of it being relevant for a study document under consideration.

### 3.2.3 Experimental Setup

**Dataset.** For our experiments, we re-used qualitative data shared via the ESDS Qualidata / the UK Data Service. We selected four out of the eight cases from the case study on "Changing Organizational Forms and the Re-shaping of Work" [MRW04]. Each case

has verbatim transcriptions or summaries of in-depth Face-to-face interviews conducted in England and Scotland between 1999 and 2002. The study surveyed employees from inter-organisational networks as new organisational forms, analysing how they operate in practice and focusing on the aspect of employment relationship.

1. *Airport case*: four airlines, engineering department, airport security, baggage handling, full handling, cleaning company, fire service (30 files)
2. *Ceramics case*: five ceramics manufacturers (32 files)
3. *Chemicals case*: a pigment manufacturing plant, two Suppliers, two Transportation specialists, two Business Service Contractors (28 files)
4. *PFI case*: Hotel Services Company, Facilities Design Company, Special Purpose Vehicle, NHS Trust Monitoring Team (41 files)

Interviews were held in semi-structured form given guidelines for questions along the main research themes of managing, learning and knowledge development, experience of work, and performance – particularly investigating the links between these topics and changing organizational forms[1]. Participants were managers and employees at all levels, sometimes also union representatives. The number of pages per document varies between two and 32 for verbatim transcriptsand summaries are usually of two to ten pages in length. These interview documents consist of transcribed spoken, natural language with answers being usually short, often elliptic, and requiring co-text and context for interpretation.

**Experimental Settings.** For tailoring the Cropping corpus we used the top 20 most representative terms as identified in the working corpus analysis phase. The Bing Search engine was queried for each of those terms individually to retrieve relevant Wikipedia pages. This resulted in a Cropping corpus of about 10,000 documents.

An important parameter in learning the topic model is the number of topics to be learned. With an increasing number of topics – a parameter of the topic model learning process – the topics get more fine-grained. The challenge here is to find a number, which results in good topic coverage for the study (all relevant topics are in) and in sufficiently fine-grained topics to help exploring unknown qualitative material while still being useful for human understanding and for spotting areas with similar topics. There is no general notion of a "good" number of topics since this strongly depends on the corpus and the application. We decided to take topic diversity as a measure for an appropriate number of topics, more precisely the diversity of the topics assigned to the study based on the topics learned from the Cropping corpus. The intuition behind this is that we need a sufficiently large topic model to cover all aspects of the study. Once the diversity stops increasing substantially the newly added topics are either not relevant for the study or they just provide subtopics by splitting topics, which does not substantially add to the diversity. Figure 3.2

[1]For more details see: http://discover.ukdataservice.ac.uk/catalogue?sn=5041

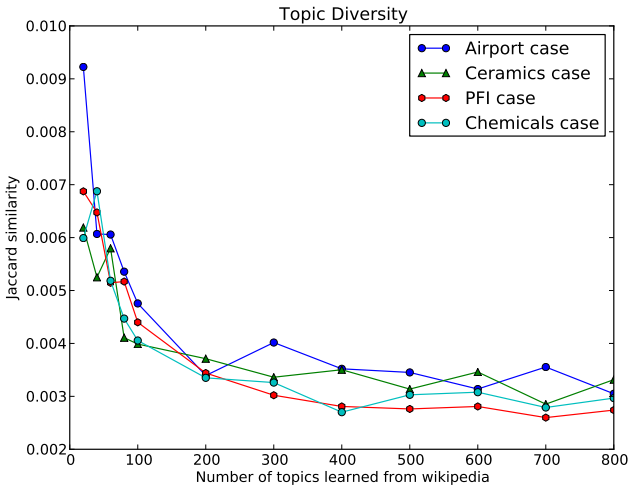



**Figure 3.2** Topic diversity, measured via Jaccard similarity for various number of topics learned from the Cropping corpus

shows the increase in topic diversity for various numbers of topics learned from the Cropping corpus. For this topic inference we used a threshold of 0.01 to cut out "noisy" topics with very low probabilities. Figure 3.2 is discussed in more detail in the next section.

### 3.2.4 Results and Discussions

We judge the quality of the automatically detected topics exploiting both, internal (intrinsic) and external (extrinsic) evaluation [MRS08, NLGB10]. In topic analysis, an internal evaluation prefers low similarity between topics whilst within a topic high similarity is favored. We adopt this idea by measuring *topic diversity* capturing variance between the different topics in a model and *topic coherence* within the single topics respectively. We additionally measure *topic relevance* externally by comparing with human annotators. In this section, we evaluate both the topics learned directly from the working corpus and those from the Cropping corpus with the same setting and analyze them with respect to these quality dimensions.

**Topic Diversity.** It is an important criterion for judging the quality of a learned model. The more diverse, i.e. dissimilar, the resulting topics are, the higher will be the coverage regarding the various aspects talked about in our interview data. It has been shown in earlier work that the Jaccard Index is an adequate proxy for diversity [DSZ12] and its output value correlates with a number of clusters (topics in our case) within the dataset. Thus, to estimate the average similarity between produced clusters, we employ the popular Jaccard coefficient [MRS08]. Given two topic models $T_i$ and $T_j$ where each topic is a set of terms $\{w_1^i, w_2^i, ..., w_k^i\}$ and $\{w_1^j, w_2^j, ..., w_k^j\}$ respectively, their Jaccard similarity $JS(T_i, T_j)$

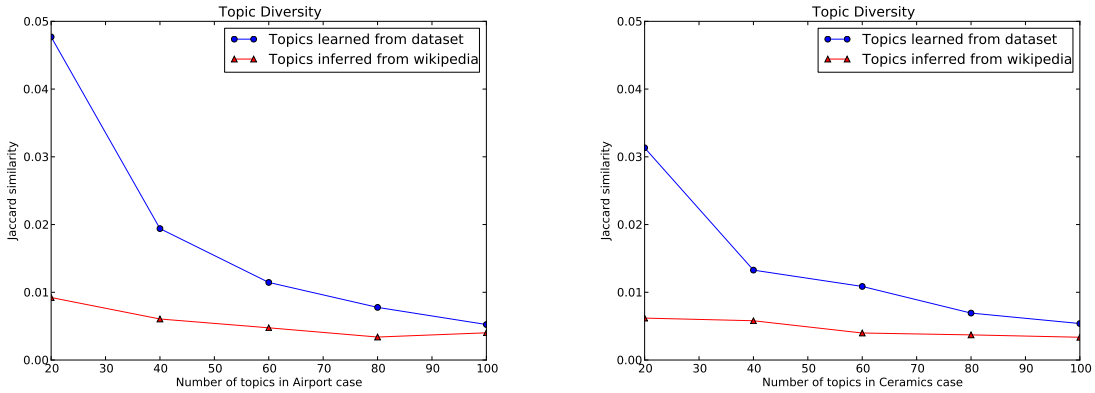



**Figure 3.3** Topic diversity, measured via Jaccard similarity, and its variance for different numbers of topics learned during topic modeling.

is defined as follows:

$$JS(T_i, T_j) = \frac{|T_i \cap T_j|}{|T_i \cup T_j|} \tag{3.2}$$

Given a collection of topic models $T_1, \ldots, T_n$, the refined (excluding self-similar pairs) average Jaccard similarity [DSZ12] is defined as follows ($1 \leq i < j \leq n$):

$$\text{avgSim} = \frac{2}{n(n-1)} \sum_{i<j} JS(T_i, T_j) \tag{3.3}$$

For all available cases, Figure 3.2 plots topic diversity with respect to the number of inferred topics. We observe that similarity values sharply decrease until the number of topics reaches the range 80-100. They do not substantially change in the tail. This may be an indicator for a reasonable number of topics for our datasets. Similarly, Figure 3.3 shows the change of the average Jaccard similarity, comparing the diversity of topics learned from the working and the Cropping dataset. We observe that topics learned from the Cropping corpus are generally more diverse in the beginning of the curve, indicating that our approach covers more aspects of the data even for smaller number of topics.

**Topic Coherence.** We tackle the task of topic coherence evaluation by rating coherence or interpretability based on an adaptation of the Google similarity distance, which performs effectively in measuring similarity between words [CV07]. The more similar, i.e less distant, the representative words within a topic, the higher or easier is its interpretability. Cilibrasi and Vitanyi's *normalized Google distance* (**NGD**) function measures how close word $x$ is to word $y$ on a zero to infinity scale using the formula:

$$\textbf{NGD}(x, y) = \frac{\max\{\log f(x), \log f(y)\} - \log f(x, y)}{\log M - \min\{\log f(x), \log f(y)\}} \tag{3.4}$$

where $f(x)$ and $f(y)$ are the number of hits of words $x$ and $y$, respectively, $f(x, y)$ is the page-counts for the query $x\ AND\ y$ and $M$ is the total number of web pages that Google

| Corpus | Topics | NGD |
|---|---|---|
| W | bag day company baggage ramp | 0.44 |
| | airline service issue baggage handling | 0.38 |
| C | workers labor work employment workforce | 0.19 |
| | employee employees tax employer pay | 0.19 |

**Table 3.1** Example topics with coherence measured via normalized Google distance, topics inferred from the working corpus (*W*) or the Cropping corpus (*C*).

| Case | AvgNGD$_W$ | SD$_W$ | AvgNGD$_C$ | SD$_C$ |
|---|---|---|---|---|
| Airport | 0.34 | 0.07 | **0.21** | 0.08 |
| Ceramics | 0.32 | 0.08 | **0.25** | 0.09 |
| Pfi | 0.35 | 0.1 | **0.22** | 0.08 |

**Table 3.2** Average (Avg) and standard deviation (SD) of topic coherence of three cases, measured via normalized Google distance (NGD). Topics are inferred from the working corpus (*W*) or the Cropping corpus (*C*).

indexes. A NGD of zero indicates that word $x$ and word $y$ are practically the same. They are independent when their distance reaches approximately one.

Given a topic $T_i$ which is represented by its top-m words (we set m=5 in this experiment) denoted by $\mathbf{w} = (w_1, ..., w_m)$, its normalized Google distance is:

$$\mathbf{NGD}(T_i) = \frac{2}{m(m-1)} \sum_{w_i, w_j \in \mathbf{w}} \mathbf{NGD}(w_i, w_j) \tag{3.5}$$

To estimate overall topic coherence, we randomly choose a list of 30 learned topics per case, i.e.$T = (T_1, ..., T_n)$, compute NGD for each $T_i$, and then take the average of the list $\mathbf{AvgNGD}(T) = \frac{1}{n}\mathbf{NGD}(T_i)$.

Table 3.2 reports the average normalized Google distances and their deviations for topics inferred for three cases. For all cases evaluated, we obtain consistent improvement. Specifically, evaluating over the 90 topics of these three cases, we improve 32% in terms of normalized Google distance. This indicates that the topics inferred from the Cropping corpus are significantly more coherent than those learned directly from the working corpus (significance of a t-test $p < 0.001$).

**Topic Relevance.** While topic diversity and topic coherence can help to estimate the quality of the topics with respect to information-theoretic considerations, validity of our results, i.e., the usefulness of the derived topics for the working corpus, needs to be assessed by

human evaluation of topic relevance. Here, we decided to compare our inferred topics with topics assigned by human annotators. For this evaluation, we randomly selected 16 documents from the study to be manually annotated by four users. Each document was split into smaller units – typically question and answer pairs – resulting in about 60 units per document. Thus, a total of 1000 units was annotated. We asked users to define topics discussed in each given unit. Each unit could have one or more topics and there were no restrictions on how topics are to be phrased. Typically the topics assigned were single words or short phrases.

Topic relevance is then assessed by automatically matching user defined topics with the learned ones. For this, the terms used by the user for a topic are matched with the top terms learned for a topic by the topic model. We consider it a match if the term used by the user appears in the top terms of the respective topic. By design, this evaluation gives preference to the topic model learned directly from the working corpus since the users tend to use terms that appear in the text. Similarly, the topic models learned directly on the working corpus use exactly those terms for their topics. In order to even out this terminology disadvantage, we made use of word synonyms from WordNet [Mil95] to extend sets of topic words before matching.

A learned topic $T_i$ is considered to be relevant if its representative words and their synonyms $\mathbf{w} = (w_1, ..., w_k)$ share one or more terms with user defined topics $\mathbf{t} = (t_1, ..., t_r)$

$$\mathbf{Rel}(T) = \begin{cases} 1 & \text{if } \mid \mathbf{w} \cap \mathbf{t} \mid > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

There are two reasons to use this type of evaluation in spite of its weakness: First, the alternative solution of showing the user the learned topic together with the text for relevance assessment puts a high burden on the user since it is not trivial to judge automatically learned topics. In addition, there is the risk that the user also unintentionally assesses topic quality in terms of coherence at the same time. Second, we are aiming for a self-optimizing loop, where parameters of the process are adapted iteratively through learning based on quality assessment. In this context, the evaluation of topic relevance chosen here only has to be done once and can be re-used in every iteration. The alternative manual evaluation of the relevance of each learned topic as a whole would have to be repeated in every loop to assess the newly learned topics.

For two example documents, Figure 3.4 compares topics learned from the working and Cropping corpus with respect to the number of relevant topic at rank $k$, $\mathbf{R}@k = \sum_{i=1}^{k} \mathbf{Rel}(T_i)$, where the rank is determined by the probability of the topic assignment (resulting from topic inference). We achieve similar results for other documents. On average, at rank 10 we obtain 9.8 relevant topics with a deviation of 0.35 for the working topics and 9.2 with a deviation of 1.0 for the Cropping topics. It can be seen from the results that the topics learned from Wikipedia reach a comparable level of relevance as those learned directly from the corpus, while being more coherent and diverse.

To sum up, in this work, we propose a method for a *fully automated* and adaptable process of tailoring a domain-specific sub-corpus from a general corpus such as Wikipedia

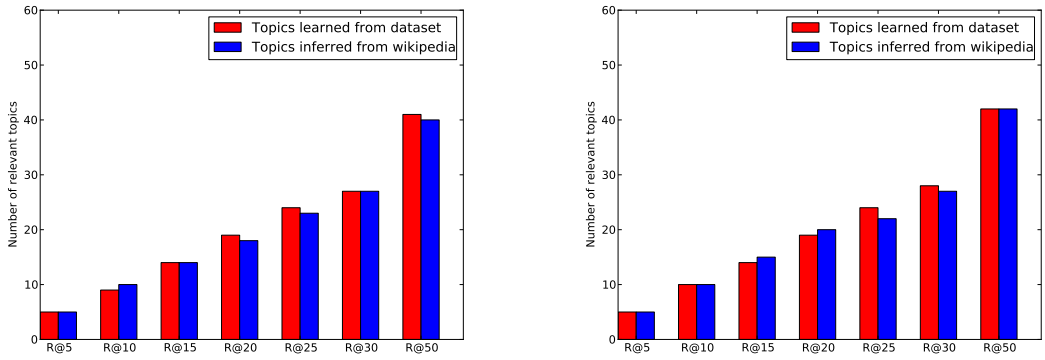



**Figure 3.4** Topic relevance as the number of relevant topics at rank $k$, for two documents

and exploiting it to increase the topic model quality for limited size corpora such as studies in sociology and other qualitative material in the digital humanities. Our experiments show substantial improvements in diversity as well as in internal coherence of inferred topics compared to a naive approach using the limited size corpora exclusively. At the same time our method keeps the topic relevance high as confirmed by human annotators. We believe that our approach can be further improved by exploiting the automatic evaluation for adjusting the input parameters of the algorithm.

## 3.3 Multiplicative Tree-Structured LSTMs for Semantic Representations

In the previous section, documents are represented by a distribution of topics, in which each topic is a list of related words. In this section, we focus on another representation of documents, i.e., distributed vector representation. Learning such distributed representation has drawn great attention recently, and become a crucial step of various natural language processing (NLP) tasks such as text classification [ZLP15, Kim14], semantic matching [LQZ+16], and machine translation [CvMG+14]. Seminal work uses recurrent neural networks (RNN) [Elm90], convolutional neural networks [KGB14], and tree-structured neural networks [SLNM11, TSM15] for sequence and tree modeling. Long Short-Term Memory (LSTM) [HS97] networks are a type of recurrent neural network that are capable of learning long-term dependencies across sequences and have achieved significant improvements in a variety of sequence tasks. LSTM has been extended to model tree structures (e.g., TreeLSTM) and produced promising results in tasks such as sentiment classification [TSM15, ZSG15] and relation extraction [MB16b].

Figure 3.5 shows the topologies of the conventional chain-structured LSTM [HS97] and the TreeLSTM [TSM15], illustrating the input $(x)$, cell $(c)$ and hidden node $(h)$ at a time

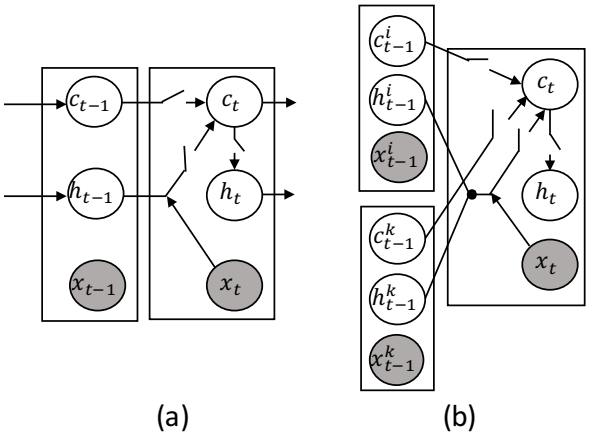



**Figure 3.5** Topology of sequential LSTM and TreeLSTM: (a) nodes in sequential LSTM and (b) nodes in tree-structured LSTM

step $t$. The key difference between Figure 3.5 (a) and (b) is the branching factor. While a cell in the sequential LSTM only depends on the single previous hidden node, a cell in the tree-structured LSTM depends on the hidden states of child nodes.

Despite their success, the tree-structured models have a limitation in their inability to fully capture the richness of compositionality [SBMAY13]. The same combination function is used for all kinds of semantic compositions, though the compositions have different characteristics in nature. For example, the composition of the adjective and the noun differs significantly from the composition of the verb and the noun.

To alleviate this problem, some researchers propose to use multiple compositional functions, which are predefined according to some partition criterion [SHMN12, SBMAY13, DWT+14]. Socher et al. [SBMAY13] defined different compositional functions in terms of syntactic categories, and a suitable compositional function is selected based on the syntactic categories. Dong et al. [DWT+14] introduced multiple compositional functions and a proper one is selected based on the input information. These models accomplished their objective to a certain extent but they still face critical challenges. The predefined compositional functions cannot cover all the compositional rules and they add much more learnable parameters, bearing the risk of overfitting.

In this work, we propose *multiplicative TreeLSTM*, an extension to the TreeLSTM model, which injects relation information into every node in the tree. It therefore allows the model to have different semantic composition matrices to combine child nodes. To reduce the model complexity and keep the number of parameters manageable, we define the composition matrices using the product of two dense matrices shared across relations, with an intermediate diagonal matrix that is relation dependent.

Though the syntactic-based models have shown to be promising for compositional semantics, they do not make full use of the linguistic information. For example, semantic nodes are often the argument of more than one predicate (e.g., coreference) and it is gen-

erally useful to exclude semantically vacuous words like articles or complementizers, i.e., leave nodes unattached that do not add further meaning to the resulting representations. Recently, Banarescu et al. [BBC+13] introduced Abstract Meaning Representation (AMR), single rooted, directed, acyclic graphs that incorporate semantic roles, correference, negation, and other linguistic phenomena. In this work, we investigate a combination of the semantic process provided by TreeLSTM model with the lexical semantic representation of the AMR formalism. This differs from most of existing work in this area, where syntactic rather than semantic information is incorporated to the tree-structured models. We seek to answer the question: *To what extent can we do better with AMR as opposed to syntactic representations, such as constituent and dependency trees, in tree-structured models?*

We evaluate the proposed models on three common tasks: sentiment classification, sentence relatedness, and natural language inference. The results show that the multiplicative TreeLSTM models outperform TreeLSTM models on the same tree structures. The results further suggest that using AMR as the backbone for tree-structured models is helpful in the complex task, e.g., sentence inference but not in the sentiment classification task, where lexical information alone suffices.

In short, the contributions of this work can be summed up as follows:

- We propose the new multiplicative TreeLSTM model that effectively learns distributed representation of a given sentence from its constituents, utilizing not only the lexical information of words, but also the relation information between the words.
- We conduct an extensive investigation on the usefulness of lexical semantic representation induced by AMR formalism in tree-structured models.

### 3.3.1 Related Literature

There is a line of research that extends the standard LSTM [HS97] in order to model more complex structures. Tai et al.[TSM15] and Zhu et al. [ZSG15] extended sequential LSTMs to tree-structured LSTMs by adding branching factors. They showed such extensions outperform competitive LSTM baselines on several tasks such as sentiment classification and semantic relatedness prediction (which is also confirmed in this work). Li et al. [LLJH15] further investigated the effectiveness of TreeLSTMs on various tasks and discussed when tree structures are necessary. Chen et al. [CZL+17] combined sequential and tree-structured LSTM for NLI and has achieved state-of-the-art results on the benchmark dataset. Their approach uses *n-ary* TreeLSTM based on syntactic constituency parsers. In contrast, we focus more on child-sum TreeLSTM which is better suited for trees with high branching factor.

Previous works have studied the use of relation information. Dyer et al. [DBL+15] considered each syntactic relation as an additional node and included its embedding to their composition function for dependency parsing. Peng et al. [PPQ+17] introduced a different set of parameters for each edge-type in their LSTM-based approach for relation extraction.

In contrast to these works, our mTreeLSTM model incorporates relation information via a multiplicative mechanism, which we have shown is more effective and uses less parameters.

AMR has been successfully applied to a number of NLP tasks, besides the ones we considered in this work. For example, Bitra et al. [MB16a] made use of AMR to improve question answering; Liu et al. [LFT+15] utilized AMR to produce promising results toward abstractive summarization. Using AMR as the backbone in TreeLSTM has been investigated in Takase et al. [TSO+16]. They incorporated AMR information by a neural encoder to the attention-based summarization method [RCW15] and it performed well on headline generation. Our work differs from these studies in that we aim to investigate how semantic information induced by AMR formalism can be incorporated to tree-structured LSTM models, and study which properties introduced by AMR turn out to be useful in various tasks. In this work, we use the start-of-the-art AMR parser provided by Flanigan et al.[FDSC16] which additionally provides the alignment between words and nodes in the parsed tree.

Though we have considered AMR in this work, we believe the conclusions we drew here largely apply to other semantic schemes, such as GMB and UCCA, as well. Abend et al. [AR17] has recently noted that the differences between these schemes are not critical, and the main distinguishing factors between them are their relation to syntax, their degree of universality, and the expertise they require from annotators.

### 3.3.2 Tree-Structured LSTMs

A standard LSTM processes a sentence in a sequential order, e.g., from left to right. It estimates a sequence of hidden vectors given a sequence of input vectors, through the calculation of a sequence of hidden cell vectors using a gate mechanism. Extending the standard LSTM from linear chains to tree structures leads to TreeLSTM. Unlike the standard LSTM, TreeLSTM allows richer network topologies, where each LSTM unit is able to incorporate information from multiple child units.

As in standard LSTM units, each TreeLSTM unit contains input gate $i_j$, output gate $o_j$, a memory cell $c_j$, and hidden state $h_j$ for node $j$. Unlike the standard LSTM, in TreeLSTM the gating vectors and the memory cell updates are dependent on the states of one or more child units. In addition, the TreeLSTM unit contains one forget gate $f_{jk}$ for each child $k$

instead of having a single forget gate. The transition equations of node $j$ are as follows:

$$\begin{aligned}
\tilde{h}_j &= \sum_{k \in C(j)} h_k \,, \\
i_j &= \sigma\left(W^{(i)} x_j + U^{(i)} \tilde{h}_j + b^{(i)}\right) \,, \\
o_j &= \sigma\left(W^{(o)} x_j + U^{(o)} \tilde{h}_j + b^{(o)}\right) \,, \\
f_{jk} &= \sigma\left(W^{(f)} x_j + U^{(f)} h_k + b^{(f)}\right) \,, \\
u_j &= \tanh\left(W^{(u)} x_j + U^{(u)} \tilde{h}_j + b^{(u)}\right) \,, \\
c_j &= i_j \odot u_j + \sum_{k \in C(j)} f_{jk} \odot c_k \,, \\
h_j &= o_j \odot \tanh(c_j) \,,
\end{aligned} \tag{3.7}$$

where $C(j)$ is the set of children of node $j$, $k \in C(j)$ in $f_{jk}$, $\sigma$ is the sigmoid function, and $\odot$ is element-wise (Hadamard) product. $W^{(*)}$, $U^{(*)}$, $b^{(*)}$ are model parameters with $* \in \{u, o, i, f\}$.[2]

### 3.3.3 Multiplicative Tree-Structured LSTMs

Encoding rich linguistic analysis introduces many distinct edge types or relations between nodes, such as syntactic dependencies and semantic roles. This opens up many possibilities for parametrization, but was not considered in prior syntax-aware LSTM approaches, which only make use of input node information.

In this work, we fill this gap by proposing multiplicative TreeLSTM, an extension to the TreeLSTM model, injecting relation information into every node in the tree. The multiplicative TreeLSTM model, *mTreeLSTM* for short, introduces more fined-grained parameters based on the edge types. As inspired by the multiplicative RNN [SMH11], the hidden-to-hidden propagation in mTreeLSTM contains a separately learned transition matrix $W_{hh}$ for each possible edge type and is given by

$$\tilde{h}_j = \sum_{k \in C(j)} W_{hh}^{r(j,k)} h_k \,, \tag{3.8}$$

where $r(j, k)$ signifies the connection type between node $k$ and its parent node $j$. This parametrization is straightforward, but requires a large number of parameters when there are many edge types. For instance, there are dozens of syntactic edge types, each corresponding to a Stanford dependency label.

To reduce the number of parameters, as well as leverage potential correlation among fine-grained edge types, we learned an embedding of the edge types and factorized the

[2] In Tai et al. [TSM15], the TreeLSTM defined in Equation (3.7) was referred to as *child-sum TreeLSTM*, which is a good choice for trees with high branching factor.

transition matrix $W_{hh}^{r(j,k)}$ by using the product of two dense matrices shared across edge types, with an intermediate diagonal matrix that is edge-type dependent:

$$W_{hh}^{r(j,k)} = W_{hm}\text{diag}(W_{mr}e_{jk})W_{mh}\,, \tag{3.9}$$

where $e_{jk}$ is the edge-type embedding and is jointly trained with other parameters. The mapping from $h_k$ to $\tilde{h}_j$ is then given by

$$\begin{aligned} m_{jk} &= (W_{mr}e_{jk}) \odot (W_{mh}h_k)\,, \\ \tilde{h}_j &= \sum_{k \in C(j)} W_{hm}m_{jk}\,. \end{aligned} \tag{3.10}$$

The gating units – input gate $i$, output gate $o$, and forget gate $f$ – are computed in the same way as in the TreeLSTM with Eq. (3.7).[3]

Multiplicative tree LSTM can be applied to any tree where connection types between nodes are given. For example, in dependency trees, the semantic relations $r(j,k)$ between nodes are provided by a dependency parser.

### 3.3.4 Tree-Structured LSTMs with Abstract Meaning Representation

Tree-structured LSTMs have been applied successfully to syntactic parse trees [TSM15, MB16b]. In this work, we look beyond *syntactic* properties of the text and incorporate *semantic* properties to the tree-structured LSTM model. Specifically, we utilize the network topology offered by a tree-structured LSTM and incorporate semantic features induced by AMR formalism. We aim to address the following questions: *In which tasks using AMR structures as the backbone for the tree-structured LSTM is useful? Furthermore, which semantic properties are useful for the given task?*

AMR is a semantic formalism where the meaning of a sentence is encoded as a single rooted, directed and acyclic graph [BBC+13]. For example, the sentence "*A young girl is playing on the edge of a fountain and an older woman is not watching her*" is represented as

```
(a / and
    :op1 (p / play-01
        :ARG0 (g / girl
            :mod (y / young))
        :ARG1 (e / edge-01
            :ARG1 (f / fountain)))
    :op2 (w / watch-01
        :ARG0 (w2 / woman
            :mod (o / old))
        :ARG1 g
        :polarity -))
```

[3] In the rest of the work, we use the term *TreeLSTM* in a narrow sense to refer to the model corresponding to Equation (3.7) and the term *tree-structured LSTM* to include both TreeLSTM and mTreeLSTM, unless specified otherwise.

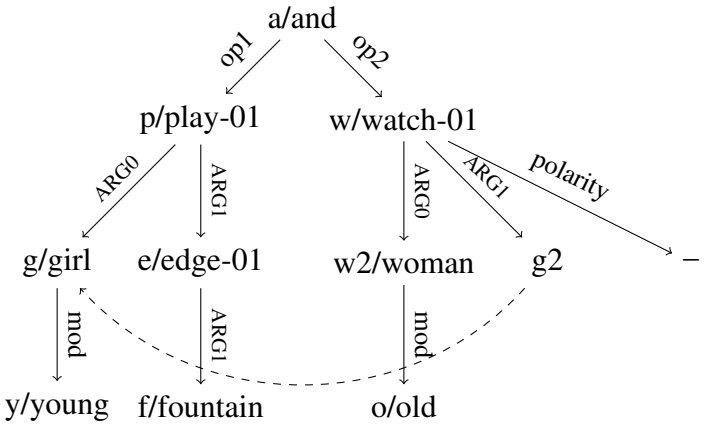



**Figure 3.6** An AMR representing the sentence “A young girl is playing on the edge of a fountain and an older woman is not watching her".

The same AMR can be represented as in Figure 3.6, in which the nodes in the graph (also called concepts) map to words in the sentence and the edges represent the relations between words. AMR concepts consist of predicate senses, named entity annotations, and in some cases, simply lemmas of English words. AMR relations consist of core semantic roles drawn from the Propbank [PGK05] as well as fine-grained semantic relations defined specifically for AMR. Since AMR provides a whole-sentence semantic representation, it captures long-range dependencies among constituent words in a sentence. Similar to other semantic schemes, such as UCCA [AR13], GMB [BBEV12], UDS [WRS+16], AMR abstracts away from morphological and syntactic variability and generalize cross-linguistically.

To use AMR structures in a tree-structured LSTM, we first parse sentences to AMR graphs and transform the graphs to tree structures. The transformation follows the procedure used by Takase et al. [TSO+16], splits the nodes with an indegree larger than one, which mainly present coreferential concepts, to a set of separate nodes, whose indegrees exactly equal one. We use JAMR [FTC+14, FDSC16], a statistical semantic parser trained on AMR bank, for AMR parsing.

On one hand, the AMR tree structure can be used directly with the TreeLSTM architecture described in Section 3.3.2, in which only node information is utilized to encode sentences into certain fixed-length embedding vectors. On the other hand, since AMR provides rich information about semantic relations between nodes, the mTreeLSTM architecture is more applicable due to its capability of modeling edges in the tree. We evaluate both encoded vectors produced by TreeLSTM and mTreeLSTM on AMR trees in Section 3.3.6.

### 3.3.5 Applications

In this section, we describe three specific models that apply the mTreeLSTM architecture and the AMR tree structures described in the previous sections.

**Sentiment Classification**: In this task, we wish to predict the sentiment of sentences, in which two sub-tasks are considered: binary classification and fine-grained multiclass classification. In the former, sentences are classified into two classes (*positive* and *negative*), while in the latter they are categorized into five classes (*very positive, positive, neutral, negative*, and *very negative*).

Given a sentence $x$, we first compute the distributed representation $h_r$ of the sentence at the root node $r$ of the tree. A softmax classifier is then used to predict the label $\hat{y}$ of the sentence:

$$\begin{aligned} \hat{p}_\theta\left(y \,|\, x\right) &= \text{softmax}\left(W^{(s)} h_r\right) , \\ \hat{y} &= \underset{y}{\text{argmax}}\, \hat{p}_\theta\left(y \,|\, x\right) , \end{aligned} \tag{3.11}$$

where $\theta$ is the set of model parameters. The cost function is the negative log-likelihood of the true sentiment class of the sentence

$$J(\theta) = -\frac{1}{m} \sum_{k=1}^{m} \log \hat{p}_{y_r^{(k)}} + \frac{\lambda}{2} \|\theta\|^2 , \tag{3.12}$$

where $m$ is the number of labeled sentences in the training set, $\lambda$ is a regularization parameter, and $y_r^{(k)}$ is the ground-truth label of the $k$th training sentence.

**Semantic Relatedness**: The goal of this task is to estimate the similarity between two sentences. Given a sentence pair, we aim to predict an integer-valued similarity score in $\{1, 2, ..., K\}$, where higher scores indicate greater degrees of similarity.

Following the procedure described in [TSM15], we first produce semantic representation $h_L$ and $h_R$ for each sentence in the pair using the described models over each sentence's parse trees. Then, we predict the similarity score $\hat{y}$ using a neural network that considers both distance and angle between the pair $(h_L, h_R)$:

$$\begin{aligned} x_s &= \begin{bmatrix} |h_L - h_R| \\ h_L \odot h_R \end{bmatrix} , \\ h_s &= \text{sigmoid}\left(W^{(s)} x_s\right) , \\ \hat{p}_\theta &= \text{softmax}\left(W^{(p)} h_s\right) , \\ \hat{y} &= r^\top \hat{p}_\theta , \end{aligned} \tag{3.13}$$

where $r^\top = [1, 2, \ldots, K]$ and the absolute value function is applied element-wise. Similar to Tai et al. [TSM15], we define a sparse target distribution $p$ such that the ground-truth

rating $y \in [1, K]$ equals $r^\top p$ and use the regularized KL-divergence between $p$ and $\hat{p}_\theta$ as the cost function:

$$J(\theta) = \frac{1}{m} \sum_{k=1}^{m} \text{KL}\left(p^{(k)} \,\middle\|\, \hat{p}_\theta^{(k)}\right) + \frac{\lambda}{2} \|\theta\|^2 \,, \tag{3.14}$$

where $m$ is the number of training pairs.

**Natural Language Inference**: In this task, the model reads two sentences (a premise and a hypothesis), and outputs a judgment of *entailment, contradiction*, or *neutral*, reflecting the relationship between the meanings of the two sentences. The aim of this task is to evaluate a model's ability to extract broadly informative representations of sentence meaning.

Following Bowman et al. [BGR+16], we frame the inference task as a sentence pair classification. First we produce representations $h_P$ and $h_H$ for the premise and hypothesis, respectively, and then construct a feature vector $x_c$ for the pair that consists of the concatenation of these two vectors, their difference, and their element-wise product. This feature vector is then passed to a neural network with a softmax layer to yield a distribution over the three labels:

$$\begin{aligned} x_c &= \begin{bmatrix} h_P \\ h_H \\ h_P - h_H \\ h_P \odot h_H \end{bmatrix} \,, \\ h_c &= \text{sigmoid}\left(W^{(c)} x_c\right) \,, \\ \hat{p}_\theta &= \text{softmax}\left(W^{(p)} h_c\right) \,. \end{aligned} \tag{3.15}$$

The negative log-likelihood of the true class labels for sentence pairs is used as the cost function:

$$J(\theta) = -\frac{1}{m} \sum_{k=1}^{m} \log \hat{p}_{y^{(k)}} + \frac{\lambda}{2} \|\theta\|^2 \,, \tag{3.16}$$

where $m$ is the number of sentence pairs in the training set.

### 3.3.6 Experimental Setup

The model parameters are optimized using AdaGrad [DHS11] with a learning rate of $0.05$ for the first two tasks, and Adam [KB15] with a learning rate of $0.001$ for the NLI task. The batch size of $25$ was used for all tasks and the model parameters were regularized with a per-minibatch L2 regularization strength of $10^{-4}$. The sentiment and inference classifiers were additionally regularized using dropout with a dropout rate of $0.5$.

Following Tai et al. [TSM15] and Zhu et al. [ZSG15], we initialized the word embeddings with $300$-dimensional Glove vectors [PSM14]. In addition, we use the aligner provided by JAMR parser to align the sentences with the AMR trees and then generate the embedding by using the Glove vectors. The relation embeddings were randomly sampled

| Model | Phrase-level training | | Root-level training | |
|---|---|---|---|---|
| | Fine-grained | Binary | Fine-grained | Binary |
| LSTM | 48.0 (1.0) | 86.7 (0.7) | 45.6 (1.1) | 85.6 (0.5) |
| TreeLSTM (C) | **49.8** (0.8) | **87.9** (0.9) | 46.3 (0.7) | **85.8** (0.5) |
| TreeLSTM (D) | 46.9 (0.2) | 85.5 (0.4) | 46.0 (0.3) | 85.0 (0.4) |
| TreeLSTM (A) | n/a | n/a | 44.4 (0.2) | 82.9 (0.6) |
| mTreeLSTM (A) | n/a | n/a | 45.2 (0.5) | 83.2 (0.5) |
| mTreeLSTM (D) | 47.5 (0.7) | 85.7 (0.1) | **46.7** (0.8) | 85.7 (0.8) |

**Table 3.3** Accuracy on the Stanford Sentiment Treebank dataset with standard deviation in parentheses (numbers in percentage).

from an uniform distribution in $[-0.05, 0.05]$ with a size of $100$. The word and relation embeddings were updated during training with a learning rate of $0.1$.

We use one hidden layer and the same dimensionality settings for sequential LSTM and tree-structured LSTMs. LSTM hidden states are of size $150$. The output hidden size is $50$ for the relatedness task and the NLI task. Each model is trained for $10$ iterations. The same training procedure repeats $5$ times with parameters being evaluated at the end of every iteration on the development set. The model having the best results on the development set is used for final tests.

For all sentences in the datasets, we parse them with constituency parser [KM03], dependency parser [CM14], and AMR parser [FTC+14, FDSC16] to obtain the tree structures. We compare our mTreeLSTM model with two baselines: LSTM and TreeLSTM. We use the notation (C), (D), and (A) to denote the tree structures that the models are based on, where they stand for constituency trees, dependency trees, and AMR trees, respectively. The code to reproduce the results is available at https://github.com/namkhanhtran/m-treelstm.[4]

### 3.3.7 Results and Discussions

**Sentiment Classification**: For this task, we use the Stanford Sentiment Treebank [SPW+13] with the standard train/dev/test splits of 6920/872/1821 for the binary classification sub-task, and 8544/1101/2210 for the fine-grained classification sub-task. We used two different settings for training: *root-level* and *phrase-level*. In the root-level setting, we use each sentence as a data point, while in the phrase-level setting, each phrase is reconstructed from nodes in the parse tree and treated as a separate data point. It is noted that in the phrase-level setting we obtain much more data for training, but the root-level setting is closer to real-world applications. In addition, since it is too expensive to have labeled data for AMR

[4]The correctness of our implementation is also suggested by the fact that we have reproduced the results of LSTM and TreeLSTM by Tai et al. [TSM15], up to small variations.

trees in the phrase-level setting, we only report the results on the root-level setting. We evaluate our models and baseline models at the sentence level.

Table 3.3 shows the main results for the sentiment classification task. While LSTM model obtains quite good performance in both settings, TreeLSTM model on constituency tree obtains better results, especially in the phrase level setting, which has more supervision. It confirms the conclusion from Tai et al. [TSM15] that combining linguistic knowledge with LSTM leads to better performance than sequence models in this task. Table 3.3 also shows mTreeLSTM consistently outperform TreeLSTM on the same tree structures in both settings. It demonstrates the effectiveness of the relation multiplication mechanism and the importance of modeling relation information. The TreeLSTM and mTreeLSTM models with AMR trees do not perform well on this task. Synthetic information along goes a long way in determining the sentiment of a sentence. Moreover, the noisy sentences in this task impact the accuracy of the AMR parser. Parse errors confuse the LSTM learner, limiting the potential gain. The performance may be improved with future, better AMR parsers.

We dive deep into what the models learn by listing the composition matrices $W_{hh}^{r(j,k)}$ with the largest Frobenius norms. Intuitively, these matrices have learned larger weights, which are in turn being multiplied with the child hidden states. That child will therefore have more weight in the composed parent vector. In decreasing order of Frobenius norm, the relationship matrices for mTreeLSTM on dependency trees are: conjunction, adjectival modifier, object of a preposition, negation modifier, verbal modifier. The relationship matrices for mTreeLSTM on AMR trees are: negation (`:polarity`), attribute (`:ARG3`, `:ARG2`), modifier (`:mod`), conjunction (`:opN`). The model learns that verbal and adjective modifiers are more important than nouns, as they tend to affect the sentiment of sentences.

**Sentence Relatedness**: For this task, we use the Sentences Involving Compositional Knowledge (SICK) dataset, consisting of 9927 sentence pairs with the standard train/dev/test split of 4500/500/4927. Each pair is annotated with a relatedness score $y \in [1, 5]$, with 1 indicating the two sentences are completely unrelated, and 5 indicating they are very related. Each label is the average of 10 ratings assigned by different human annotators. Following Tai et al. [TSM15], we use Pearson, Spearman correlations, and mean squared error (MSE) as evaluation metrics.

Our results are summarized in Table 3.4. The tree-structured LSTMs, both TreeLSTM and mTreeLSTM, reach better performance than the standard LSTM. The model using dependency tree as the backbone achieves best results. The mTreeLSTM with AMR trees obtain slightly better results than the TreeLSTM with constituency trees. The multiplicative TreeLSTM models outperform the TreeLSTM models on the same parse trees, illustrating again the usefulness of incorporating relation information into the model.

Similar to the previous experiment, we list the composition matrices $W_{hh}^{r(j,k)}$ with the largest Frobenius norms. The relationship matrices for dependency trees include: indirect object, marker for introducing a finite clause subordinate to another clause, negation modifier, adjectival modifier, phrasal verb particle, conjunction. The relationship ma-

| Model | Pearson | Spearman | MSE |
|---|---|---|---|
| LSTM | 0.8409 (0.0036) | 0.7782 (0.0058) | 0.3035 (0.0033) |
| TreeLSTM (C) | 0.8497 (0.0049) | 0.7904 (0.0040) | 0.2861 (0.0101) |
| TreeLSTM (D) | 0.8631 (0.0026) | 0.8034 (0.0024) | 0.2600 (0.0045) |
| TreeLSTM (A) | 0.8415 (0.0027) | 0.7742 (0.0012) | 0.2986 (0.0045) |
| mTreeLSTM (A) | 0.8527 (0.0006) | 0.7884 (0.0006) | 0.2788 (0.0015) |
| mTreeLSTM (D) | **0.8717** (0.0037) | **0.8141** (0.0048) | **0.2443** (0.0069) |

**Table 3.4** Results on the SICK dataset for semantic relatedness task with standard deviation in parentheses

| Model | All | LS | Negation |
|---|---|---|---|
| LSTM | 77.3 (0.5) | 74.6 (1.4) | 77.5 (0.4) |
| TreeLSTM (C) | 79.0 (1.4) | 78.1 (2.9) | 85.3 (1.2) |
| TreeLSTM (D) | 82.9 (0.3) | 81.0 (2.6) | 84.3 (1.2) |
| TreeLSTM (A) | 82.6 (0.2) | 84.0 (1.5) | 88.2 (0.4) |
| mTreeLSTM (A) | 83.3 (0.2) | **85.3** (0.4) | **88.5** (0.8) |
| mTreeLSTM (D) | **84.0** (0.5) | 81.6 (1.3) | 87.8 (0.8) |

**Table 3.5** Accuracy on the SICK dataset for the natural language inference task with standard deviation in parentheses (numbers in percentage)

trices for AMR trees are: patient (`:ARG1`), comparatives and superlatives (`:degree`), agent (`:ARG0`), attribute (`:ARG3`), medium (`:medium`), possession (`:poss`), manner (`:manner`).

**Natural Language Inference**: In this task, we first look at the SICK dataset described in the previous section. In this setting each sentence pair is classified into three labels, *entailment, contradiction*, and *neutral*.

In addition to the standard test set, we also report performances of our models on two different subsets. The first subset, *Long Sentence (LS)*, consists of sentence pairs in the test set where the premise sentence contains at least 18 words. We hypothesize that long sentences are more difficult to handle by sequential models as well as tree-structured models. The second subset, *Negation*, is a set of sentence pairs where negation words (*not*, *n't* or *no*) do not appear in the premise but appear in the hypothesis. In the test set, 58.7% of these examples are labeled as *contradiction*.

Table 3.5 summarizes the results of our models on different test sets. The mTreeLSTM models obtain highest results, followed by TreeLSTM models. The standard LSTM model does not work well on this task. The results reconfirm the benefit of using the structure information of sentences in learning semantic representations. In addition, Ta-

| Model | Acc (%) |
|---|---|
| LSTM [BAPM15] | 77.6 |
| Syntax TreeLSTM [YBD+17] | 80.5 |
| CYK TreeLSTM [MCY17] * | 81.6 |
| Gumbel TreeLSTM [CYgL18] | 81.8 |
| Gumbel TreeLSTM + leaf LSTM [CYgL18] | 82.6 |
| TreeLSTM (D) | 81.0 |
| mTreeLSTM (D) | 81.9 |

**Table 3.6** Results on the SNLI dataset. The first group contains results of some best-performing tree-structured LSTM models on this data. (*: a preprint)

ble 3.5 shows that TreeLSTM on dependency trees and AMR trees outperform the models with constituency trees. The dependency trees provide some semantic information, i.e., semantic relations between words at some degrees, while AMR trees present more semantic information. The multiplicative TreeLSTM on AMR trees perform much better than other models on the *LS* and *Negation* subsets. The results on the *LS* subset shows that mTreeLSTM on AMR trees can handle long-range dependencies in a sentence more effectively. For example, only mTreeLSTM (A) is able to predict the following example correctly:

**Premise**: *The grotto with a pink interior is being climbed by four middle eastern children, three girls and one boy.*
**Hypothesis**: *A group of kids is playing on a colorful structure.*
**Label**: *entailment*

Similar to previous experiments, we list the composition matrices with the largest Frobenius norms to get some insights into what the models learn. The relationship matrices for mTreeLSTM on dependency trees are: negation modifier, nominal subject, adjectival modifier, direct object, passive auxiliary, adverb modifier. These matrices for mTreeLSTM on AMR trees are: attribute (`:ARG2`), patient (`:ARG1`), conjunction (`:opN`), location, negation (`:polarity`), domain. In contrast to the sentiment classification task, where adjectives are crucial, the model learns that subjects and objects are important to determine the meaning of sentences.

Furthermore, we evaluate our mTreeLSTM model with SNLI (Stanford Natural Language Inference), a larger NLI dataset [BAPM15]. It is composed of about 550K/10K/10K sentence pairs in train/dev/test sets. We use dependency tree as the backbone for tree-structured LSTMs. All models in Table 3.6 use a hidden size of 100 for a fair comparison. The table shows that mTreeLSTM (D) outperforms many other syntax-based TreeLSTM models including TreeLSTM (D), reconfirming our conclusion drawn with SICK.

Incorporating relation information in the tree-structured LSTM increases model complexity. In this experiment, we analyze the impact of the dimensionality of relation embedding on the model size and accuracy. Table 3.7 shows the model with the relation

| Model | rDim | # Params | Acc (%) |
|---|---|---|---|
| TreeLSTM (A) | n/a | 301K | 82.6 |
| mTreeLSTM (A) | 50 | 354K | 82.7 |
| mTreeLSTM (A) | 75 | 358K | 83.1 |
| mTreeLSTM (A) | 100 | 361K | 83.6 |
| mTreeLSTM (A) | 200 | 376K | 83.0 |

**Table 3.7** Effects of the relation embedding size on SICK dataset for the NLI task

| Model | # Params | Acc (%) |
|---|---|---|
| TreeLSTM (D) | 301K | 82.9 |
| addTreeLSTM (D) | 361K | 83.4 |
| fullTreeLSTM (D) | 1.1M | 83.5 |
| mTreeLSTM (D) | 361K | 84.0 |

**Table 3.8** Comparison between different methods using relation information on the SICK dataset for the NLI task

embedding size of 100 achieves the best accuracy, while the overall impact of the embedding size is mild. The multiplicative TreeLSTM has only 1.2 times the number of weights in TreeLSTM (with the same number of hidden units). We did not count the number of parameters in the embedding models since these parameters are the same for all models.

Table 3.8 shows a comparison between mTreeLSTM and two other plausible methods for integrating relation information with TreeLSTM. In *addTreeLSTM*, a relation is treated as an additional node input in the TreeLSTM model; In *fullTreeLSTM*, the model corresponds to Equation 3.8, where each edge type has a separate transition matrix. Both models achieve better results than TreeLSTM, indicating the usefulness of relation information. While addTreeLSTM and fullTreeLSTM obtain comparable performances, mTreeLSTM outperforms both of them. It is also to note that the number of parameters of mTreeLSTM is much less than those of fullTreeLSTM.

To sum up, in this work, we present multiplicative TreeLSTM, an extension of existing tree-structured LSTMs to incorporate relation information between nodes in the parsed tree. Multiplicative TreeLSTM allows different compositional functions for child nodes, which makes it more expressive. In addition, we investigate how lexical semantic representation can be used with tree-structured LSTMs. Experiments on three common NLP tasks showed that multiplicative TreeLSTMs outperform conventional TreeLSTMs, illustrating the usefulness of relation information. Moreover, with AMR as backbone, tree-structured models can effectively handle long-range dependencies.

## 3.4 Improved Representation Learning for Question Answer Matching

Question answering (QA) is an important end-user task at the intersection of natural language processing and information retrieval. QA itself can be divided into factoid QA, which enables the retrieval of facts, and non-factoid QA, which enables finding of complex answer texts such as descriptions, opinions, or explanations as shown in Table 3.9. A typical architecture of QA systems is composed of two high level major components: i) question analysis and retrieval of candidate answers; ii) ranking and selecting of the most suitable answer. In this work, we focus on the latter component, i.e. answer selection (AS), and demonstrate the usefulness of representation learning for tackling this task. Formally, the AS task can be defined as follows: *Given a question and a pool of candidate answers, the goal is to select the positive answer*.

One main challenge of this task lies in the complex and versatile semantic relations that can be observed between questions and answers. While for factoid QA the task of answer selection may be largely cast as a textual entailment problem, for non-factoid QA what makes an answer better than another often depends on many factors. Different from many other matching tasks, the linguistic similarities between questions and answers may or may not be indicative for the good answers; depending on what the question is looking for, a good answer may come in different forms. Sometimes a correct answer completes the question precisely with the missing information, and in other scenarios, good answers need to elaborate part of the question to rationalize it, and so on. In other cases, the best answers can also be noisy and include extraneous information irrelevant to the question. In addition, while a good answer must relate to the question, they might not share common lexical units. For example, in the question in Table 3.9, the word "*companies*" is not directly mentioned in the answer. This issue may confuse simple word-matching systems.

| |
|---|
| **Question:** Are companies in California obliged to provide invoices? |
| **Ground-truth answer:** We run into this all the time with our EU clients. As far as I can tell, the only requirements when it comes to invoicing have to do with sales tax, which is determined at the state level, and only in the case that items are taxable. It seems that the service provided to you is not taxable and so there is no obligation under Californian law to provide what you need. |

**Table 3.9** An example of a question with a correct answer. The segments in the answer are related to the segments in the question by the same color.

These challenges consequently make the traditional models which are commonly based on lexical features [YCMP13, WM10, WSM07] less effective compared to deep learning based methods [FXG+15, WN15]. The neural models often follow the two step procedure: Firstly, representations of questions and answers are learned via a neural encoder such as long short-term memory (LSTM) networks or convolutional neural networks (CNN);

Secondly, these representations of questions and answers are composed by an interaction function to produce an overall matching score. In the first step, each word in a question or an answer sequence is first represented with a hidden vector and then all the hidden vectors are aggregated for sequence representations. These models have shown very successful results in the AS task, however they still suffer from an important issue. The answers can be very long and contain lots of words that are not related to the question at hand, especially in non-factoid QA; consequently, the resulting representation might be distracted by non-useful information.

Recent years, attention-based models are proposed to deal with this challenge and have shown great success in many tasks such as machine translation [BCB15, SVL14], machine reading comprehension [HKG+15] and textual entailments [RGH+15]. In the AS task, attention-based approaches aim to focus on segments within the candidate answer that are most related to the question [TdSXZ16, WLZ16]. The segments with a stronger focus are treated as more important and have more influence on the resulting representations. For example, in attention-based LSTM models [TdSXZ16] as shown in Figure 3.8, a weight is automatically generated for each word in the answer via an attention model, and the answer is represented as the weighted sum of the hidden vectors. Various attention mechanisms have been proposed in previous studies in which additive attention [BCB15] and multiplicative attention [RCW15] are the two most commonly used. While additive attention is associated with a multi-layer perceptron for computing attention weights, multiplicative attention uses inner product for the weight estimations. Though these attention mechanisms have shown promising results in answer selection, they do not make use of surrounding word context when calculating attention weights, which has been proved to enhance the performance of LSTM based QA [CHH+17]. To address this issue, we adopt another attention mechanism, i.e. *sequential attention* [BYRB17] in which an additional LSTM is added to compute a context-aware weight for each hidden vector. This mechanism helps generate more accurate answer representation regarding to the question.

A common characteristic of the previous attention-based approaches is that the question is represented by one feature vector and a round of attention is applied for learning the representation of the answer. However, in many cases different segments of the answer can be related to different parts of the question [PKVM17]. For example, in Table 3.9 the segment "*the only requirements when it comes to invoicing have to do with sales tax*" is relevant to "*provide invoices*" mentioned in the question, while "*there is no obligation under Californian law*" answers "*California obliged*" stated in the question. Consequently, using one feature vector pair for question answer matching may be not capable of capturing the complex semantic relations between questions and answers. This can lead to suboptimal results. Clearly, it is expected that the more aspects an answer covers the better the answer is. A good system should reflect this expectation.

In this work, we propose Multihop Attention Networks (MANs) to deal with this problem. MANs locate, via multiple steps of attention, answer segments that are relevant to different aspects of the question. As illustrated in Figure 3.7(b), the MAN first uses the question vector to deduce the answer vector in the first attention layer, then the question

vector is refined in the next step to learn the answer vector in the second attention layer. Each attention step gives a different attention distribution focusing on the segments that are relevant to one aspect of the question. Finally, we sum up the matching score in each step for scoring the answer. We perform experiments on both factoid QA and non-factoid QA datasets. Experimental results show that our proposed models obtain highly competitive results and outperform state-of-the-art approaches.

The main contributions of this work can be summarized as follows:

- We are the first to investigate the effectiveness of sequential attention mechanism for answers' attentive representations in answer selection.
- We propose Multihop Attention Networks which represent questions by multiple vectors and use multiple steps of attention for learning the representation of answers. By doing this, MANs can capture different semantic relations between questions and answers.
- We provide extensive experimental evidence of the effectiveness of our model on both factoid question answering and community-based question answering on different domains. Our proposed approach outperforms many other neural architectures on these datasets.

### 3.4.1 Related Literature

Our work is concerned with ranking question and answer pairs to select the most relevant answer for each question. Previous work on this task have primarily used feature engineering, linguistic tools, or external resources [YCMP13, WM10, WSM07]. In [YCMP13], Yih et al. constructed semantic features based on WordNet and paired semantically related words based on word semantic relations. In [WM10, WSM07], the answer selection problem was transformed to a syntactical matching between the question and answer parse trees. Some work tried to fulfill the matching using minimal edit sequences between dependency parse trees [HS10, YVDCBC13, SM13]. However, apart from relying on the availability of additional resources, the effort of feature engineering and the systematic complexity introduced by the linguistic tools, such models have limited performance and are outperformed by modern deep learning approaches [YHBP14, SM15].

Yu et al. [YHBP14] employed a convolutional neural network (CNN) for feature learning of QA pairs and subsequently applied logistic regression for prediction. Despite its simplicity, the approach outperforms all traditional approaches [YCMP13, SM13, YVDCBC13]. Another attractive quality of deep learning architectures is that features can be learned in an end-to-end fashion. Severyn et al. [SM15] presented a unified architecture that trains a convolutional neural network together with a multi-layer perceptron, in which features are learned while the parameters of the network are optimized for the task at hand. In addition to CNN, recurrent neural networks such as the long short-term memory (LSTM) networks are also very popular for learning sequence representation and have been widely

adopted in QA [WN15, TXZ15, TdSXZ16]. In [WN15], Wang and Nyberg incorporate stacked LSTMs to learn a joint feature vector of question and answer for classification. In [TXZ15], Tan et al. combined CNN and LSTM into a hybrid architecture which utilizes the advantages of both architectures. However, these approaches are outperformed by models with attention mechanism.

Recently, attention-based systems have shown very promising results on a variety of NLP tasks, such as machine translation [BCB15, SVL14], machine reading comprehension [HKG+15], text summarization [RCW15] and textual entailment [RGH+15]. Such models learn to focus their attention to specific parts of their input and most of them are based on a one-way attention, in which the attention is basically applied over one type of input based on another input (e.g. over target languages based on the source languages for machine translation, or over documents according to queries for reading comprehension). Most recently, several two-way attention mechanisms are proposed, where the information from two input items can influence the computation of each others representations. Rocktaschel et al. [RGH+15] develop a two-way attention mechanism including another one-way attention over the premise conditioned on the hypothesis, in addition to the one over hypothesis conditioned on premise. Santos et al. [dSTXZ16] and Yin et al. [YSXZ16] generate interactive attention weights on both inputs by assignment matrices. Yin et al. [YSXZ16] use a simple Euclidean distance to compute the interdependence between the two input texts, while Santos et al. [dSTXZ16] resort to attentive parameter matrices. Both types of attention show similar performances on AS [dSTXZ16, TXZ15], thus in this work we only use the one-way attention. However, unlike the previous work, our proposed models use multiple steps of attention instead of one attention step only.

Additive attention [BCB15] and multiplicative attention [RCW15] are the two most commonly used attention mechanisms. Self-attention or intra-attention is a special case of the additive attention mechanism. It relates elements at different positions from a single sequence by computing attention between each pair of tokens. In recent works, it has been shown effective in natural language inference [LSLW16], reading comprehension [HPQ17] and neural machine translation [VSP+17]. Sequential attention [BYRB17] is another type of attention mechanisms which uses an additional bi-directional RNN layer. This additional layer allows local alignment information to be used when computing the attentional score for each token. It has shown promissing results on reading comprehension [BYRB17]. In this work, we show that sequential attention can be well adopted for the AS task and obtain highly competitive results.

Our proposed MANs are also related to Dynamic Memory Networks (DMNs) [KIO+16] in the sense that we both use an iterative attention process instead of only one round of attention. However, each memory in DMNs depends on the memory in the previous step, aiming to narrow down their focus on individual facts or sentences in regard to the single question representation, while in MANs each attention step is applied independently for selecting the informative parts of answers relating to different aspects of the question.

### 3.4.2 Multihop Attention Networks

Although CNNs can be used for representation learning in the AS task, LSTMs have been shown to obtain better performances [TdSXZ16, dSTXZ16]. Hence, in this work we base our attention-based models on a variation of the LSTM model. We first describe the basic framework for answer selection based on LSTMs, called QA-LSTM [TdSXZ16]. Next we describe in detail different attention-based models that build on top of the QA-LSTM framework. After that, we present our Multihop Attention Networks.

**Long Short-Term Memory (LSTM).** LSTM networks [HS97] are a type of recurrent neural network that are capable of learning long term dependencies across sequences. Given an input sequence $X = (x_1, x_2, ..., x_n)$, where each $x_t$ is an $E$-dimension word vector ($x_t \in \mathbb{R}^E$), the LSTM returns a sequence embedding or hidden vector $h_t$ with a size $d$ ($h_t \in \mathbb{R}^d$) at every time step $t$ defined as follows:

$$\begin{aligned} i_t &= \sigma\left(W_i x_t + U_i h_{t-1} + b_i\right) \\ o_t &= \sigma\left(W_o x_t + U_o h_{t-1} + b_o\right) \\ f_t &= \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \\ u_t &= \tanh\left(W_u x_t + U_u h_{t-1} + b_u\right) \\ C_t &= i_t \odot u_t + f_t \odot C_{t-1} \\ h_t &= o_t \odot \tanh(C_t) \end{aligned} \tag{3.17}$$

where $W_*, b_*, U_*$ are the parameters of the LSTM network ($W_* \in \mathbb{R}^{d \times E}$, $U_* \in \mathbb{R}^{d \times d}$, $b_* \in \mathbb{R}^d$) and $* = \{i, o, f, u\}$ in which the input $i$, forget $f$ and output $o$ are three gates, and $C_t$ is the cell state. $\sigma$ is the sigmoid function and $\odot$ denotes element-wise multiplication. Here, we omit the technical details of LSTM which can be found in many related works.

Single-direction LSTMs suffer from the weakness of not making use of the contextual information provided by future tokens. Bidirectional LSTMs (biLSTMs) use both the previous and future context by processing the sequence in two directions, and generate two sequences of output vectors. The output for each token is the concatenation of the two vectors from both directions, i.e. $h_t = \overrightarrow{h_t} \parallel \overleftarrow{h_t}$. The output of biLSTM layer is a sequence of hidden vectors $H \in \mathbb{R}^{L \times 2d}$ where $L$ is the maximum sequence length and $d$ is the dimensional size of LSTM.

**LSTM for Answer Selection.** The basic LSTM-based framework for answer selection (QA-LSTM) introduced by Tan et al. [TdSXZ16] is shown as in Figure 3.7(c) (without the attention layer). Given an input pair $(q, a)$, where $q = (q_1, ..., q_l)$ is a sequence of word indices for question and $a = (a_1, ..., a_m)$ is a sequence of word indices for candidate answer, the word embeddings (WEs) of both $q$ and $a$ are first retrieved by passing the sequences of word indices through a look-up layer. The parameters of this layer are $W \in \mathbb{R}^{V \times E}$ where $V$ is the size of vocabulary and $E$ is the dimensionality of the word embeddings. We initialize $W$ with pretrained word embeddings which is inline with previous works [YHBP14, TdSXZ16].

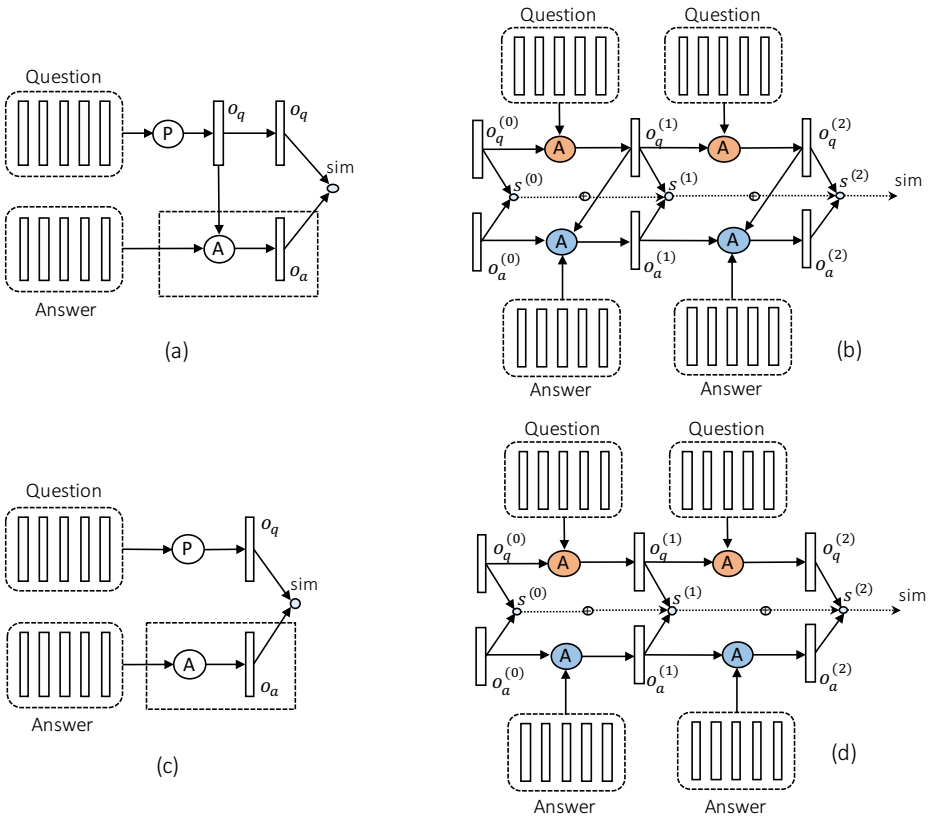



**Figure 3.7** Traditional attention-based networks: (a) Interactive attention network; (c) Self-attention network; and our proposed MANs: (b) Multihop interactive attention network; (d) Multihop self-attention network. **P**: pooling layer, **A**: attention layer

A biLSTM is then applied separately over the two sequences of WEs creating hidden vectors for the question and answer, i.e. $h_q(t) = \text{LSTM}(\overrightarrow{h_q}(t-1), q_t) \parallel \text{LSTM}(\overleftarrow{h_q}(t+1), q_t)$ and $h_a(t) = \text{LSTM}(\overrightarrow{h_a}(t-1), a_t) \parallel \text{LSTM}(\overleftarrow{h_a}(t+1), a_t)$. Subsequently, the final representations $o_q$ and $o_a$ for question and answer, respectively, can be taken by max or mean pooling over all the hidden vectors or the last hidden vector. As discussed in [FXG+15, TXZ15], sharing the same network parameters is significantly better than using separate question and answer parameters, and converges much faster. Therefore, we follow the same procedure by using the same network parameters for processing questions and candidate answers.

Finally, a cosine similarity $sim(q, a)$ is defined to score the input pair $(q, a)$ and the hinge loss function is used as training objective.

$$\mathcal{L} = \max\{0, M - sim(q, a_+) + sim(q, a_-)\} \tag{3.18}$$

where $a_+$ is a ground truth answer, $a_-$ is an incorrect answer randomly chosen from the entire answer space, and $M$ is the margin which controls the extent of discrimination be-

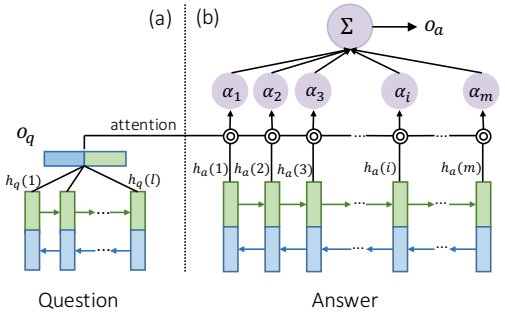



**Figure 3.8** (a) The question vector representation and (b) The attention mechanism for answer vector generation

tween positive QA pairs and corrupted QA pairs. We treat any question with more than one ground truth as multiple training examples. During training, for each question we randomly sample $N$ negative answers, but only use the one with the highest $\mathcal{L}$ to update the model similar to [TdSXZ16, RHL16].

**Attention Mechanisms.** The aforementioned QA-LSTM is basically a siamese network [CHL05] which might fail to notice a potential issue. The answers might be extremely long and contain lots of words that are not related to the question at hand, especially in non-factoid question answering. For example, in Table 3.9 the first sentence "*we run into this all the time with our EU clients*" does not relate directly to the question. Even if advanced neural networks are exploited, the resulting representation might still be distracted by non-useful information. Thus, a number of attention-based models for the answer vector generation have been proposed in order to alleviate this weakness by dynamically aligning the more informative parts of answers to the question. Conceptually, attention mechanisms give more weight to certain words which have more influence on the resulting representation. In the AS task the expectation is that words in the candidate answer that are more important with regard to the input question should receive larger weights. Most previous works such as [TdSXZ16, RHL16] proceed as follows: the input question is represented by a vector $o_q$ using last, max or average pooling and an attention model is used over a sequence of hidden vectors to learn the representation of the answer $o_a$ as shown in Figure 3.8. An attention mechanism which takes into account the question vector for computing the attention weights in learning the answer representation, is called as an interactive attention mechanism. In contrast, an attention mechanism which is employed only on the candidate answer, is considered as a self-attention or intra-attention mechanism. One of the advantages of the intra-attention mechanism is that questions and answers can be embedded into a joint vector space without being paired, so that arbitrary question and answer vectors in that space are directly comparable.

Let $H_a = \{h_a(1), h_a(2), ..., h_a(m)\}$ denote the hidden vectors of the answer after passing through the biLSTM layer. To produce the final representation of the answer, instead of using the last hidden vector or average or max pooling, an additional attention layer is

used as follows:

$$\begin{aligned} \alpha_t &\propto f_{attention}\left(\tilde{o}_f, h_a(t)\right) \\ o_a &= \sum_t \alpha_t h_a(t) \end{aligned} \tag{3.19}$$

where $h_a(t)$ is the hidden vector of the answer at time $t$. When the interactive attention mechanism is used, $\tilde{o}_f$ is often equal to the question representation $o_q$ as shown in Figure 3.7(a). When the intra-attention mechanism is used, $f_{attention}$ only depends on $h_a(t)$ as in Figure 3.7(c). Next, we describe in detail the different implementations of $f_{attention}$ function.

*MLP Attention:* Additive attention (or multi-layer perceptron attention) [BCB15] is one of the most commonly used attention mechanisms. It is first used for answer selection by Tan et al. [TdSXZ16]. In [TdSXZ16], the attention function $f_{attention}$ is computed by a multi-layer perceptron network as follows:

$$\begin{aligned} m(t) &= \tanh\left(W_a h_a(t) + W_q o_q\right) \\ f_{attention}\left(\tilde{o}_f, h_a(t)\right) &= \text{softmax}\left(w_m^\top m(t)\right) \end{aligned} \tag{3.20}$$

where $W_a, W_q$ are attentive weight matrices and $w_m$ are attentive weight vector. The size of the matrices and vector are often equal to the size of input vectors $h_a(t)$ and $o_q$.

*Bilinear Attention:* This is another commonly used attention mechanism. For example, Chen et al. [CBM16] found it effective in machine reading comprehension, Rush et al. [RCW15] used it in abstractive summarization. Santos et al. [dSTXZ16] used this attention mechanism for AS task. In contrast to the additive attention, this attention mechanism makes use of a bilinear term instead of using a tanh layer to estimate the attention function $f_{attention}$:

$$f_{attention}\left(\tilde{o}_f, h_a(t)\right) = \text{softmax}_t\left(o_q^\top W_s h_a(t)\right) \tag{3.21}$$

where $W_s$ is a network parameter.

*Sequential Attention:* The previous approaches to attention select words with only very indirect consideration of their context, Brarda et al. [BYRB17] address this issue by taking into account explicit context sensitivity for computing the attention scoring function $f_{attention}$. Specifically, instead of producing a single value of $f_{attention}$ for each word in the answer by using a bilinear term as the bilinear attention, a vector $\gamma_t$ is defined as

$$\gamma_t = o_q \odot h_a(t) \tag{3.22}$$

where $\odot$ is element-wise multiplication. The vector $\gamma_t$ is then fed into a new biLSTM layer to get the hidden attention $\eta_t$ vector representation: $\eta_t = \text{LSTM}(\overrightarrow{\eta}_{t-1}, \gamma_t) \parallel \text{LSTM}(\overleftarrow{\eta}_{t+1}, \gamma_t)$. Finally, the attention function $f_{attention}$ is computed as

$$f_{attention}\left(\tilde{o}_f, h_a(t)\right) = \text{softmax}_t\left(1^\top \eta_t\right) \tag{3.23}$$

Chen et al. [CHH+17] showed that utilizing context information can enhance the performance of LSTM based QA. Therefore, in this work we aim to investigate the effectiveness of sequential attention in answer selection. Experimental results show that sequential attention can be well adapted for the AS task and outperform other attention mechanisms on different QA datasets.

*Self-Attention:* In contrast to aforementioned attention mechanisms where the question vector $o_q$ is used to learn the representation of the answer, in this attention mechanism the answer is autonomously embedded into the embedding space without being paired with the question, so that arbitrary question and answer vectors in the space are directly comparable. Generally, self-attention relates different positions of a single sequence in order to compute the final representation of the sequence. This has been successfully employed in a variety of tasks including reading comprehension, abstractive summarization and textual entailment [LFdS+17]. In the context of answer selection, $f_{attention}\left(\tilde{o}_f, h_a(t)\right)$ can be estimated merely based on $h_a(t)$ as follows:

$$\begin{aligned} s(t) &= \tanh\left(W_s h_a(t) + b_s\right) \\ f_{attention}\left(\tilde{o}_f, h_a(t)\right) &= \text{softmax}\left(w_s^\top s(t)\right) \end{aligned} \tag{3.24}$$

where $W_s, b_s$ and $w_s$ are attention parameters.

**Multihop Attention Networks.** In many cases, the semantic relations between a question and an answer can be very complex. Similar to [PKVM17], we observe that different parts of the answer can relate to different aspects or intentions addressed by the question. For example, in Table 3.9, the question *"are companies in California obliged to provide invoices"* refers to two aspects *invoices* and *California law* and each aspect is covered by different parts of the answer. Consequently, using single vectors for the question and answer representations might not be able to uncover their complex semantic relations. This can lead to suboptimal results.

In this work, we propose Multihop Attention Networks (MANs) to tackle this problem. Our models are represented in Figure 3.7(b) and 3.7(d). Unlike existing models [TdSXZ16, dSTXZ16], we do not compress the question to a single representation, but instead use multiple vectors for the question representation. Each question vector is then used to match with the answer representation which is learned via an attention layer. Specifically, given a question and a candidate answer, the model first reads the question and the answer using a biLSTM layer. Then, it deploys an iterative matching process to uncover the semantic relations between the question and the answer. In this phase, it first attends to some parts of the question, then finds their corresponding matches by attending to the answer. In the next step, it gives more attention to other parts of the question and searches for their matching parts in the answer. After a fixed number of iterations, the model uses the sum of the matching scores of each step to rank the question-answer pair.

Let $H_q = \{h_q(1), ..., h_q(l)\}$ denote the hidden vectors of the question after passing through the biLSTM layer. To obtain the representation of the question, one of three following mechanisms is usually used: last, mean, or max pooling. Last pooling takes the last

vector $h_q(l)$, mean pooling averages all vectors and max pooling takes the element-wise maximum of $H_q$. In [LFdS+17], Lin et al. proposed self-attention mechanism to replace the max pooling or averaging step. In this work, we adapt this mechanism with some modifications for creating different question vector representations. Specifically, in each step $k$, the question representation $o_q^{(k)}$ is computed as follows:

$$\begin{aligned} s_t^{(k)} &= \tanh\left(W_q^{(k)} h_q(t)\right) \odot \tanh\left(W_m^{(k)} m_q^{(k-1)}\right) \\ \alpha_t^{(k)} &= \text{softmax}\left({w_s^{(k)}}^\top s_t^{(k)}\right) \\ o_q^{(k)} &= \sum_t \alpha_t^{(k)} h_q(t) \end{aligned} \tag{3.25}$$

where $W_q^{(k)}, W_m^{(k)}$ and $w_s^{(k)}$ are network parameters, $m_q^{(k)}$ is a separate memory vector for guiding the next attention step. It is recursively updated by

$$m_q^{(k)} = m_q^{(k-1)} + o_q^{(k)} \tag{3.26}$$

The initial memory vector $m_q^{(0)}$ is defined based on the context vector $o_q^{(0)}$ where

$$o_q^{(0)} = \frac{1}{l} \sum_t h_q(t)$$

In each step, the question is represented by a vector $o_q^{(k)}$ which focuses specifically on some aspects of the question. The vector $o_q^{(k)}$ is then used as input for the attention models described in the previous section to extract the answer representation $o_a^{(k)}$. After that, we compute the similarity between question and answer vectors by their cosine similarity. This similarity score reflects on how the answer relates to the corresponding parts of the question. After performing $K$ matching steps, the final similarity between the given question and answer becomes

$$sim(q, a) = \sum_k^K \cos(o_q^{(k)}, o_a^{(k)}) \tag{3.27}$$

The overall architecture of this model when $K = 2$ is shown in Figure 3.7(b). Figure 3.7(d) presents MAN with using self-attention mechanism for the answer vector generation. In this case, we use two separate attention models described in Equation 3.25, one model for questions and another for answers. After each step, the same procedure is applied as implied by Equation 3.27. It is important to note that separate attention models are applied to questions and answers but the same attention parameters are used for questions (answers) in different steps. Therefore, our network parameters are comparable to the models with a single attention layer [TdSXZ16, dSTXZ16] but we outperform the latter models on the datasets tested.

### 3.4.3 Experimental Setup

**Datasets**: To evaluate the proposed approaches, we conduct an empirical evaluation based on three popular and well-studied benchmark datasets for both factoid and non-factoid question answering. In addition, we use another newly released dataset for the financial domain. These four datasets cover different domains and exhibit different characteristics:

- TREC-QA - This is a benchmark dataset created by Wang et al. [WSM07] based on Text REtrieval Conference (TREC) QA track (8-13) data. The dataset contains a set of factoid questions, where candidate answers are limited to a single sentence. To enable direct comparison with the previous work, we follow the approach of train/dev/test questions selection from [WN15], in which all questions with only positive or negative answers are removed. In total, we have 1162 training questions, 65 development questions and 68 test questions. The maximum number of tokens for questions and answers are set to 11 and 60, respectively, the length of the vocabulary $|V|$=55060 and for each question there are 38 candidate answers on average.

- WikiQA - This is a recent popular benchmark dataset for open-domain question answering, based on factual questions from Wikipedia and Bing search logs. For each question, Yang et al. [YYM15] selected Wikipedia pages and used sentences in the summary paragraph as candidates, which are then annotated on a crowdsourcing platform. We follow the same preprocessing steps as Yang et al., where questions with no correct candidate answers are excluded and answer sentences are truncated to 40 tokens. In total, we end up with 873 training questions, 126 development questions and 243 test questions. Since there are only few negative answers for each question in WikiQA, we extend it by randomly selecting a bunch of negative candidates from the answer pool.

- InsuranceQA - This is a recently released large-scale non-factoid QA dataset from the insurance domain created by Feng et al. [FXG+15]. In this work we use the first version of the dataset. The dataset is already divided into a training set, a validation set, and two test sets, in which a question may have multiple correct answers and normally the questions are much shorter than the answers. The average length of questions and answers in tokens are 7 and 95, respectively. Such difference imposes additional challenges for the answer selection task. For each question in the development and test sets, there is a set of 500 candidate answers, which include the ground-truth answers and randomly selected negative answers.

- FiQA - This is a new non-factoid QA dataset from the financial domain which has been recently released for WWW 2018 Challenges.[5] The dataset is built by crawling Stackexchange, Reddit and StockTwits in which part of the questions are opinionated, targeting mined opinions and their respective entities, aspects, sentiment

[5]https://sites.google.com/view/fiqa/home

| Dataset | Train | Dev | Test | Avg. Q | Avg. A | Avg. Cand |
|---|---|---|---|---|---|---|
| TREC-QA | 1162 | 65 | 68 | 8 | 28 | 38 |
| WikiQA | 873 | 126 | 243 | 6 | 25 | 9 |
| InsuranceQA | 12887 | 1000 | 1800x2 | 7 | 95 | 500 |
| FiQA | 5999 | 323 | 324 | 11 | 135 | 500 |

**Table 3.10** The statistics of the four employed answer selection datasets. For WikiQA and TREC-QA we remove all questions that have no right or wrong answers.

polarity and opinion holder. We minimally preprocess the data only performing tokenization and lowercasing all words. To reduce the size of resulting vocabulary, we remove all rare words which occur less than 5 times. In this dataset questions and answers are longer than in other datasets, which will consequently bring extra challenges. The maximum number of tokens for questions and answers are set to 20 and 150 respectively. Following the setup for other datasets, we split this dataset into training, development and test sets as shown in Table 3.10. For each question in the development and test sets, we construct the answer pools by including the correct answer(s) and randomly selected candidates from the complete set of unique answers. Finally we have 500 candidate answers for each question.

Table 3.10 presents some statistics about the datasets, including the number of questions in each set, average length of questions and answers as well as average number of candidate answers in the development and test sets.

**Employed Baselines**: For all datasets, we report performances of the basic matching model QA-LSTM and the models with a single attention layer described in Section 3.4.2 including MLP-LSTM, Bilinear-LSTM, Self-LSTM and Sequential-LSTM. Furthermore, we also introduce other baselines for each dataset separately.

- TREC-QA - The key competitors of this dataset are the CNN model of Severyn and Moschitti [SM15], the Attention-based Neural Matching model [TdSXZ16, dSTXZ16] and the RNN with Positional Attention proposed by Chen et al. [CHH+17]. In addition, due to the long standing nature of this dataset, we also report works based on traditional feature engineering approaches [WSM07, HS10].
- WikiQA - The competitors of this dataset include the Paragraph Vector (PV) and PV + Cnt models [YCMP13], CNN + Cnt model [YHBP14] which are reported in the original WikiQA paper [YYM15]. Furthermore, we report additional strong baselines including AP-CNN and AP-LSTM [dSTXZ16], ABCNN [YSXZ16] and RNN-POA [CHH+17]. We also report the Pairwise Ranking MP-CNN model [RHL16].
- InsuranceQA - The key competitors of this dataset are the CNN-based ARC-I/II architecture by Feng et al. [FXG+15], QA-LSTM from [TdSXZ16] along with AP-

LSTM which are attentive pooling improvements of the former and Inner attention-based RNN [WLZ16].

- FiQA - For this dataset, we reimplemented QA-LSTM [TdSXZ16] and different attention mechanisms on top of QA-LSTM for comparison.

We denote our proposed models as *Multihop-MLP-LSTM*, *Multihop-Bilinear-LSTM*, *Multihop-Sequential-LSTM* and *Multihop-Self-LSTM* which are MANs based on additive attention, bilinear attention, sequential attention and self-attention, respectively, for learning answers' attentive representations.

**Hyperparameters and Training**: Here we describe the key evaluation protocol and metrics as well as implementation details of our experiments.

*Evaluation Metrics:* For the evaluation protocols we follow the prior work. Specifically, in TREC-QA and WikiQA we use the Mean Reciprocal Rank (MRR) and Mean Average Precision (MAP) metrics which are commonplace in IR and QA research. On the other hand, InsuranceQA and FiQA evaluate on Precision@1 (P@1) which is determined based on whether the top predicted answer is the ground truth. For all competitor methods, we report the performance results from the original paper.

*Implementation Details and Hyperparameters:* The models are implemented in Pytorch. The model parameters are optimized using Adam [BK15] optimizer with a learning rate of $0.001$. A batch size of $100$ are used for all datasets. The parameters are regularized with a per-minibatch L2 regularization strength of $10^{-5}$ and a dropout of $d = 0.3$ is also applied to prevent overfitting. The hidden layer size of LSTM models are the same as in previous works for a fair comparison. Specifically, for TREC-QA and InsuranceQA the sizes are set to 300 and 141 respectively as in [TdSXZ16]; the number for WikiQA is 141 as in [dSTXZ16]. For FiQA, we tried different numbers and found out that the size of 512 yields the best results. We tried different margins $M$ in the hinge loss function and finally fixed the margin to $M = 0.2$. A number of negative answers $N = 50$ was used during training. The number of attention steps $K$ is tuned amongst $\{1, 2, 3\}$ and we also experimented with the set of three vectors by using last, max and average pooling as different representations for the questions. We initialized the word embeddings with $300$-dimensional Glove vectors [PSM14] trained on 840 billion words. Embeddings for words not present in the Glove vectors are randomly initialized with each component sampled from the uniform distribution over $[-0.25, 0.25]$. The word embeddings are also part of the parameters and are optimized during training. Since sequences within a mini-batch have different lengths, we use a mask matrix to indicate the real length of each sequence. We trained all models for a maximum of 40 epochs. We take MAP scores for TREC-QA and WikiQA and P@1 scores for InsuranceQA and FiQA on the development set at every epoch and save the parameters of the network for the top three models. We report the best test score from the saved models. All experiments were conducted on a Linux machine with Nvidia GTX Ti 1080 GPU (12GB RAM). The code to reproduce the reported results and FiQA splits are publicly available at https://github.com/namkhanhtran/nn4nqa.

| Model | TREC-QA | | WikiQA | |
|---|---|---|---|---|
| | **MAP** | **MRR** | **MAP** | **MRR** |
| PV + Cnt (Yih et al. [YCMP13]) | - | - | 0.599 | 0.609 |
| CNN + Cnt (Yu et al. [YHBP14]) | - | - | 0.652 | 0.665 |
| AP-LSTM (Santos et al. [dSTXZ16]) | - | - | 0.670 | 0.684 |
| ABCNN (Yin et al. [YSXZ16]) | - | - | 0.692 | 0.710 |
| Rank MP-CNN (Rao et al. [RHL16]) | - | - | 0.701 | 0.718 |
| Wang et al. [WSM07] | 0.603 | 0.685 | - | - |
| Heilman & Smith [HS10] | 0.609 | 0.692 | - | - |
| Wang & Nyberg [WN15] | 0.713 | 0.791 | - | - |
| CNN (Severyn & Moschitti) [SM15] | 0.746 | 0.808 | - | - |
| AP-LSTM (Tan et al.) [TdSXZ16] | 0.753 | 0.830 | - | - |
| AP-CNN (Santos et al.) [dSTXZ16] | 0.753 | 0.851 | 0.689 | 0.696 |
| RNN-POA (Chen et al. [CHH+17]) | 0.781 | 0.851 | 0.721 | 0.731 |
| QA-LSTM | 0.737 | 0.810 | 0.654 | 0.665 |
| MLP-LSTM | 0.764 | 0.839 | 0.686 | 0.695 |
| Bilinear-LSTM | 0.755 | 0.832 | 0.677 | 0.686 |
| Self-LSTM | 0.759 | 0.830 | 0.693 | 0.704 |
| Sequential-LSTM | 0.797 | 0.865 | 0.702 | 0.715 |
| Multihop-MLP-LSTM | 0.768 | 0.849 | 0.703 | 0.712 |
| Multihop-Bilinear-LSTM | 0.788 | 0.864 | 0.715 | 0.725 |
| Multihop-Self-LSTM | 0.771 | 0.864 | 0.702 | 0.710 |
| Multihop-Sequential-LSTM | **0.813** | **0.893** | **0.722** | **0.738** |

**Table 3.11** Experimental results on TREC-QA and WikiQA. Baselines for TREC-QA and WikiQA are reported in the first group. The second group shows the performance of models with a single attention layer. We report the performance of MANs in the last group.

### 3.4.4 Experimental Results

In this section, we present our empirical results on all datasets. For all reported results the best result is in boldface.

**TREC-QA**: Our results on TREC-QA dataset is summarized in Table 3.11. Firstly, we observe that all attention-based models outperform the basic matching model QA-LSTM by large margins. Second, the model based on sequential attention mechanism obtains a clear performance gain of around 3% on MAP/MRR against the model with additive or multiplicative attention mechanism. Compared to these models, Self-LSTM achieves comparable results though it does not use any query information for extracting answer representation. It also performs better than QA-LSTM, which indicates that self-attention

mechanism can give better representations than simply max or average pooling method.

Furthermore, Table 3.11 shows that MANs outperform all models with only one attention layer. When using the same attention mechanism, the averages increase over the baselines with a single attention layer are 2% – 3% in terms of MAP/MRR. Specifically, Multihop-Sequential-LSTM gains an improvement of 1.6% on MAP and 2.8% on MRR compared to Sequential-LSTM. Similarly, Multihop-Bilinear-LSTM obtains 3.3% and 3.2% improvements in terms of MAP and MRR respectively. Multihop-MLP-LSTM also shows some degree of improvement compared to MLP-LSTM. Based on self-attention mechanism, MAN outperforms the model with one attention layer by 1.2% on MAP and 3.4% on MRR. Overall, Multihop-Sequential-LSTM obtains the best results on TREC-QA dataset and surpasses the strong baseline RNN-POA [CHH+17] by 3.2% on MAP and 4.2% on MRR.

**WikiQA**: Table 3.11 reports the experimental results on WikiQA. First, we observe that MAN-based models outperform the models with a single attention layer. Multihop-Sequential-LSTM outperforms Sequential-LSTM by 2% in terms of MAP and 2.3% in terms of MRR. Multihop-Bilinear-LSTM shows improvements of 3.8% on MAP and 3.9% on MRR compared to Bilinear-LSTM. Multihop-MLP-LSTM and Multihop-Self-LSTM also perform slightly better than MLP-LSTM and Self-LSTM, respectively. Overall, Multihop-Sequential-LSTM achieves the best results on WikiQA and shows some degree of improvement compared to the strongest baseline RNN-POA [CHH+17].

**InsuranceQA**: Table 3.12 reports the experimental results on InsuranceQA. Our proposed approaches achieve highly competitive performances on this dataset, where Multihop-Sequential-LSTM obtains the best P@1 performance overall. Our best model surpasses the strong baseline IARNN-Gate on both test sets. Although most MAN-based models show some degree of improvement compared to the models with a single attention layer, applying one step of attention seems to be sufficient on this dataset. Interestingly, Self-LSTM performs quite well on InsuranceQA dataset even outperforms some interactive attention-based models. Multihop-Self-LSTM does not show any improvement against Self-LSTM. In addition, Sequential-LSTM again shows better results than MLP-LSTM and Bilinear-LSTM on this dataset.

**FiQA**: The results of the proposed models are shown in Table 3.13. On this new dataset we observe similar behaviours to other datasets. Firstly, attention-based models outperform the basic matching model QA-LSTM. MAN-based models perform better than the models with a single attention layer, in which Multihop-Sequential-LSTM obtains the best performance overall. More specifically, Multihop-Sequential-LSTM improves Sequential-LSTM by 4.5% in terms of P@1 while Multihop-Bilinear-LSTM shows an improvement of 2.8% on P@1 against Bilinear-LSTM. Multihop-MLP-LSTM indicates 1% enhancement on P@1 compared to MLP-LSTM whereas Multihop-Self-LSTM also increases over Self-LSTM by 1.3% in terms of P@1. Amongst interactive attention-based models Sequential-LSTM outperforms MLP-LSTM and Bilinear-LSTM. Compared to these models, Self-

| Model | Test1 | Test2 |
|---|---|---|
| CNN (Feng et al. [FXG+15]) | 0.628 | 0.592 |
| CNN with GESD (Feng et al. [FXG+15]) | 0.653 | 0.610 |
| AP-LSTM (Tan et al. [TdSXZ16]) | 0.690 | 0.648 |
| IARNN-Gate (Wang et al [WLZ16]) | 0.701 | 0.628 |
| QA-LSTM | 0.643 | 0.617 |
| MLP-LSTM | 0.693 | 0.648 |
| Bilinear-LSTM | 0.689 | 0.658 |
| Self-LSTM | 0.699 | 0.653 |
| Sequential-LSTM | 0.702 | 0.665 |
| Multihop-MLP-LSTM | 0.695 | 0.655 |
| Multihop-Bilinear-LSTM | 0.694 | 0.662 |
| Multihop-Self-LSTM | 0.682 | 0.648 |
| Multihop-Sequential-LSTM | **0.705** | **0.669** |

**Table 3.12** Experimental results on InsuranceQA. Baselines for InsuranceQA are reported in the first group. The second group shows the performance of models with a single attention layer. We report the performance of MANs in the last group.

LSTM shows highly competitive results.

Overall, we summarize the key findings of our experiments:

- Similar to previous work, we observe that attention-based models perform significantly better than the basic matching model.
- Multihop Attention Networks are better in capturing the complex semantic relations between questions and answers and outperform the models with only one attention layer.
- Sequential attention can be well adopted for the AS task and gains considerably improvements compared to traditional attention mechanisms.
- Self-attention can produce better representations than simply max or average pooling method and obtain competitive results on the AS task.

### Effect of Attention Steps

Table 3.14 shows the influence of the number of steps on performance on FiQA dataset. Multihop-Sequential-LSTM* is the model where we consider the vectors returned by using max, mean and last pooling as different representations for the questions. Overall, Multihop-Sequential-LSTM performs better than Sequential-LSTM in which with $K = 2$

| Model | MAP | MRR | P@1 |
|---|---|---|---|
| QA-LSTM | 0.433 | 0.566 | 0.469 |
| MLP-LSTM | 0.497 | 0.616 | 0.509 |
| Bilinear-LSTM | 0.492 | 0.606 | 0.506 |
| Self-LSTM | 0.493 | 0.608 | 0.509 |
| Sequential-LSTM | 0.504 | 0.621 | 0.522 |
| Multihop-MLP-LSTM | 0.498 | 0.613 | 0.519 |
| Multihop-Bilinear-LSTM | 0.507 | 0.631 | 0.534 |
| Multihop-Self-LSTM | 0.488 | 0.619 | 0.522 |
| Multihop-Sequential-LSTM | **0.529** | **0.655** | **0.567** |

**Table 3.13** Experimental results on FiQA. The first group shows the performance of models with a single attention layer. We report the performance of MANs in the second group.

| Model | MAP | MRR | P@1 |
|---|---|---|---|
| Sequential-LSTM | 0.504 | 0.621 | 0.522 |
| Multihop-Sequential-LSTM* | 0.514 | 0.636 | 0.543 |
| Multihop-Sequential-LSTM (K=1) | 0.527 | 0.649 | 0.561 |
| Multihop-Sequential-LSTM (K=2) | **0.529** | **0.655** | **0.567** |
| Multihop-Sequential-LSTM (K=3) | 0.523 | 0.644 | 0.546 |

**Table 3.14** Effect of different number of attention steps on FiQA

Multihop-Sequential-LSTM obtains the best performance. This might be due to the fact that the questions are rather short and often express two different aspects at most.

### A Case Study

Figure 3.9 depicts the heat map of a test question from FiQA that was correctly answered by Multihop-Sequential-LSTM. The stronger the color of a word in the question (answer), the larger the attention weight of that word. As we can see in the figure, in the first step Multihop-Sequential-LSTM puts more focus on some segments of the question and the parts of the answer that have some interactions with the question segments. In the second step, the multihop attention network gives more attention to other segments of the question and consequently some other parts of the answer get more attention.

To sum up, in this work, we present Multihop Attention Networks for question answer selection. Our proposed MANs use multiple vectors which focus on different parts of a question to represent the overall semantics of the question and then apply multiple steps of attention to learn representations for the candidate answers. In addition, we also show that sequential attention mechanism can be well adapted for this task.

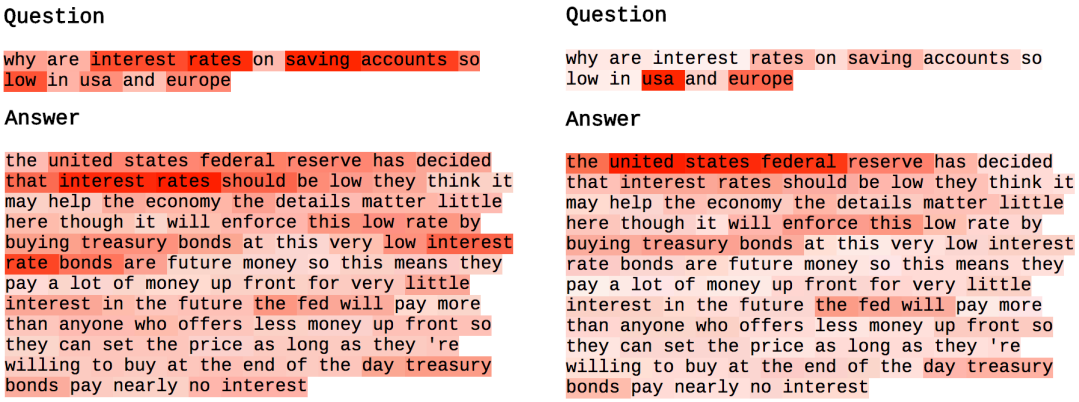



**Figure 3.9** Attention heat map from Multihop-Sequential-LSTM (K=2) for a correctly selected answer.

## 3.5 Chapter Summary

In this chapter, we addressed the problem of document representation learning and proposed different approaches for tackling the research questions which were discussed in the previous chapter.

Firstly, to improve learned topics in the topic-based representation of documents in small collections (**RQ1.1**), we described a topic cropping approach which automatically tailors additional domain-specific documents with similar topical content and then map topics learned from this larger collection to the working collection. By integrating the automatic evaluation of topic quality we took a first step towards a self-optimizing process of selecting parameters for topic cropping in different settings.

Secondly, to improve the distributed representation of documents (**RQ1.2**), we presented multiplicative tree-structured LSTMs, which are capable of incorporating both syntactic and semantic information from text to tree-structured LSTM models and proved the usefulness of the proposed models in various downstream applications. In contrast to traditional approaches, the proposed models employ not only word information but also relation information between words. Hence, it is more expressive as different combination functions can be applied for each word.

Finally, we demonstrated that the distributed representation of documents can be further improved by using attention mechanisms (**RQ1.3**). We illustrated the usefulness of such representation in the application of question answering by proposing Multihop Attention Networks. Unlike previous approaches, the proposed MANs use multiple vectors which focus on different aspects of a question to represent the overall semantics of the question and apply multiple steps of attention to learn representations for candidate answers.

# 4 Bridging Temporal Context Gaps for Supporting Document Interpretation

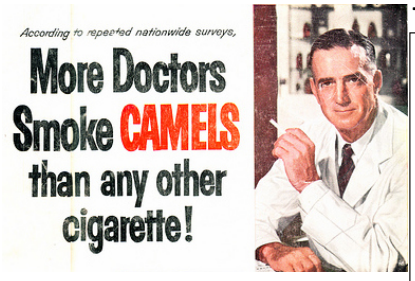
*"You shall know a word by the company it keeps."* J. R. Firth, 1957

Without context words have no meaning and the same is true for documents, in that often a wider context is required to fully interpret the information they contain. For example, a photo is practically useless if you do not know who the people portrayed in it are, and likewise a document that refers to the president of the US is of little use without knowing who held the job at the time the document was written. This becomes even more important when considering the long term preservation of documents as not only is human memory faliable, but over long periods the people accessing the documents will change as will their understanding and knowledge of the world.

In the simplest form, context is external information which is either required or aids in understanding an "items". This extra information may just be a few words to help disambiguate a single word [Fir57] or it might be large quantities of extra information [TCKN15a]. Time also plays an important factor in what context is needed. Terms that are unambiguous in todays world can easily fall out of use or their use change over time. A good example of this is the word "computer" which used to refer to a person employed to do computations, a meaning which many people today are unaware of. This means that while many concepts might not need to be explained now, their meaning would need to be captured as context information and preserved so that they could be unambiguously interpreted in the future.

As with any complex task there are many ways in which solving the contextualization problem can be approached. For example, approaches will clearly differ depending upon the type of document being processed; text versus images, long documents versus short documents. It is also possible, of course, to have multiple competing and/or complimentary approaches that can be tried on the same set of documents. This chapter is devoted to the approaches to contextualization focusing on text documents.

Understanding document, which was written some time ago, can be compared to trans-




**Time-aware Contextualization**

Prior to 1964, many of the cigarette companies advertised their brand by claiming that their product did not have serious health risks. A couple of examples would be "Play safe with Philip Morris" and "More doctors smoke Camels". Such claims were made both to increase the sales of their product and to combat the increasing public knowledge of smoking's negative health effects.

**Figure 4.1** Camel advertisement and its context.

lating a text from another language. Complete interpretation requires a mapping, in this case, a kind of time-travel translation between present context knowledge and context knowledge at time of text creation. In this chapter, we look in detail at challenges allowing us to develop a framework in which context information can be collected for the later interpretation of documents.

## 4.1 Introduction

Reading a current news article about your own country is typically straightforward as your own world knowledge allows you to unambiguously understand the text. Things are different if you read an article, for example, from the 60s or the 70s as can be found in news archives such as the New York Times Archive.[1] In this chapter, we are especially interested in *time-aware re-contextualization*, where explicit context information is required to bridge the gap between our current understanding of the world and the situation at the time of content creation. This includes changes in background knowledge, the societal and political situation, language, technology, or simply the passage of time leading readers to forget.

The importance of time-aware re-contextualization is well illustrated by the advertisement poster from the 1950s in Figure 4.1. From today's perspective it is more than surprising that doctors would be recommending smoking. It can, however, be understood from the context information at the right side of Figure 4.1, which has been extracted from the Wikipedia article on tobacco advertising.

As another example, if we see the 1950's advertisement shown in Figure 4.2, we might find trouble in understanding the point it makes, or we are outraged about the (not so) implicit message given that women are too weak or too stupid to open a ketchup bottle. At first glance, it seems difficult to imagine how this can be used for advertising a household product. However, if we look into the context information on the right side of the figure, we might understand that the advertisement follows the gender stereotype of a housewife

[1] http://catalog.ldc.upenn.edu/LDC2008T19


**Time-aware Contextualization**

In 1960s, the world of American women was limited in almost every respect, from family life to the workplace. A woman was expected to follow one path: to marry in her early 20s, start a family quickly, and devote her life to homemaking. [...] They were legally subject to their husbands via head and master laws, and they had no legal right to any of their husbands' earnings or property, aside from a limited right to proper support; husbands, however, would control their wives' property and earnings.

**Figure 4.2** Ketchup advertisement and its context.

at that time.

The research challenge addressed in this chapter is how such context can be computed for helping in the interpretation of past or forgotten stories, e.g., from a news archive. We call this process *time-aware re-contextualization* [CTKN14, TCKN15a] or contextualization, for short. The process automatically provides complementing information to a textual document, which reflects required but not expressed context for fully understanding it. Although contextualization might also be necessary due to differences in cultural background or domain expertise, we focus on supporting time-aware interpretation, where a large time-gap between creation and reading time has to be bridged.

The need for dealing with content from the past is not restricted to expert users, such as, journalists, historians or researchers. Due to the growing age of the Web, general Web users are increasingly confronted with the content of different age assuming knowledge of the context at the respective time for its interpretation.

Just adding information, which is related to the entities and concepts mentioned in the text, as it is done in Wikification approaches, for example, [MC07, MW08b] or for a domain specific case [HdRS+11], is not sufficient for many reasons. First, we require a kind of a virtual time-travel, in which - by the information about the past - we are mentally transported into the time of content creation, in our example the US of the 50s. Second, the context information should be digestible in a short time with minimal disruption from the main reading. Therefore, we aim for a contextualization unit granularity, which is considerably smaller than a full Wikipedia page. Finally, contextualization has to coherently consider the specific aspects about entities, concepts or terms, which are relevant in the text under consideration.

Therefore, time-aware re-contextualization, which aims to associate an information item $d$ (such as, a paragraph in a text) with time-aware, concise and coherent context information $c$ for easing its understanding, is a challenging task. Several subgoals of the information search process have to be combined with each other [CTKN14, TCKN15a]: (1) $c$ has to be relevant for $d$, (2) $c$ has to complement the information already available in

$d$ and the surrounding document, (3) $c$ has to consider the time of creation (or reference) of $d$, and (4) the set of collected context information should be concise to avoid overloading the user.

In this chapter, we first define the problem and present the process of time-aware re-contextualization and provide advanced approaches for retrieval of contextualization candidates and ranking them by taking into consideration complementarity. In more detail, we follow a two-step process. In the first step, we identify contextualization candidates based on contextualization hooks, i.e., the parts of document that require contextualization.[2] For this purpose, we explore and analyze different methods for formulating (generating) queries, which are used for retrieving adequate contextualization candidates from an underlying knowledge source. In the second step, we rank the candidates. Similarly to diversification approaches, (e.g., [ZCM02]), this requires balancing two goals: high content-based and temporal relevance for the text to be contextualized, on one hand, and complementarity for providing information that cannot already be found in the text, on the other hand. In this work, we use Wikipedia as the knowledge source (because of its world-wide topical and temporal coverage) for contextualizing old stories from news articles.

Our main contributions in this chapter can be summarized as follows:

1. We are the first to frame the problem of time-aware re-contextualization for supporting the interpretation of documents.

2. We propose effective query formulation methods that take into account the contextualization hooks as well as recall-oriented query performance prediction using a set of novel features for adaptivity to the difficulty of the documents to be contextualized.

3. We present a time-aware ranking method based on learning-to-rank techniques using a novel feature set, and we propose a complementarity computation, which exploits ideas from search result diversification in ranking.

4. Using real-world datasets, we conduct extensive experiments to evaluate our time-aware re-contextualization approach, which achieves high precision and gains considerable improvement over the baselines. For fostering further research on this challenging task, a manually annotated ground-truth is made available.

## 4.2 Related Literature

Basic forms of contextualization have already been suggested in early works (such as [MC07, MW08b]). The Wikify! system [MC07], for example, enables an automated linkage of entity and concept mentions with Wikipedia pages. Meanwhile, a lot of progress has been made in further developing the entity disambiguation step (see e.g. [HYB+11]), which is

[2] Possible contextualization hooks are, for example, entity or concept mentions, and other phrases.

crucial for robust linking of entity mentions to Wikipedia entity pages or entity representations in other knowledge bases, such as, Yago, DBPedia or Freebase. Entity linking, or Entity Disambiguation, detects entity name mentions within text and links them to the corresponding entities in a knowledge base. In contrast to our approach, both Wikification and entity linkage approaches lack two ingredients of time-aware contextualization, (a) they do not take into account the temporal aspect of the text to be enriched and (b) the additional information provided is rather general (e.g., Wikipedia articles about an entity) and not focused to the topical information need resulting from the text under consideration.

In the area of time-aware information retrieval (IR), it has been shown that explicitly modeling the time dimension in ranking can improve the retrieval effectiveness for time-sensitive queries. There are two types of temporal information particularly useful for time-aware information retrieval: (1) the publication or creation time of a document [JD07, KN10], and (2) temporal expressions mentioned in a document or a query [BBAW10]. Aforementioned works address one of two main aspects for temporal relevance, i.e., recency ranking [DSD11, DCZ+10] or time-dependent ranking [BBAW10, KN10]. The first aspect takes into account the freshness of web documents, whereas the second aspect considers temporal information needs and the temporal profiles of documents.

Retrieving and processing external information to be added to documents gain increasing interest in the recent years. In [KBM11], for example, news articles are enriched with related predictions – sentences containing temporal references to the future – retrieved from other documents in the same collection. Other works [GLD12, TdRW11, vTP+13] exploit social media (e.g., Twitter) as external sources when processing news articles. In [vTP+13], the most *interesting* tweets regarding a given news are selected by formulating the tweet selection as an optimization problem. The objective function, representing how much a tweet set is interesting with respect to a news, takes into account diversity, popularity, authority, and opinions of tweets within the set.

The work in [TdRW11] discovers social media utterances that discuss a given news article. Multiple query models are generated from a news article by considering its internal structure, term selection strategies, as well as utterances which explicitly contain links to the article. The different resulting ranked list are then merged through data–fusion techniques. In [GLD12], the authors present a topic modeling approach which jointly exploits news articles and Twitter for event summarization. In order to generate a representative but not redundant summary of an event, complementarity between tweets and news article sentences is assessed by considering both their similarity and their difference. In contrast to those approaches, our work adds the time dimension to the contextualization task. Moreover, we are not looking for more information on the current context, but we try to re-construct the original context of a document.

The contextualization task is also related to the diversification problem in IR [CKC+08, ZCM02, ZLG+14]. In [ZCM02], different metrics are proposed to measure redundancy in order to investigate the novelty and redundancy of relevant documents in filtering systems. In [CKC+08], Clarke et. al. presented a framework for evaluation that systematically rewards novelty and diversity, whereas Zhu et. al. [ZLG+14] addressed diversification as

a learning problem and proposed a novel relational learning-to-rank approach to formulate the task.

In contrast to these studies on analyzing the relation between results to select diverse outcomes for a given query, we mainly focus on the relation between queries (documents in our case) and results (contexts) for finding the ones that are not only topical and temporal relevant, but also complement information already available in the documents.

Automatically formulating queries from text [HCMB03] can be done by using tf–idf, mutual information, natural language processing, or machine learning [Tur00, WPF+99, YBD+09]. Assuming the presence of basic metadata and structure for documents, as in [TdRW11], some of methods in this work build queries by exploiting the title and lead paragraph of documents. Similarly to [GPS99], we also explore approaches that assume the availability of manual annotations as seeds for query formulation. The advantage of having such additional information is that the information needs of the users are made explicit, possibly driving to more effective queries. We formulate queries by combining annotations via Query Performance Prediction (QPP) [CTZC02], using both pre–retrieval [HHdJ08] and post–retrieval [CYT10] features. The formers are based only on the query and corpus-based statistics, while the latter also analyze the retrieved list of results. In line with the previous work on time–aware performance predictor [KN11], we investigate novel features for QPP that explicitly take the temporal dimension into account. Differently from the previously mentioned approaches, which focus on precision metrics, we consider the performances of queries in terms of recall, which have been recently remarked and considered in different information retrieval scenarios [CG14, LWRM14].

## 4.3 Problem Definition and Approach Outline

Given a document $d$ with creation date $t_d$ and a source of background information $C$ (or a *contextualization source*), we define *time-aware re-contextualization* as the process of reconstructing the relevant part of the original context of document $d$ at time $t_d$ by retrieving information from $C$ that helps in interpreting $d$.

In the general contextualization model underlying our approach, we distinguish the information items $d$ to be contextualized and the contextualization source, where the information for the contextualization comes from. Within $d$ a contextualization hook $h$ is an aspect or part of $d$ that requires further information for its time-aware interpretation. The contextualization source is organized into contextualization units $cu$. More specifically, we have pre-processed a Wikipedia dump as the contextualization source resulting in annotated and indexed Wikipedia paragraphs as contextualization units (see Figure 4.3). For information items $d$ to be contextualized, we use articles from the New York Times Archive[3] with manually annotated contextualization hooks, i.e., we assume that a reader has marked the places he/she finds difficult to understand.

[3] http://catalog.ldc.upenn.edu/LDC2008T19

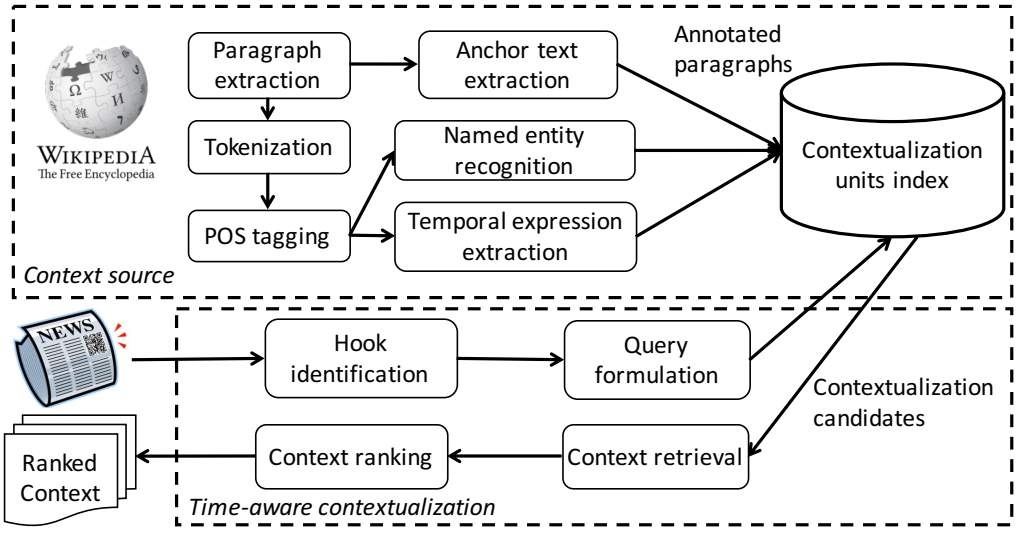



**Figure 4.3** Time-aware re-contextualization approach.

Starting from the contextualization hooks, the next process is to retrieve a ranked list of contextualization units from the context source. In time-aware re-contextualization, the time gap between the creation and reading time of $d$ imposes additional challenges. In our approach, the contextualization process consists of two main steps: (1) formulating queries that are able to retrieve contextualization units, which are good candidates for contextualization; (2) retrieving and ranking the candidates from the context source using the queries from step (1). For step (1), we explore document-based and hook-based query formulation methods and present a procedure that selects good queries based on recall-oriented query performance prediction. For step (2), we employ a retrieval method based on language modeling and re-rank the retrieved contextualization candidates based on a variety of features and a learning to rank approach for ensuring complementarity. The methods developed for steps (1) and (2) are described in more detail in the following sections.

## 4.4 Query Formulation

The goal of the query formulation phase consists of generating a set of queries $Q_d$ for a given document $d$ to retrieve contextualization candidates as input for re-ranking. We explore two families of query formulation methods, one using the document to be contextualized itself as a “generator" of queries (Section 4.4.1), and the other using contextualization hooks as generators (Sections 4.4.2 and 4.4.3). Since some of these methods can generate more than one query from an input document, we will discuss two procedures to merge the ranked result lists in Section 4.5.1.

### 4.4.1 Document-based Query Formulation

The first family of query formulation methods exploits the document content and structure. Similarly to [TdRW11], we use three methods to formulate queries from documents: *title*, *lead*, and *title+lead*. *Title* formulates a query consisting of the document title, which is indicative of the main topic of the article. *Lead* uses the lead paragraph of a document, representing a concise summary of the article and including its main actors. *Title+lead*, as a combination of the previous two methods, formulates a query consisting of both the title and the lead paragraph of the document.

Before being performed, all the queries are pre–processed by tokenization, stop-word removal, and stemming. We did not investigate further information extraction approaches for query formulation, since it has been already proven in [TdRW11] that the methods described above perform comparably or even better than more complex information extraction techniques, e.g., keyphrase extraction.

### 4.4.2 Basic Hook-based Query Formulation

As already introduced in Section 4.3, documents in our model are assumed to contain a set of hooks explicitly representing the information needs of the reader or, more precisely, what requires contextualization to be understood and interpreted. The analysis done in [CTKN14] showed that contextualization hooks are not only entity mentions, concept mentions, but also general terms and even short phrases.

We consider two basic hook–based query formulation methods: *all_hooks* and *each_hook*. *All_hooks* includes all the hooks for a document in a single query, representing a tailored perspective of the user's combined information needs for the document. *Each_hook* queries each hook separately, focusing on specific information about single actors, aspects, or sub–topics of the document. The queries generated by these methods are augmented with the title of the document, under the assumption that it is a good representative of the document's topic.

We also experimented with more advanced methods based on identifying hook relationships, for instance considering their co–occurrence in a document collection. However, since these approaches did not perform better than the *all_hooks* method described before, we will not discuss them further.

### 4.4.3 Learning to Select Hook-based Queries

Different methods based on ranking and selection of query terms from an initial query might be employed [BC08, LCKC09, MC13], considering the entire set of hooks for a document as the initial query. We explore an adaptive method which formulates queries based on the characteristics of the input document and hooks. Our approach consists of predicting the performances of candidate queries representing subsets of hooks for a given document, ranking them according to the predicted performance, and selecting the top-$m$

of them to be actually performed for the document. The value of $m$ is identified through experiments. In contrast to previous works in query performance prediction, the prediction model is trained on *recall* instead of *precision*. Furthermore, we define novel features for query performance prediction that explicitly take the temporal dimension into account. Finally, our method assesses performances of subsets of query terms (hooks) and can generate more than one query (subsets of hooks).

**Candidate Queries.** Given a document $d$ and the set of its hooks $H_d$, we compute its power set $\mathcal{P}(H_d)$ and create a candidate query for each set of hooks $p \in \mathcal{P}(H_d)$. Again, candidate queries are augmented with the title of the document.

The effort of the computation of features for each element in the power set is not critical in our scenario for two reasons. First, working with short text like news articles limits the number of hooks within the text. Second, the features employed to predict the query performances are either pre-retrieval measures, which can be computed off–line, or do not require heavy post-retrieval computation.

**Learning Features.** We measure the performances of each candidate query in terms of its recall because, as already explained, at retrieval phase we are interested in retrieving as many contextualization candidates as possible. In this work we predict query performances with a regression model learned via Support Vector Regression (SVR) [DBK+97]. In this model, each learning sample $s = \left(\boldsymbol{f}_q, r_q\right)$ consists in a feature vector $\boldsymbol{f}_q$ describing query $q$ (as well as the document it refers to) and its recall $r_q$, i.e., the label to be predicted. Note that different numbers of top–$l$ results can be used to compute the recall, i.e., the labels, and the choice is discussed in Section 4.7.

The feature set that we use to represent queries and the document they belong to are described in the rest of this section. It is composed of novel temporal features for query performance prediction, along with more standard features [CK12, HO04, MT05].

*Linguistic Features.* We compute a family of linguistic features [MT05] for a query by considering its text and the document it refers to. This results in a set of features both at query and document level: the length of the query, in words; the number of duplicate terms in the query; the number of entities (people, locations, organization, artifacts) in the query; the number of nouns in the query; the number of verbs in the query; the number of hooks in the query; the length of the document's title; the length of the document's lead paragraph; the number of entities in the document (title and lead paragraph); the number of nouns in the documents; the number of verbs in the document; the number of hooks for the document; the number of duplicates in the document.

*Document Frequency.* The Document Frequency of a hook $h$ represents the percentage of contextualization units in the corpus containing $h$ and it is computed as:

$$df(h) = \log \frac{N_h}{N} \tag{4.1}$$

where $N_h$ is the number of contextualization units in the corpus containing $h$ and $N$ is the size of the corpus. At document level, we compute the document frequency for every hook

of the document the query belongs to, i.e., $df(h)\ \forall h \in H_d$, and then we derive aggregate statistics like average, standard deviation, maximum value, minimum value. Similarly, at query level, we compute $df(h)$ for every hook in the query and we derive the same aggregate statistics as before. In the following, we will refer to average, standard deviation, maximum value, and minimum value simply as *aggregate statistics*.

*Temporal Document Frequency.* In order to restrict the popularity of a term to a particular time period $T = [t_0 - w; t_0 + w]$, we compute Equation 4.1 only for those contextualization units having at least one temporal reference contained in $T$. This can be done efficiently since contextualization units in our corpus have been annotated with the temporal references mentioned in them. The time period we are interested in is centered around the publication date of the document, i.e., $t_0 = p_d$, and the parameter $w$ determines the width of the interval. After experimenting different values of $w$, we set $w = 2$*years* for our study.

*Scope.* The scope of a query has been defined in [HO04] as the percentage of documents (contextualization units in our case) in the corpus that contain at least one query term. Besides the scope of the query itself, we also compute the scope of the document title and the scope of the document hooks $H_d$ when queried together.

*Temporal Scope.* We define the temporal scope of a query as the percentage of contextualization units in the corpus that contain at least one query term and at least one temporal expression within a given time period. The time period that we consider is the same as the one considered for the computation of temporal document frequency, i.e., a period centered around the publication date of the document and with a temporal window equal to $w$. Again, we experimented different values of $w$ and we set $w = 2$*years*.

*Relevance.* For a given query $q$, we retrieve the top-$k$ contextualization units and we compute aggregated statistics of their relevance scores given by the underlying retrieval model. The value of $k$ has been empirically set to 100 after experimenting different candidate values. We also computed relevance features at document level, using both document's title and document's hooks as queries.

*Temporal Similarity.* For a given query $q$ generated from a document $d$ and every retrieved contextualization unit $c$ in its top-$k$ result set (again, $k = 100$), we compute the temporal similarity between $q$ and $c$ and we derive aggregated statistics over the elements in the result set. Temporal similarity between time points $t_1$ and $t_2$ is computed through the time-decay function [KN10]:

$$TSU(t_1, t_2) = \alpha^{\lambda \frac{|t_1 - t_2|}{\mu}} \tag{4.2}$$

where $\alpha$ and $\lambda$ are constants, $0 < \alpha < 1$ and $\lambda > 0$, and $\mu$ is a unit of time distance. The temporal similarity between a query $q$ and a result $c$ is computed as $\max_{t \in T_c}\{TSU(t, p_d)\}$, where $T_c$ is the set of temporal references mentioned in $c$ and $p_d$ is the publication date of the document where $q$ refers to. This can be done efficiently since temporal references mentioned in contextualization units have been extracted and stored at indexing time.

We also computed temporal similarity features at document level, using both docu-

ment's title and document's hooks as queries. The computation of the features is the same as the one described above.

## 4.5 Context Ranking

In this section, we describe the methods used to address the second part of the contextualization process outlined in Section 4.3: retrieving and re-ranking context. For the retrieval step, given the queries generated from different query formulation methods described in previous section, we use a retrieval model based on language modeling to create a ranked list of contextualization candidates. Later, learning to select relevant context items is applied to this ranked list.

### 4.5.1 Retrieval Model

For the retrieval step, we use query-likelihood language modeling [PC98] to determine the similarity of a query with the context. In particular, given a query $q$ generated by using one of the methods described in Section 4.4 for the document $d$, we compute the likelihood of generating the query $q$ from a language model estimated from a context $c$ with the assumption that query terms are independent.

$$P(c|q) \propto P(c) \prod_{w \in q} P(w|c)^{n(w,q)} \tag{4.3}$$

where $w$ is a query term in $q$, $n(w, q)$ is the term frequency of $w$ in $q$, and $P(w|c)$ is the probability of $w$ estimated using Dirichlet smoothing:

$$P(w|c) = \frac{n(w, c) + \mu P(w)}{\mu + \sum_{w'} n(w', c)} \tag{4.4}$$

where $\mu$ is the smoothing parameter, $P(w)$ is the probability of each term $w$ in the collection.

To combine the rankings produced by each query of a document, we exploited two combining methods namely round-robin, which chooses one result from each ranked list, skipping any result if it has occurred before, and CombSUM, which sums up a result's scores from all ranked lists where it was retrieved. In the experiment, we observed that round-robin method achieves better performance than CombSUM especially in terms of recall, which also reported in [TdRW11]. Therefore, we decided to use round-robin method for combining different ranked lists.

### 4.5.2 Learning to Rank Context

Once we have obtained a ranked list of contextualization candidates for each document, we turn to context selection (re-ranking) where we need to decide which of the context items

are most viable. Our ranking algorithm needs to balance two goals, i.e., high topical and temporal relevance for the document, as well as complementarity for providing additional information. In this work, we use supervised machine learning, that takes as input a set of labeled examples (context to document mappings) and various complementarity features of these examples similar to diversity features [ZCM02].

*Topic Diversity.* This class of features is aimed to compare the dissimilarity between document $d$ and context $c$ on a higher level by representing them using topics. We use latent Dirichlet allocation (LDA) [BNJ03] to model a set of implicit topics distribution of the document and context. We define this feature as follows.

$$R_1(c,d) = \sqrt{\sum_{k=1}^{m}(p(z_k|d) - p(z_k|c))^2}$$

where $m$ is the number of topics and $z_k$ is the topic index.

*Text Difference.* In this case, we represent the document and context as a set of words. The novelty of context $c$ is measured by the number of new words in the smoothed set representation of $c$. If a word $w$ occurred frequently in context $c$ but less frequently in document $d$, it is likely that new information not covered by $d$ is covered by $c$. For computation, a document and its context are represented by a set of informative words (removing stop words and stemming) denoted by $Set(d)$ and $Set(c)$, respectively. We compute the text difference feature as follows.

$$R_2(c,d) = \|Set(c) \cap \overline{Set(d)}\|$$

*Entity Difference.* The way of computing entity difference is similar to the one for text difference, with the difference that a document and its context are represented by a set of entities. The feature is denoted as $R_3(c,d)$.

*Anchor Text Difference.* Anchor texts can be regarded as a short summary (i.e., a few words) of the target document and captures what the document is about. This feature can be computed similarly as text and entity features, which is denoted as $R_4(c,d)$. We extract anchor texts using WikiMiner [MW08b] with a confidence threshold $\gamma$.

*Distributional Similarity.* The next feature we use is distribution similarity, which is denoted as $R_5(c,d)$.

$$R_5(c,d) = -KL(\theta_c, \theta_d) = -\sum_{w_i} P(w_i|\theta_c) \log \frac{P(w_i|\theta_d))}{P(w_i|\theta_c)}$$

where $\theta_d$ and $\theta_c$ are the language models for a document $d$ and its context $c$, respectively, and are multinomial distributions. We compute $\theta_d$ (and similarly for $\theta_c$) using maximum likelihood estimation (MLE) given as:

$$P(w_i|d) = \frac{tf(w_i,d)}{\sum_{w_j} tf(w_j,d)}$$

The problem with using MLE is that if a word never occurs in the document $d$, the probability $P(w_i|d)$ will be zero; $P(w_i|d) = 0$. Thus, a word in the context $c$ but not in the document $d$ will make $KL(\theta_c|\theta_d) = \infty$. In order to solve this problem, we make use of the Dirichlet smoothing method.

$$P_\lambda(w_i|d) = \frac{tf(w_i, d) + \lambda p(w_i)}{\sum_{w_j}(tf(w_j, d) + \lambda p(w_j))}$$

*Geometric Distance.* There are several ways to compute geometric distance measure, such as, Manhattan distance and Cosine distance. We leverage Cosine distance because of its robustness to document length.

$$R_6(c, d) = \cos(c, d) = \frac{\sum_{k=1}^{n} w_k(c) w_k(d)}{\|d\| \, \|c\|}$$

In our experiment, we used each unique word as one dimension and the *tf.idf* score as the weight of each dimension.

*Relevance and Temporal Features.* In order to retrieve high topical and temporal relevant contextualization candidates for the document, we consider also relevance and temporal features. For the former one, we exploit the retrieval scores of context returned by our retrieval model. For the later one, we apply temporal similarity measurement, i.e., TSU which is described in the previous section.

## 4.6 Experimental Setup

### 4.6.1 Document Collections

In our experiments, we used the New York Times Annotated Corpus, which contains 1.8 million documents from January 1987 to June 2007, as the document collection to be contextualized. For context source, we employed Wikipedia because it is considered the largest and most up-to-date online encyclopedia covering a wide temporal range of general and specific knowledge. We obtained the Wikipedia dump of February 4, 2013 and considered *paragraphs* as contextualization units. In this particular snapshot, we obtain 4,414,920 Wikipedia articles that contain 25,708,539 paragraphs. For each paragraph, we used Stanford CoreNLP [MSB+14] for tokenization, entity annotation and temporal expression extraction. In addition, *anchor* texts found in the paragraph hyperlinks are also extracted. We used Apache Solr[4] to index the annotated paragraphs.

### 4.6.2 Ground-Truth Dataset

In order to obtain ground-truth dataset (both for training and evaluation), we manually selected a set of 51 articles that spanned a wide range of topics (business, technology,

[4] https://lucene.apache.org/solr/

education, science, politics, and sports) focusing on the older ones (29 articles published in 1987, 2 articles in 1988, 6 articles in 1990, 7 articles in 1991, and 7 articles in 1992) and recruited six human annotators to manually annotate those articles. The annotators were presented with an annotation interface with which they can evaluate article/context pairs (relevant or non-relevant). The annotation guidelines specified that the annotators should assign relevance to the context that contains *additional information* which complements the information in the article and does provide a good answer to (at least) one of the questions they think up when reading the article. For each article, we retrieved up to 20 contextualization candidates with each query formulation method and removed duplicates afterwards. In total, our annotation dataset consists of 9,464 article/context pairs, where the annotators evaluated 26.9 relevant context per article on average. To foster further research on this challenging task, our ground-truth dataset is publicly available.[5] We measured the inter-annotator agreement using Cohen's kappa statistic. We averaged the pairwise kappa values of all possible combinations of annotators that had overlapping candidates they had annotated and we obtained a fair agreement of $\kappa_c$ = 0.37 given the high complexity of this contextualization task, which includes objectivity and subjectivity.

**Parameter Settings.** For query performance prediction, the regression model described in Section 4.4.3 was built by using the Support Vector Regression implementation of LibSVM[6]. In particular, we trained a n–SVR model with Gaussian Kernel through 10–fold cross validation. The open parameters were tuned via grid search to $C = 3$, $\gamma = 0.5$ and $\nu = 0.75$. Linguistic features were extracted using Stanford CoreNLP [MSB+14].

For re-ranking context, we performed 5-fold cross validation at document level. We reported scores averaged over all testing folds. We conducted experiments using several machine learning algorithms to confirm the robustness of our approach, i.e., it does not depend on any specific algorithm. In this work, we employed Random Forests (RF), RankBoost (RB) and AdaRank that are implemented in RankLib.[7] In order to compute topic-based feature, we employed the topic modeling tool Mallet[8] by specifying the number of topics to 100, for this task. In addition, we set the confidence threshold to $\gamma = 0.3$ for extracting anchor texts using WikiMiner. For smoothing, we set $\mu$ = 2000 and $\lambda p(w_i)$ = 0.5.

For computing temporal similarity feature, we set $\lambda = 0.25$, $\alpha = 0.5$, and $\mu = 2$*years* in our experiments. We also observed that changing those parameters did not affect the correlation capabilities of the feature.

### 4.6.3 Evaluation Metrics

The evaluation metrics, we considered precision at rank 1, 3, 10 (P@1, P@3, P@10 respectively), recall, and mean average precision (MAP). These measures provide a short summary of quality of the retrieved context. In our experiment, a context is considered rel-

[5] http://www.l3s.de/~ntran/contextualization/
[6] http://www.csie.ntu.edu.tw/~cjlin/libsvm/
[7] http://sourceforge.net/p/lemur/wiki/RankLib/
[8] http://mallet.cs.umass.edu/topics.php

evant if it is marked as relevance by an annotator, otherwise we consider it as non-relevance. We used the top-20 returned context for evaluation because it is not expected that readers consider more than 20 contextualization units. Statistical significance was performed using a two-tailed paired t-test and is marked as $^{\blacktriangle}$ and $^{\triangle}$ for a significant improvement (with $p <$ 0.01 and $p <$ 0.05, respectively), and significant decrease with $^{\blacktriangledown}$ and $^{\triangledown}$ (for $p <$ 0.01 and $p <$ 0.05, respectively).

### 4.6.4 Baselines

For comparing to our approach, we considered three following competitive baselines.

*Milne and Witten (M&W).* The method proposed by Milne and Witten [MW08b] which represents the state-of-the-art in automatic linking approaches. We use the algorithm and best-performing settings as described in [MW08b]. In order to apply this method for our task, we consider all paragraphs of all linked pages as a candidate set.

*Language Model (LM).* The standard query-likelihood language model is used for the initial retrieval as described in Section 4.5 which provides the top retrieved documents as a candidate set for the contextualization task.

*Time-aware Language Model (LM-T).* Since we aimed at adding context to past stories, the temporal dimension is important. We selected a state-of-the-art time-aware ranking method, which has been shown very effective for answering temporal queries, as our third baseline. It assumes the textual and temporal part of the document $d$ are generated independently from the corresponding parts of the context $c$, yielding

$$P(d|c) = P(d_{text}|c_{text}) \times P(d_{time}|c_{time}) \tag{4.5}$$

where $d_{time}$ is the document's publication date, $c_{time}$ is the set of temporal expressions in the context $c$.

The first factor $P(d_{text}|c_{text})$ can be computed by Equation 4.3 and Equation 4.4. The second factor in (4.5) is estimated, based on a simplified variant of [BBAW10], as

$$P(d_{time}|c_{time}) = \frac{1}{\mid c_{time} \mid} \sum_{t \in c_{time}} P(d_{time}|t) \tag{4.6}$$

If the document has zero probability of being generated from the context, Jelinek-Mercer smoothing is employed, and we estimate probability of generating the document's publication date from context $c$ as

$$P(d_{time}|c_{time}) = (1-\lambda)\frac{1}{\mid C_{time} \mid} \sum_{t \in C_{time}} P(d_{time}|t) + \lambda \frac{1}{\mid c_{time} \mid} \sum_{t \in c_{time}} P(d_{time}|t) \tag{4.7}$$

where $\lambda \in [0, 1]$ is a tunable mixing parameter which is set to $\lambda = 0.5$ in our experiment (changing this parameter does not affect our results), and $C_{time}$ refers the temporal part of the context collection treated as a single context and $P(d_{time}|t)$ is estimated by using time-decay function, i.e., TSU computed as in Equation 4.2.

## 4.7 Results and Discussion

### 4.7.1 Query Formulation

We evaluate and compare the performances of the different query formulation methods described in Section 4.4, focusing on recall metric. The results reported in the rest of this section are averaged over the 51 documents in our dataset.

In order to fairly evaluate and compare the recall capabilities of the different methods, which can generate different numbers of queries, we allow each method to retrieve the same number of results $k$. The choice of the method that we used to create a single result set of $k$ elements from different ranked lists have been discussed in Section 4.5.1.

**Prediction Performances.** The query formulation method described in Section 4.4.3 is based on predicting the performances (recall in our case) of candidate queries, ranking them according to the prediction, and then using the top-$m$ queries to retrieve results. Thus, the quality of the query performance prediction itself has to be evaluated before assessing and comparing the performances of the whole query formulation method.

The regression model has been trained via 10–fold cross validation, and the results reported hereafter have been averaged over the 10 folds. The Correlation Coefficient is equal to 0.973, the Root Mean Squared Error equals to 0.056, and the Mean Absolute Error equals to 0.037. The low error values and high correlation value, if compared with the performances in predicting query precision reported in previous works (e.g. [CYT10, RK14]), show that the recall of queries in our task can be predicted quite accurately by using the features described in Section 4.4.3.

**Feature Analysis.** In order to analyze which are the most important features in our model, we identified the top–10 features according to their absolute correlation coefficient. Referring to Section 4.4.3, these are: *max query relevance*, *number of hooks in document*, *min document's hooks df*, *max document's hooks temporal df*, *document's hooks scope*, *avg query temporal similarity*, *document's title temporal scope*, *std query relevance*, *avg document's title temporal similarity*, and *std query temporal similarity*. The presence of temporal document frequency, temporal similarity, and temporal scope shows that the temporal features that we defined play an important role in the model. We can also note that both query–level and document–level features are important, since the set is made of 4 features from the former and 6 features from the latter class. Finally, there is only one linguistic feature in the set, namely the number of hooks in the document, confirming that

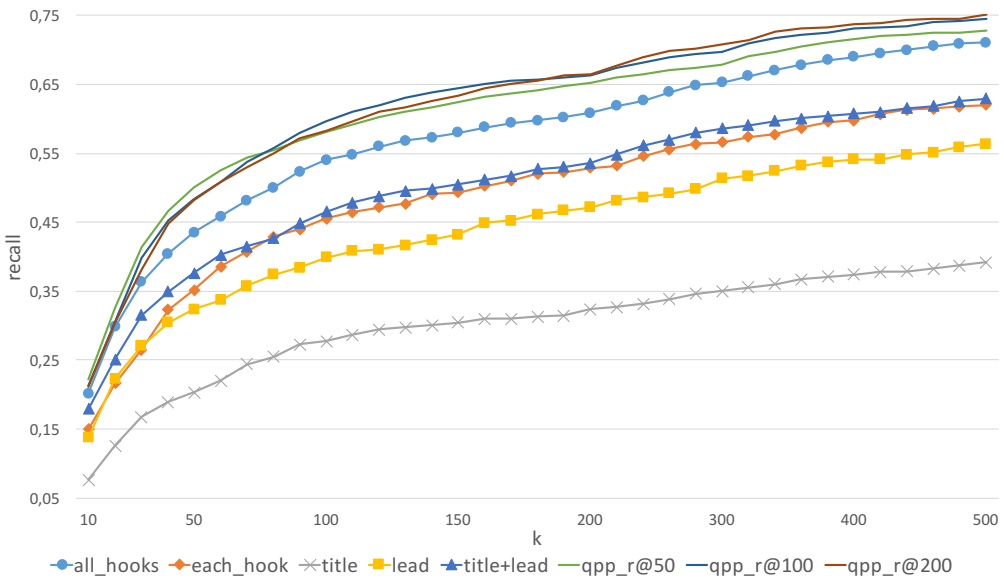



**Figure 4.4** Recall curves of document-based and hook-based methods.

this class of features alone does not correlate well with query performances [CK12].

**Comparison of Query Formulation Methods.** In the following, we compare recall values for the document–based methods (*title*, *lead*, *title+lead*), the basic hook–based methods (*each_hook*, *all_hooks*), as well as the method based on query performance prediction, hereafter called $qpp$. For the latter method, we report the performances achieved when using prediction models trained with different labels: we experimented with different $l$ values, namely $l = 50, 100$ and 200, for the computation of the recall at $l$ to be used as label. These three methods will be called *qpp_r@50*, *qpp_r@100* and *qpp_r@200*, respectively, in the rest of the experiments. Note that each $qpp$ method considered here uses the top-2 queries, according to their predicted performances, to retrieve the results. The choice of selecting $m = 2$ queries will be explained in more detail in the following paragraph.

The recall curves of the different methods, for different values of top–$k$ results, are shown in Figure 4.4. The curves of $title$ and $lead$ are the lowest ones, while their combination ($title + lead$) becomes comparable with $each\_hook$. Querying using all the hooks of a document together, i.e., $all\_hooks$, exposes higher recall values than all the aforementioned methods, showing that performing hook–based queries does lead to better performances in terms of recall with respect to document–based methods. The difference in performances between $each\_hook$ and $all\_hooks$ is due to the fact that querying all the hooks together prefers contextualization candidates that contain many hooks. These are potentially more relevant, as they refer to different aspects (hooks) of the same document. Regarding the $qpp$ methods, for $k > 20 - 30$, the recall values achieved are between 3% and 7% higher than the ones obtained by $all\_hooks$. For larger values of $k$, e.g. $k > 400$, the difference between the $qpp$ methods and $all\_hooks$ reduces because the prediction models used by the $qpp$ methods have been optimized for lower values of $k$ (recall that $l = 50, 100, 200$).

| | R@50 | | R@100 | | R@200 | |
|---|---|---|---|---|---|---|
| | *qpp* | *all_hooks* | *qpp* | *all_hooks* | *qpp* | *all_hooks* |
| *easy* | 0.6208 | 0.5666 | 0.7361 | 0.6969 | 0.7951 | 0.7686 |
| *hard* | 0.3837 | 0.3094 | 0.4606 | 0.3892 | 0.5391 | 0.4550 |

**Table 4.1** Recall of *all_hooks* and *qpp* methods over different classes of documents grouped by their retrieval difficulty.

This means that, if the number of $k$ results to be retrieved for the re-ranking phase is known and fixed in advance, this information can be exploited early in the training of the query performance prediction model by setting $l = k$, leading to higher recall values for that particular $k$.

Another comparative analysis between $qpp$ methods and $all\_hooks$ can be done by categorizing the documents according to their *difficulty*, which we define in terms of the amount of relevant context that can be retrieved for a given document. This means that difficult documents are those for which few relevant context can be retrieved, before the re-ranking phase. We categorize documents in $easy$ and $hard$ with respect to the $all\_hooks$ method, since it represents a baseline in this comparative analysis with $qpp$ methods.

The splitting of the documents in easy and hard was performed by considering the recall at $k = 200$ achieved by $all\_hooks$ for the different documents. Since the recall values associated to the different documents exhibited a uniform distribution, we split the document set in two equal parts, one representing easy documents and the other representing hard documents.

Table 4.1 shows the performances of $qpp\_r@50$, $qpp\_r@100$, and $qpp\_r@200$ compared to the ones of $all\_hooks$ for the different categories of difficulty. The comparison between each $qpp$ method and $all\_hooks$ is done considering the recall at those $k$ values used to train the prediction model (i.e. $k = l$, $l = 50, 100, 200$). Besides $qpp\_r@50$, $qpp\_r@100$, and $qpp\_r@200$ are on average better than $all\_hooks$ both for easy and hard documents, their improvements are greater for hard documents. In case of $qpp\_r@100$, for instance, the relative improvement with respect to the recall value achieved by $all\_hooks$ is 5.6% for easy documents and 18.3% for hard documents. We believe that the capability of getting higher recall improvements for documents whose relevant context units are difficult to retrieve is a considerable characteristic for the $qpp$ methods.

As a conclusion, in this section we proved that exploiting hooks in query formulation is more effective, in terms of recall, than document–based query formulation methods. Moreover, we showed that learning to select candidate hook-based queries can be better, again in terms of recall, than the basic hook–based query formulation methods.

**Number of Queries.** The number of top ranked queries that $qpp$ methods perform is an open parameter, which we tuned via an empirical analysis observing the recall performances when selecting different numbers of top–$m$ ranked queries. Recall that, for sake

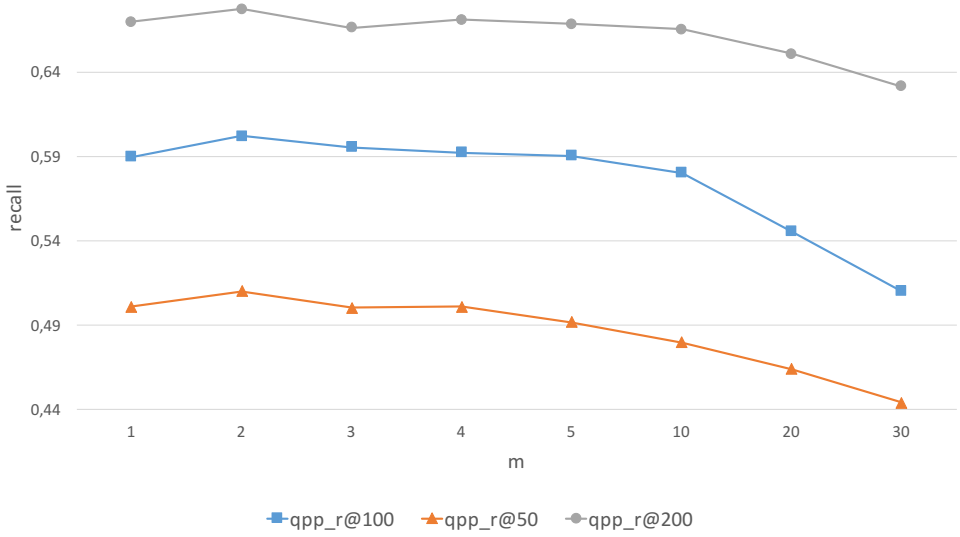



**Figure 4.5** Recall values of $qpp\_r@50$, $qpp\_r@100$, and $qpp\_r@200$ by varying the number of top–$m$ queries.

of fair comparison, we allow each method to pick the same number of results $k$ from the result lists retrieved by the queries that it generated for a given document. This means that increasing the number of queries to be selected and performed does not necessarily lead to higher recall.

Figure 4.5 shows the recall values achieved by $qpp\_r@50$, $qpp\_r@100$ and $qpp\_r@200$ (computed at top–50, top–100 and top–200 results, respectively) for different numbers of top–$m$ selected queries. A common trend over the different curves can be observed that they stay quite stable for small values of $m$, exhibiting a little peak for $m = 2$, and then they decrease for increasing values of $m$. After observing this behavior, we decided to fix the number of performed queries to $m = 2$.

### 4.7.2 Context Ranking

In this section, we report the retrieval performances of different query formulation methods and analyze the effectiveness of our context ranking methods trained by using different machine learning algorithms. Firstly, we investigate the performance of the standard, well-known Wikification technique, i.e., the M&W method, in retrieving contextualization candidates. Our experiment considers all paragraphs of all linked pages as candidates. This method achieves the low recall value of 0.229, which indicates that current semantic linking approaches are not appropriate for the contextualization task.

Table 4.2 shows the results of different query formulation methods. The first group (top) reports results for candidate retrieval based on document-based query models in which the best performing model is *title+lead* that uses content from the article's title and lead

| | P@1 | P@3 | P@10 | MAP | Recall |
|---|---|---|---|---|---|
| *Document-based query models* | | | | | |
| title | 0.2156 | 0.1895 | 0.1745 | 0.2446 | 0.1211 |
| lead | 0.4902▲ | 0.4641▲ | 0.3333▲ | 0.4908▲ | 0.2603▲ |
| title + lead | 0.5294▲ | 0.4705▲ | 0.3901▲ | 0.5161▲ | 0.2723▲ |
| *Basic hook-based query models* | | | | | |
| each_hook | 0.3333 | 0.3464 | 0.2745 | 0.4003 | 0.1969 |
| all_hooks | 0.5490 | 0.5098 | 0.4137 | 0.5640 | 0.2979 |
| *Query performance prediction model* | | | | | |
| qpp_r@100 | **0.5882** | **0.5490**▲ | **0.4529**▲ | **0.5802**▲ | **0.3097**▲ |

**Table 4.2** Retrieval performance of document-based and hook-based query models. The significance test is compared with Row 1 (within the first group) and Row 3 (for the second and third groups).

| | P@1 | P@3 | P@10 | MAP | Recall |
|---|---|---|---|---|---|
| all_hooks | 0.5000 | 0.3462 | 0.2885 | 0.4487 | 0.2217 |
| qpp_r@100 | 0.5000 | **0.4743**△ | **0.3730**△ | **0.5048**△ | **0.2357** |

**Table 4.3** Retrieval performance of *all_hooks* and *qpp_@100* on a set of difficult documents.

paragraph. Turning into models derived from contextualization hooks, Table 4.2 shows that the *qpp_r@100* model performs the best among all hook-based query models and significantly improves over *title+lead* on all metrics.

Similar to the previous experiment, Table 4.3 reports the results of *qpp_r@100* and *all_hooks* retrieval baselines on a subset of difficult documents (here recall is computed on top-20 candidates). On this subset, *qpp_r@100* also shows significant improvement over *all_hooks* in terms of precision. In short, the results on different query formulation methods indicate that using hook-based approaches outperforms the document-based approach that based on merely article internal structure. Using the query performance prediction method obtains the highest performance on all metrics, followed by *all_hooks*.

We now present the results of our re-ranking approach when using a set of innovative complementarity features to further improve performances of the context ranking step, especially in terms of precision. We select *title+lead* for the document-based approach and *all_hooks, qpp_r@100* for the hook-based approach.

The first (top) group in Table 4.4 shows the results when applying machine learning to *title + lead* retrieval baseline. All three algorithms are able to improve precision at rank k, MAP and Recall. Random forest (RF) and RankBoost (RB) obtain significant improvement where RF achieves the highest scores on most metrics, except precision at rank 3

| | P@1 | P@3 | P@10 | MAP | Recall |
|---|---|---|---|---|---|
| **title + lead** | | | | | |
| LM | 0.5294 | 0.4705 | 0.3901 | 0.5161 | 0.2723 |
| RandomForest | **0.7672**$^{\blacktriangle}$ | 0.5757$^{\triangle}$ | **0.4909**$^{\blacktriangle}$ | **0.6170**$^{\blacktriangle}$ | **0.3522**$^{\blacktriangle}$ |
| RankBoost | 0.6036 | **0.5945**$^{\triangle}$ | 0.4694$^{\blacktriangle}$ | 0.5945 | 0.3417$^{\blacktriangle}$ |
| AdaRank | 0.6254 | 0.5406 | 0.4143 | 0.5457 | 0.3249 |
| **all_hooks** | | | | | |
| LM | 0.5490 | 0.5098 | 0.4137 | 0.5640 | 0.2979 |
| RandomForest | **0.8272**$^{\blacktriangle}$ | **0.6630**$^{\blacktriangle}$ | **0.5014** $^{\blacktriangle}$ | **0.6427**$^{\triangle}$ | 0.3611$^{\blacktriangle}$ |
| RankBoost | 0.7855$^{\blacktriangle}$ | 0.6593$^{\blacktriangle}$ | 0.5009$^{\blacktriangle}$ | 0.6475$^{\triangle}$ | **0.3637**$^{\blacktriangle}$ |
| AdaRank | 0.6472 | 0.5836 | 0.4687 | 0.6034 | 0.3372$^{\triangle}$ |
| **qpp_r@100** | | | | | |
| LM | 0.5882 | 0.5490 | 0.4529 | 0.5802 | 0.3097 |
| RandomForest | **0.8054**$^{\blacktriangle}$ | **0.6993**$^{\blacktriangle}$ | 0.5140$^{\blacktriangle}$ | 0.6498$^{\blacktriangle}$ | **0.3951**$^{\blacktriangle}$ |
| RankBoost | 0.7218 | 0.6915$^{\blacktriangle}$ | **0.5300**$^{\blacktriangle}$ | **0.6632**$^{\blacktriangle}$ | 0.3792$^{\blacktriangle}$ |
| AdaRank | 0.6072 | 0.6139 | 0.4895 | 0.6109 | 0.3479$^{\blacktriangle}$ |

**Table 4.4** Retrieval performance of different machine-learned ranking methods compared to the best performing retrieval baselines.

where RB is the best. The second (middle) group reports the results of *all_hooks* retrieval baseline, augmented by the re-ranking step. In this case, RF and RB are again able to significantly improve over *all_hooks* on all metrics while AdaRank is also performing significantly better than *all_hooks* in terms of recall. Among three algorithms, RF achieves the highest results, except for recall. Similarly, all three machine learning algorithms perform significantly better than the *qpp_r@100* retrieval baseline. Again, in this case RF obtain the highest performances, closely followed by RB.

In order to compare our approach to time-aware language model which takes into account temporal information, we use the queries derived from query performance prediction method, i.e., *qpp_r@100* that obtain the highest results among our query formulation methods. Table 4.5 shows that using time-aware language models is not efficient in our case. This is possibly due to that lots of relevant context (paragraphs in our case) do not have any temporal information. Consequently, these candidates are ranked low (e.g., higher than 20) in the ranked list returned by LM-T. This result indicates that purely using the time dimension in context retrieval is not sufficient in the contextualization task. It also confirms the importance of complementarity that is used in our re-ranking step.

| **qpp_@100** | P@1 | P@3 | P@10 | MAP | Recall |
|---|---|---|---|---|---|
| LM-T | 0.5882 | 0.4967 | 0.4176 | 0.5446 | 0.2796 |
| LM | 0.5882 | 0.5490 | 0.4529 | 0.5802 | 0.3097$^{\triangle}$ |
| RandomForest | **0.8054**$^{\triangle}$ | **0.6993**$^{\blacktriangle}$ | **0.5140**$^{\blacktriangle}$ | **0.6498**$^{\blacktriangle}$ | **0.3951**$^{\blacktriangle}$ |

**Table 4.5** Retrieval performance of our proposed ranking method and the state-of-the-art time-aware language modeling approach. The significance test is compared against LM-T.

## 4.8 Chapter Summary

As we already discussed in Chapter 1, fully understanding documents requires context knowledge from the time of document creation and finding information about such context is a tedious and time-consuming task. To study this, we introduced the task of time-aware re-contextualization of context with a gap between creation and reading time. In particular, we aimed to answer the research question: *How to bridge temporal context gaps for supporting interpretations of documents by time-aware re-contextualization?* (**RQ2**).

For this purpose, we presented (1) different query formulation methods for retrieving contextualization candidates and (2) ranking methods taking into account topical and temporal relevance as well as complementarity with respect to the original text. Our results showed that our approach can compute relevant and complementing contextualization information with high precision. In addition, hook-based query formulation methods have outperformed document-based ones supporting the validity of our contextualization model, and the predominance of query formulation methods relying on several hooks shows the importance of comprehensive contextualization approaches that go beyond the consideration of individual hooks. Furthermore, our experiments have confirmed that complementarity, which is used in the re-ranking step, plays an important role in contextualization.

# 5 Dynamic Context-Aware Entity Recommendation

In the previous chapters, we described approaches to gain overall understanding of documents based on the document content and structure and additional context retrieved via re-contextualization. In this chapter, we turn our attention to a more fine-grained but important aspect of documents, i.e. **entities**. In particular, we present a novel approach which takes into account contextual information to suggest related entities to an entity of interest.

## 5.1 Introduction

Entities are characterized not only by their intrinsic properties, but also by the manifold relationships between them. Quantifying these entity relationships, which is the idea of entity relatedness [SP06, MW08a, HSN+12], is crucial in several tasks such as entity disambiguation [HYB+11, BOM15], contextualization of search results, and improved content analysis [TCKN15a].

Relationships between entities are not always static. While some relationships are robust and static, e.g. the relationship between a country and its cities, others change frequently, driven by dynamic contexts. In these contexts, time is just one dimension, and alone not sufficient to adequately structure the entity relationship texture. This is illustrated for the entity Brad Pitt in Figure 5.1. While time is sufficient to structure the realm of his private relationships, there are other groups of related entities with overlapping timelines, such as the persons he co-acted with in films, which relate to other contexts of his life. Such more fine granular, contextual understanding of the entity relationship texture can be used to refine methods such as entity disambiguation and entity recommendation.

In this work, we introduce the novel notion of *contextual entity relatedness*, with time and topic as two main ingredients, and show its usefulness in a new yet important problem: Context-aware entity recommendation. We propose to estimate the contextual relatedness using both entity graph extracted from knowledge sources such as Wikipedia, and also to exploit annotated text data using entity embedding methods. Furthermore, while existing

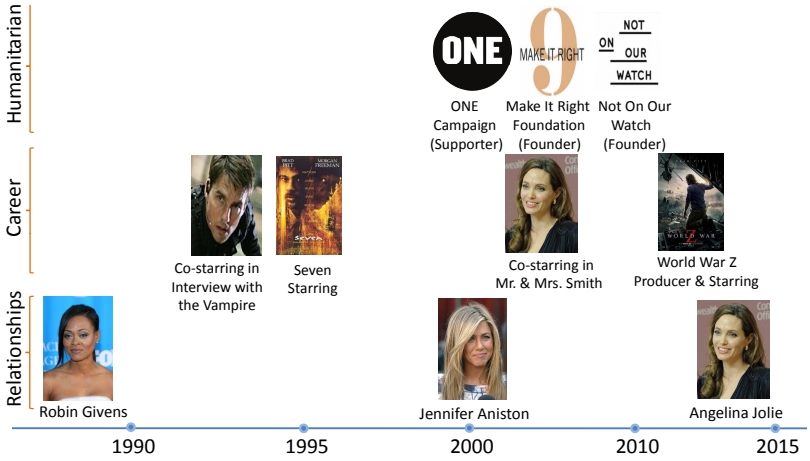



**Figure 5.1** Related entities with Brad Pitt in different topics and time periods

work adds temporal aspects into entity relationships [WZQ+10, ZRZ16], we go a step beyond by incorporating topic and proposing to enrich the relationships to form a novel *contextual entity graph*: Each entity relation is enriched with the time span and topics indicating *when* and *under which circumstances* it exists.

From the application perspective, entity recommendation is one of the directed applications of entity semantic relatedness. It assumes the input entities encode some user activities or information needs, and suggests a list of entities, normally ordered, that are most relevant. Blanco et al. [BCMT13] introduced Spark that links a user search query to an entity in a knowledge base and suggests a ranked list of related entities for further exploration. Similarly, Yu et al. [YMHH14] and Bi et al. [BMH+15] proposed personalized entity recommendation which uses several features extracted from user click logs. Our work is distinguished from these methods in that we take into account context as an additional information need, not just input entities.

In short, the main contributions of this chapter are summarized as follows:

- We introduce the idea of a contextual relatedness of entities and define the problem of context-aware entity recommendation for validating the usefulness of contextual relatedness.
- We propose a novel method for tackling the defined problem based on a statistically sound probabilistic model incorporating temporal and topical context via embedding methods.
- We evaluate the context-aware recommendation method with large-scale experiments on a real-world data set. The results of the evaluation show the usefulness of contextual entity relatedness as well as the effectiveness of our recommendation method compared to other approaches.

## 5.2 Related Literature

Estimation of entity semantic relatedness is an important task in various semantic and NLP applications, and has been extensively studied in literature [SP06, MW08a]. Strube and Ponzetto [SP06] proposed using Wikipedia link structures and the hierarchy of Wikipedia categories to provide a light-weight related estimation. Milne and Witten [MW08a] followed a similar approach, and carefully designed the relatedness measure based on Wikipedia incoming links, inspired by the Google distance metric.

These methods are close to our work in that we also combine various similarity measures, but do so in an advanced probabilistic model, taking into account context information. Hence, while the aforementioned works are static, our proposed measure is context-aware and dynamic to time.

One main issue with relatedness measures based on link structures is that they perform poorly for long-tail entities with little or no connections. Hoffart et al. [HSN+12] (KORE) addressed this issue by extracting key phrases from surrounding texts of entity mentions, and incorporate the overlaps of such key phrases between two entities. In our work, we also use the text surrounding of entity mentions. However, in contrast to KORE that uses these texts to enrich the entities, we use the texts to enrich the relations between entities, and in this regard, can contextualize the relatedness directly. In addition, KORE is still a static quantity, while our measure is fully dynamic to time and context.

Several approaches have been proposed to add temporal dimension to entity semantic relationships [WZQ+10, TEPW11]. Wang et al. [WZQ+10] extracted temporal information for entities with focus on infobox, categories and events. Tuan et al. [TEPW11] also extracted information from infobox and categories, but defined a comprehensive model comprising time, location and topic. However, these studies are limited to predefined types of relations, and cannot be easily extended to address the semantic relatedness. Recently, Zhang et al. [ZRZ16] incorporated various correlation metrics to complement the semantic relatedness, proposed a new metric that is sensitive to time. We extend this work, but incorporate time and topic in an consistent context model, and also introduce the entity embedding method.

From the application perspective, entity recommendation is one of the directed applications of entity semantic relatedness. It assumes the input entities encode some user activities or information needs, and suggests a list of entities, normally ordered, that are most relevant. Blanco et al. [BCMT13] introduced Spark that links a user search query to an entity in a knowledge base and suggests a ranked list of related entities for further exploration. Similarly, [YMHH14, BMH+15] proposed personalized entity recommendation which uses several features extracted from user click logs. Our work is distinguished from this work in that we take into account context as an additional information need, not just input entities.

## 5.3 Background and Problem Definition

### 5.3.1 Preliminaries

In this work, we use a very general notion of an entity as "a thing with distinct and independent existence" and assume that each entity has a canonical name and is equipped with a unique identifier. Typically, knowledge sources such as Wikipedia or Freebase are used as reference points for identification.

There are relations between entities. These are represented in different ways such as in the form of hyperlinks in Wikipedia or by a fact in an ontological knowledge base asserting a statement between two entities. Entities and their relationships can be captured in an **entity graph**, where the nodes are entities the edges represent relationships between entities. Such a graph can be heterogeneous in general, i.e. the edges between nodes can be of different types, corresponding to different connection types between entities.

An entity can be referred to in a text document (e.g. a news article) in the form of an *entity mention*. In our work, we assume that an **annotated corpus** is given, i.e., an annotated text dataset with well disambiguated entities.[1] Such an annotated corpus can be used to create and enrich the entity graph.

We are interested in the relatedness between entities, which is the association of one entity to another. Such a relatedness is often measured by a normalized score indicating the strength of the association. In our work, these scores depend upon the context and we speak of **contextual relatedness**. For ensuring a wide applicability, we use a simple yet flexible model of context, constituted by two dimensions: Time and Topic. We formalize this concept as follows.

**Context.** *A context $c$ is a tuple $(t, s)$, where $t$ is a time interval $[t_b, t_e]$ and $s$ is a topic describing the circumstance of the relationship.*

Our notion of time is a sequence of discrete time units in a specific granularity, e.g. a day. Time points or ranges of other granularities will be mapped to an interval of this granularity. For example, "2016" is converted to $[2016\text{-}01\text{-}01, 2016\text{-}12\text{-}31]$. For the topic $s$, we use a textual representation. It can be a single word such as "*movies*", "*wars*", or a phrase indicating an information interest such as "*scenes in the thriller movie SEVEN*".

It is important to note that our contextual relatedness is an asymmetric measure, i.e. given a context $c$, the relatedness of an entity $e_2$ to an entity $e_1$ is different from that of $e_1$ to $e_2$. For example, in the context $(2016$, "medals"), 2016_Summer_Olympics is likely to be the highest related entity for Eri_Tosaka, the Japanese female wrestler[2] who won her first Olympics gold medals in Rio. The reversed direction is not true, as there are many winners for the total $306$ sets of medals in the games.

[1] Such collections are increasingly available thanks to the advancement in information extraction research. One example is Freebase annotated KBA dataset: http://trec-kba.org/data/fakba1/

[2] https://en.wikipedia.org/wiki/Eri_Tosaka

### 5.3.2 Problem Definition

In this work, we aim to study the usefulness of context in entity relatedness. We do this by undertaking a specific recommendation task, namely *context-aware entity recommendation*. In this task, context reflects a user intent or preference in exploring an entity, and contextual relatedness can be used to guide the exploration. Accordingly, by validating the performance of the recommendation task, the effectiveness of contextual relatedness can be evaluated. More specifically, the input of the recommender system is an entity, which the user wants to explore (e.g., Brad_Pitt), and a context consisting of the aspect she is interested in (e.g., (1995-2015,"awards")); the goal is to find the most related entities given the entity and the context of interest. We give the formal definition as follows:

**Context-aware Recommendation:** *Given an entity $e_q$, a context of interest $c_q$, an entity graph $G$, and an annotated corpus $D$ containing annotated and disambiguated entity mentions, find the top-$k$ entities that have the highest relatedness to $e_q$ given the context $c_q$ (contextual relatedness).*

The query $(e_q, c_q)$ is called an entity-context query. The context-aware entity recommendation problem has some assumptions regarding the query setting. First, query entities can have free text representations, but a *text-to-entity* mapping to resolve the canonical entity name is employed. Such a mapping can be the result of using an entity linking system (e.g., [BOM15]). Second, there is also a map from the textual context representation to the time and topic component, for instance "Black Friday 2016 ads" to ([2016-11-25, 2016-11-25], "ads"). Third, in the absence of time or topic, they will be replaced by some default place holders. For time, we define two special values $b_t$ and $e_t$ to refer to the earliest and latest days represented in the corpus. For topic, we replace missing values by the token "$*$" to indicate an arbitrary topic.

## 5.4 Approach Overview

This section gives an overview of our method. In essence, we use a probabilistic model to tackle the recommendation task. To estimate the model, we incorporate different graph enrichment methods. These two components are described below.

### 5.4.1 Probabilistic Model

We formalize the context-aware entity recommendation task as estimating the probability $P(e|e_q, c_q)$ of each entity $e$ given a entity-context query $(e_q, c_q)$. The estimation score can be used to output the ranked list of entities. Based on Bayes' theorem, the probability can be rewritten as follows:

$$P(e|e_q, c_q) = \frac{P(e, e_q, c_q)}{P(e_q, c_q)} \propto P(e, e_q, c_q) \tag{5.1}$$

where the denominator $P(e_q, c_q)$ can be ignored as it does not change the ranking. The joint probability $P(e, e_q, c_q)$ can be rewritten as:

$$\begin{aligned} P(e, e_q, c_q) = P(e, e_q, t_q, s_q) = & \quad P(e_q)P(t_q|e_q)P(e|e_q, t_q)P(s_q|e, e_q, t_q) \\ \stackrel{\text{rank}}{=} & \quad P(e|e_q, t_q)P(s_q|e, e_q, t_q) \end{aligned} \tag{5.2}$$

In Equation 5.2, we drop $P(e_q)$ and $P(t_q|e_q)$ as they do not influence the ranking. The main problem is then to estimate the two components: $P(e|e_q, t_q)$ (*temporal relatedness* model), and $P(s_q|e, e_q, t_q)$ (the *topical relatedness* model).

### 5.4.2 Candidate Entity Identification

The entity graph can be very large, e.g. millions of entities and tens of millions of relationships, thus it is costly to estimate $P(e, e_q, c_q)$ for all entities in the graph. To improve the efficiency, we employ a candidate selection process to identify the promising candidates. Given the query ($e_q$, $c_q$), we extract all entities directly connected to $e_q$. Other methods can be used in this step; for example entities that co-occur with the target entity in an annotated corpus can be considered as candidate entities. However, in practice, we observe that this strategy covers sufficiently large amount of entities we need to consider.

### 5.4.3 Graph Enrichment

To facilitate the estimation methods for Equation 5.2 (see Section 5.5 for more details), we propose to enrich the entity graph, i.e. is to equip all entities as well as their relationships with rich information from the knowledge sources and the annotated corpus. This enrichment extends the entity graph into a **contextual entity graph**, where both nodes and edges are contextualized. We describe the enrichment methods below.

**Entity Relationship Enrichment.** First, we describe how we enrich the graph edges, i.e. the entity relationships. From the annotated corpus, we extract the set of bounded *text snippets* (e.g. a sentence or paragraph)[3], in which one or multiple entity mentions to the entities can be found. Then, for each edge $(e_i, e_j)$, we construct the set of all text snippets annotating both entities $e_i$ and $e_j$. For each text snippet, we employ a temporal pattern extraction method to extract the time values, and map them to day granularity, or put a placeholder if no values are found. For each successfully constructed time $t$, we create a context $c = (t, s)$, where $s$ refers to the textual representation of the snippet. As a result, for each edge $(e_i, e_j)$, we have a set of *relation contexts*, denoted by $C(e_i, e_j)$.

**Entity Embedding.** To enrich the graph node, i.e. the entity, we propose to learn a continuous vector representation of the entities in the entity graph using a neural network. Our method, *entity embedding*, maps entities to vectors of real numbers so that entities appearing in similar contexts are mapped to vectors close in cosine distance. The vectors can

[3]In our experiments, we limit to sentences level.

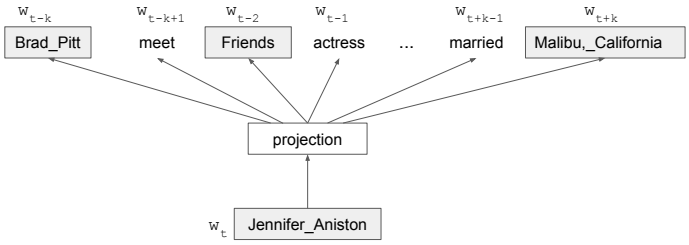



**Figure 5.2** The training example for the Jennifer Aniston entity

be estimated in a completely unsupervised way by exploiting the distributional semantics hypothesis. Here we extend the Skip-Gram model [MCCD13] as described in Chapter 2. In principle, the Skip-Gram aims to predict context words given a target word in a sliding window. In our case, we aim to predict context words given a target entity. We train the entities and the words *simultaneously* from the annotated text collection $D$, using text snippets as the window contexts. Specifically, given a context as a text sequence in which the target entity $e$ appears, i.e., $W = \{w_1, ..., w_M\}$ where $w_i$ might be either an entity or a word, the objective of the model is to maximize the average log probability

$$\mathcal{L}(W) = \frac{1}{M} \sum_{i=1}^{M} \log P(w_i|e) \tag{5.3}$$

in which the prediction probability is defined by using a softmax function

$$P(w_i|e) = \frac{\exp(\vec{w}_i \cdot \vec{e})}{\sum_{w \in W} \exp(\vec{w} \cdot \vec{e})} \tag{5.4}$$

where $\vec{w}$ and $\vec{e}$ denote the vector representation of $w$ and $e$ respectively. The training example is shown in Figure 5.2. The relatedness between two entities $e$ and $e_q$ is then defined as the cosine similarity between their vector representations. In the experiment, we show that the embedding method complements to standard relatedness metrics and help to improve the performance in estimating both models of the contextual relatedness (Equation 5.2).

## 5.5 Model Parameter Estimation

Our probabilistic model is parameterized by two relatedness models $P(s_q|e, e_q, t_q)$ and $P(e|e_q, t_q)$. In this section, we present in details the estimation of these models based on the contextual entity graph.

### 5.5.1 Temporal Relatedness Model

The distribution $P(e|e_q, t_q)$ models the entity relatedness between $e$ and $e_q$ w.r.t $t_q$. To estimate $P(e|e_q, t_q)$, we take into account both static and dynamic entity relatedness as

$$P(e|e_q, t_q) = \lambda \frac{R_s(e, e_q)}{\sum_{e'} R_s(e', e_q)} + (1 - \lambda) \frac{R_d(e, e_q, t_q)}{\sum_{e'} R_d(e', e_q, t_q)} \tag{5.5}$$

where $R_s(e, e_q)$ measures the static relatedness between $e$ and $e_q$, $R_d(e, e_q, t_q)$ measures the dynamic relatedness between $e$ and $e_q$ w.r.t $t_q$, and $\lambda$ is a parameter.

**Static Relatedness.** To measure the static relatedness between entities $e$ and $e_q$, i.e. $R_s(e, e_q)$, we use the widely adopted method introduced by Milne and Witten using the Wikipedia links [MW08a], and has been effective in various tasks. The Milne-Witten relatedness is measured as:

$$R_s^{MW}(e, e_q) = \frac{\log(\max(|E|, |E_q|)) - \log(|E \cap E_q|)}{\log |V| - \log(\min(|E|, |E_q|))} \tag{5.6}$$

where $E$ and $E_q$ are the sets of entities that links to $e$ and $e_q$ respectively and $V$ is the set of all entities.

In addition to Milne-Witten, we include the entity embeddings (Section 5.4.3) and define an embedding-based static relatedness measure as the cosine similarity between two corresponding entity vectors:

$$R_s^{Emb}(e, e_q) = \frac{\vec{e} \cdot \vec{e_q}}{\|\vec{e}\| \|\vec{e_q}\|} \tag{5.7}$$

The two static relatedness measures can be combined in linear fashion to provide the final estimation: $R_s(e, e_q) = R_s^{MW}(e, e_q) + R_s^{Emb}(e, e_q)$.

**Dynamic Relatedness.** To measure the dynamic relatedness $R_d(e, e_q, t_q)$, we first associate an activation function that captures the importance of an entity $e$ as a function of time: $\alpha_e : T \rightarrow \mathbb{R}$. This function can be estimated by analyzing the edit history of Wikipedia, in which the more edits take place for an article in a certain time interval, the higher the value of activation function. Other kinds of estimators are to analyze longitudinal corpora such as news archives. In this work, our estimation is based on Wikipedia page view statistics. The normalized value of the activation function of an entity $\alpha_e$ is estimated as follows:

$$A_e(t) = \frac{\alpha_e(t) - \mu_{\alpha_e}}{\sigma_{\alpha_e}} \text{ with } \mu_{\alpha_e} = \mathbb{E}[\alpha_e] \text{ and } \sigma_{\alpha_e} = \sqrt{\mathbb{E}[(\alpha_e - \mu_{\alpha_e})^2]} \tag{5.8}$$

where $\mu_{\alpha_e}$ and $\sigma_{\alpha_e}$ are the mean value and standard deviation of the activation function $\alpha_e$. To assess whether two entities are temporally related, we compare their activity functions. It happens that many entities exhibit very marked peaks of activity at certain points. These peaks are highly representative for an entity. Therefore, we estimate the dynamic relatedness between entities by measuring a form of temporal peak coherence

$$R_d(e, e_q, t_q) = \sum_{t=t_{q_b}}^{t_{q_e}} \max(\min(A_e(t), A_{e_q}(t)) - \theta, 0) \tag{5.9}$$

where $t_q = [t_{q_b}, t_{q_e}]$ is the time interval of interest and $\theta$ is a threshold parameter that is set as 2.5 here to avoid over-interpreting low and noisy values.

### 5.5.2 Topical Relatedness Model

The probability $P(s_q|e, e_q, t_q)$ models the likelihood of observing the text snippet $s_q$ in the relationship between entities $e$ and $e_q$ in the time of interest $t_q$.

For each context $c_i = (t_i, s_i) \in C(e, e_q)$, let $Sim(s_q, c_i, t_q)$ be the similarity between the text snippet $s_q$ and the context $c_i$ w.r.t the time $t_q$. The likelihood of observing $s_q$ in the relationship between $e$ and $e_q$ w.r.t $t_q$ is estimated as:

$$P(s_q|e, e_q, t_q) = \frac{1}{|C(e, e_q)|} \sum_{c_i \in C(e,e_q)} Sim(s_q, c_i, t_q) \tag{5.10}$$

Here we assume the context $c_i$ gives less contribution to the overall relevance of the relation w.r.t the time $t_q$ if its time $t_i$ is distant from $t_q$, then $Sim(s_q, c_i, t_q)$ is estimated as

$$Sim(s_q, c_i, t_q) = \begin{cases} CS(s_q, s_i)\ e^{-\beta|t_q - t_i|}, & \text{if } CS(s_q, s_i) \geq \xi \\ 0, & \text{otherwise} \end{cases} \tag{5.11}$$

where $\xi$ is a fixed parameter, $\beta$ is the decay parameter, $|t_q - t_i|$ is the distance between two time intervals $t_q$ and $t_i$ that is calculated by the distance between their middle points. The component $CS(s_q, s_i)$ measures the similarity between two text snippets $s_q$ and $s_i$. We employ two different methods to estimate $CS(s_q, s_i)$, described below.

**Language Model.** In this method (called *LM-based*), we represent the relation $(e, e_q)$ by a language model, i.e. the distribution over terms taken from text snippets between two entities in the entity graph. Then by assuming the independence between terms in the snippet $s_q$, we obtain the following estimation

$$CS(s_q, s_i) = \prod_{w \in s_q} P(w|\theta_{s_i})^{n(w, s_q)} \tag{5.12}$$

where $n(w, s_q)$ is the number of times the term $w$ occurs in $s_q$, $P(w|\theta_{s_i})$ is the probability of term $w$ within the language model of the snippet $s_i$ which is estimated with Dirichlet smoothing as follows

$$P(w|\theta_{s_i}) = \frac{n(w, s_i) + \mu \cdot P(w)}{\sum_{w'} n(w', s_i) + \mu} \tag{5.13}$$

where $n(w, s_i)$ is the frequency of $w$ in $s_i$, $P(w)$ is the collection language model, and $\mu$ is the Dirichlet smoothing parameter.

**Embedding Model.** The second method is an adaptation of the Word Mover's Distance (WMD) method proposed in [KSKW15]. First, we remove all stop words and keep only

content words in the text snippets. Then, we define the similarity between two text snippets $s_q$ and $s_i$ using a relaxed version of WMD, where each word in $s_q$ (and $s_i$) is mapped to its most similar word in $s_i$ (and $s_q$):

$$CS(s_q, s_i) \propto \frac{1}{2}\left(\frac{\sum_{w \in s_q}\sum_{w' \in s_i} \mathbf{T}_{ww'} \cos(\vec{w}, \vec{w'})}{|s_q|} + \frac{\sum_{w \in s_i}\sum_{w' \in s_q} \mathbf{T}_{ww'} \cos(\vec{w}, \vec{w'})}{|s_i|}\right) \tag{5.14}$$

where $|s_q|$ and $|s_i|$ are the number words in the text snippets $s_q$ and $s_i$ respectively, $\mathbf{T}_{ww'} = 1$ if $w' = \text{argmax}_{w'} \cos(\vec{w}, \vec{w'})$ or 0 otherwise, $\cos(\vec{w}, \vec{w'})$ is cosine similarity between two vectors. The vector $\vec{w}$ and $\vec{w'}$ are the vector embeddings of the words $w$ and $w'$, respectively learned from the Entity Embedding method described in Section 5.4.3. We denote this as the *WMD-based* method.

## 5.6 Experiment Setup

### 5.6.1 Entity Graph Construction

The entity graph we use in the context-aware entity recommendation task is derived from Freebase [BEP+08] and Wikipedia.[4] More specifically, we extract Wikipedia articles that overlap with Freebase topics, resulting in $3,866,179$ distinct entities, each corresponding to one article. To extract the entity activities for the dynamic temporal relatedness model, we use Wikipedia page view counts[5] in the time frame 01/01/2012 to 05/31/2016.

We use the text contents of the articles as the annotated corpus $D$. Note that due to Wikipedia editing guidelines, an article often ignores the subsequent annotations of an entity in the text, if the entity is already annotated before. For example, within the Wikipedia article of entity Brad_Pitt, *Angelina Jolie* is mentioned 32 times but only 5 of these mentions are annotated. Hence, we employ a machine learning method [NBD14] to identify more entity mentions. In average, 12 new entity mentions were added to each Wikipedia article.

To extract text snippets for the graph enrichment, we cleaned and parsed the sentences from the contents, resulting in 108 millions sentences in total. We use Stanford Temporal Tagger[6] to extract temporal patterns from these annotated sentences. For the edges of the entity graphs, we establish the undirected edge $(e_1, e_2)$ if the corresponding Wikipedia article of $e_1$ or $e_2$ (after adding new mentions using [NBD14]) contains a hyperlink to the article of the other.

[4] English Wikipedia dump version dated March 4, 2015
[5] https://dumps.wikimedia.org/other/pagecounts-ez/
[6] http://nlp.stanford.edu/software/sutime.shtml

| Entity | Context | Related entities |
|---|---|---|
| Brad Pitt | humanitarian and political causes | University of Missouri (101), John Kerry (80), Barack Obama (26)... |
| Brad Pitt | career | Fury (2014 film) (1772), Mr. & Mrs. Smith (2005 film) (973), Legends of the Fall (893)... |
| Brad Pitt | personal life | Angelina Jolie (16564), Jennifer Aniston (11306), Gwyneth Paltrow (3383)... |
| Brad Pitt | in the media | Supercouple (798), People (magazine) (126)... |

**Table 5.1** Example of entity-context queries and related entities with the number of clicks extracted from the clickstream dataset

### 5.6.2 Automated Queries Construction

We use the recently published Wikipedia clickstream dataset [WT15] from February 2015 and structural information from Wikipedia for constructing entity-context queries and the Ground Truth.

The clickstream dataset contains about 22 million (referrer, resource) pairs and their respective request count extracted from the request logs of the main namespace of the English Wikipedia. The referrers can be categorized in internal and external traffic; in this work, we only focus on request pairs stemming from internal Wikipedia traffic, i.e., referring page and requested resource are both Wikipedia pages from the main namespace.

Wikipedia articles are collaboratively and iteratively organised in sections and paragraphs, such that each section is concerned with particular aspects or contexts of the entity profile [FMA15]. Each entity mentions within these sections are therefore highly relevant to the source entity in the respective context.

Based on these observations, we propose an automated entity-context query construction using the following heuristics: (i) For each pair of source and target entities, we first extract the section heading where the target entity is mentioned in the source page (ii) The source entity is then used as query entity and the extracted heading is used as context to create a entity-context query; here we filter out noisy headings such as "*further reading*", "*see also*". (iii) We only keep queries for which at least 5 entities are clicked in the clickstream dataset.

To construct the query time, we use the publication time of clickstream dataset, which is February 2015, and convert it to [2015-02-01,2015-02-28].

Table 5.1 presents example queries created for the entity Brad_Pitt. In total, we have $219,844$ entity-context queries. To accommodate the impact of time in the queries, we define the *ratio of views*, denoted by $r$, which is the ratio between the number of times the entity was clicked in February 2015 and in January 2015. The intuition is that if $r$ is very high, the corresponding query entities and topics might have some underlying information interests emerging in February 2015 (for instance, the release of a new movie, etc.). We divide our query set into $4$ subsets based on different value ranges of $r$ (Table 5.2).

| **Query Set** | $Q_{r>0}$ | $Q_{r>1}$ | $Q_{r>5}$ | $Q_{r>10}$ |
|---|---|---|---|---|
| **Number of queries** | 219,844 | 69,489 | 1,263 | 493 |

**Table 5.2** The different set of queries $Q_r$ with varying ratios of interest

**Ground Truth.** For each query in the query set, we establish the ground truth through the click information available in the clickstream dataset. Existing work suggests that the Wikipedia viewing behaviour can be used as a good proxy of entity relevance to current user interest [RFM10, TNK+15]. Transferring this idea to navigational traffic within Wikipedia networks (as they are reflected in the click streams), we can consider an increased navigation between two entities as a signal for the importance of the relationship between the corresponding source and the target entities.

Thus, given an entity-context query, the larger number of clicks a candidate entity gets, the higher related the entity is. Based on this, for each query we take the most clicked entity as the relevant entity, and measure how good recommendation approaches rank the entity using $MRR$ metric. In addition, we extract the top-5 clicked entities for each query to measure the recall. We publish our code and data to encourage future similar research.[7]

**Evaluation Metrics.** To measure the performance of different approaches, we use two evaluation metrics. The first metric is *mean reciprocal rank* (**MRR**) which is computed as

$$MRR = \frac{1}{|Q_{test}|} \sum_{i=1}^{|Q_{test}|} \frac{1}{rank(e_{q_i})} \tag{5.15}$$

where $|Q_{test}|$ is the number of queries, and $rank(e_{q_i})$ represents the rank of the ground truth entity $e_{q_i}$ in the results for the query $q_i$. Notice that a larger MRR indicates better performance.

We also use **recall** at rank $k$ ($R@k$) as another evaluation metric. $R@k$ is measured as the ratio of the retrieved and relevant entities up to rank $k$ over the total number of relevant results. The larger $R@k$ indicates better performance.

### 5.6.3 Baselines

We implemented several baselines to compare to our methods on the task. The first group of baselines are static methods using an ad hoc ranking function without considering the given context. We consider the baselines that only use Milne-Witten or entity embeddings-based relatedness, and the combination. We denote these static methods as **Static**$_{mw}$, **Static**$_{emb}$, and **Static**$_{mw\&emb}$.

The second group of baselines are time-aware methods which are similar to our probabilistic model but without taking into account the search topic $s_q$. We reimplemented the

[7] http://www.l3s.de/~ntran/dycer.html

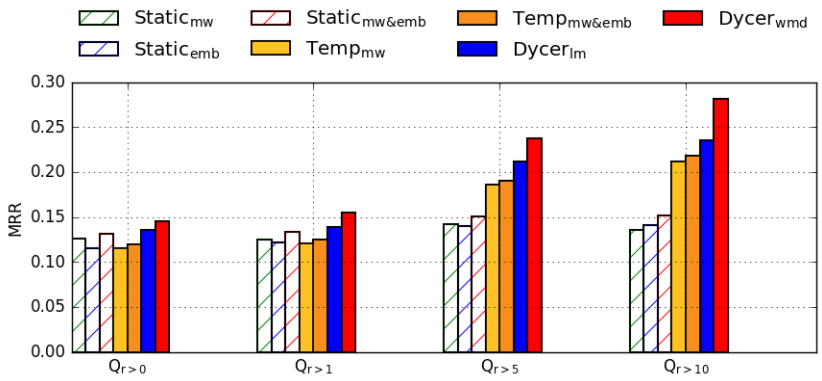



**Figure 5.3** Performance of the different approaches on the different query sets

approach proposed by Zhang et al. [ZRZ16] and extended it by combining the entity embedding and link based similarities to integrate into the model. We denote these time-aware methods as **Temp**$_{mw}$[ZRZ16] and **Temp**$_{mw\&emb}$.

Finally, we denote our methods as **Dycer**$_{lm}$ and **Dycer**$_{wmd}$ where **Dycer**$_{lm}$ uses the LM-based method and **Dycer**$_{wmd}$ uses the WMD-based method for estimating the similarity between text snippets.

**Parameter Settings.** We empirically set the similarity threshold $\xi$ to 0.35, and the decay parameter $\beta$ to 0.5. The Dirichlet smoothing parameter is fixed to 2000, and the parameter $\lambda$ is set to 0.3 by default and will be discussed in detail in the experiments.

## 5.7 Results and Discussion

Figure 5.3 presents a detailed comparison between the $MRR$ for the different methods. The proposed methods outperform the baselines on all query sets. In addition, when increasing the ratio of views $r$, our method progressively improves, with its highest score $MRR = 0.282$ on the query set $Q_{r>10}$. In contrast, the performance of the static methods is not changed much and around $MRR = 0.145$. This conforms the effectiveness of our model in capturing the dynamic contexts. Even without context, our relatedness model ($Static_{mw\&emb}$) already performs better compared to the $Static_{mw}$ and $Static_{emb}$ methods. Interestingly, the time-aware methods gain comparable, even worse results compared to the static methods on the query sets $Q_{r>0}$ and $Q_{r>1}$, however they obtain significantly better $MRR$ scores on the query sets $Q_{r>5}$ and $Q_{r>10}$ . This can be explained by the fact that the entities in $Q_{r>5}$ and $Q_{r>10}$ are more sensitive to time because of high user interests. Furthermore, the adapted implementation $Temp_{mw\&emb}$ outperforms the original method $Temp_{mw}$, which again indicates the effectiveness of the combination of the embedding-based and link-based methods. The best overall performing approach is the WMD-based method $Dycer_{wmd}$. The method performs better than the LM-based method $Dycer_{lm}$, which is due to the fact that the WMD-based method takes into account the

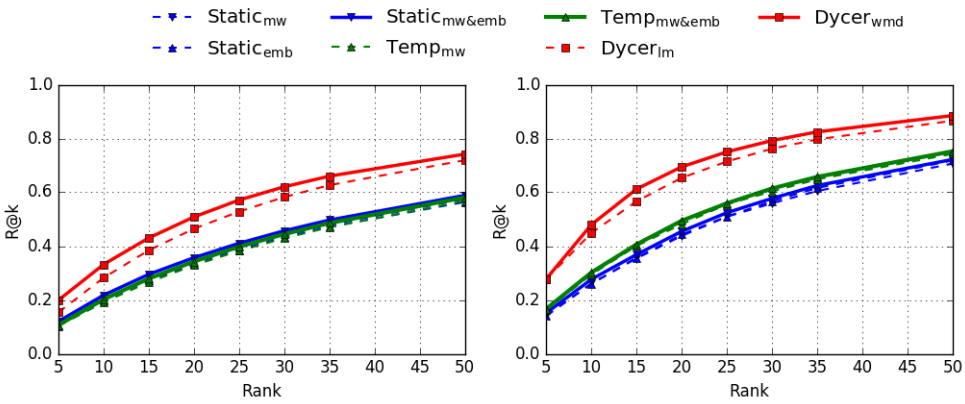



**Figure 5.4** $R@k$ for the different entity recommendation approaches under comparison. (*Left*) All queries $Q_{r>0}$. (*Right*) Queries with high ratios $Q_{r>5}$

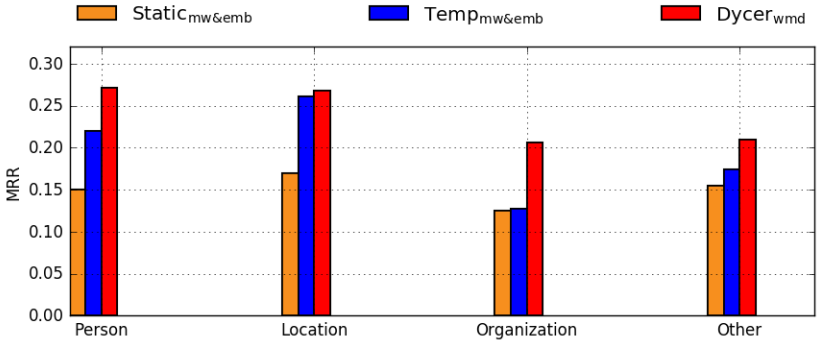



**Figure 5.5** $MRR$ of relevant entity for different query entity types in $Q_{r>5}$ and for different approaches (note, we show the results for the best method in each group)

semantic meaning of words using word embeddings for the textual similarity estimation, while the LM-based method purely uses the surface form of words.

Next, we analyse the recall at rank $k$ ($R@k$) as quality criteria. The results of $R@k$ with varying $k$ for different methods are shown in Figure 5.4. We compute the performance of methods on the different query sets $Q_{r>0}$ and $Q_{r>5}$. Figure 5.4 shows that the proposed methods outperform the baselines on both sets of queries. On the first query set $Q_{r>0}$ the WMD-based method $Dycer_{wmd}$ gains 7.9%, 10.5%, 15.2%, and 14.3% improvements compared to the static method $Static_{mw\&emb}$, and 9.3%, 13.0%, 16.6% and 17.6% improvements compared to the time-aware method $Temp_{mw\&emb}$ when the rank $k$ is 5, 10, 20, and 30 respectively. On the query set $Q_{r>5}$, it even obtains much better improvements. In addition, similar to our findings for $MRR$, the time-aware methods achieve comparable results compared to the static methods overall, but perform considerably better on the query set with the high ratio of views $Q_{r>5}$.

| Method | $\lambda = 0$ | $\lambda = .1$ | $\lambda = .2$ | $\lambda = .3$ | $\lambda = .4$ | $\lambda = .5$ | $\lambda = .6$ | $\lambda = .7$ | $\lambda = .8$ | $\lambda = .9$ | $\lambda = 1$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $Temp$ | 0.1700 | 0.1894 | 0.1902 | **0.1914** | 0.1913 | 0.1913 | 0.1903 | 0.1906 | 0.1888 | 0.1898 | 0.1511 |
| $Dycer$ | 0.2153 | 0.2371 | **0.2372** | **0.2372** | 0.2359 | 0.2366 | 0.2365 | 0.2360 | 0.2361 | 0.2363 | 0.1828 |

**Table 5.3** $MRR$ of relevant entity using the query set $Q_{r>5}$ for different $\lambda$ (with the best results in bold)

In addition, we also compare the performance in different query types, as for each type, users often have different intents and expectations. Figure 5.5 shows the comparison in terms of $MRR$ for four groups of high-level types. It can be seen that the performance differences vary quite noticeably in different type groups. Nevertheless, in all cases, the highest result is achieved by the $Dycer_{wmd}$ approach. Interestingly, for the type "*Person*" and "*Location*" the $Temp_{mw\&wmd}$ approach gains large improvements compared to the static method. One possible explanation for this is that the "*Person*" and "*Location*" entities usually involve in events which highly relate to time. Consequently, taking time into account helps improving the performance. In the case of "*Organization*", the time-aware method does not show any improvement compared to the $Static_{mw\&emb}$ method whereas the WMD-base method still obtains a huge improvement. It demonstrates the usefulness of contextual information for the task.

Table 5.3 shows the impact of $\lambda$ on the performance of the time-aware and the proposed method using the query set $Q_{r>5}$. The $\lambda = 0.3$ yields the best results on average using both methods, which is then used as in our experiments.

While we use Wikipedia for building the model in the experiments, the proposed approach can also use other knowledge bases (e.g. Freebase) to construct the entity graph, and any text collections (e.g. news archives, web archives) can also be used to enrich the entity graph. Our choice of using Wikipedia is driven by the availability of rich and high-quality meta-data in the collection, which enables us to focus on the the effectiveness of the models. In addition, we focus on frequent entities in our experiments, however the proposed method leverages both link structures and the textual representation from the document collection to estimate the entity relatedness; thus we believe that it can achieve good performance with the long-tail entities, as been shown in existing approaches [HSN+12].

## 5.8 Chapter Summary

In this chapter, we focused more on entities, an important aspect of documents, and aimed to address the research question: *How to support explorations of documents by recommending contextually related entities?* (**RQ3**).

For this purpose, we introduced the idea of a contextual relatedness of entities and defined the problem of context-aware entity recommendation for validating the usefulness of contextual relatedness. For tackling the defined problem, we proposed a novel method

based on a statistically sound probabilistic model incorporating temporal and topical context via embedding methods. We demonstrated on a large real-world evaluation set that our method can show the usefulness of contextual entity relatedness as well as the effectiveness of our recommendation method compared to baseline approaches. The related entities can then in turn guide users to effectively explore the consumed texts.

# 6 Conclusion and Future Work

In this thesis, we have studied multiple aspects of the general task of automatically understanding text. In the next section, we draw main conclusions from the findings of the research presented in Chapters 3, 4 and 5. Subsequently, we will discuss limitations and directions for future research.

## 6.1 Conclusion and Contributions

This thesis has addressed three main problems of supporting the interpretations of documents: (i) document representation, (ii) document contextualization, and (iii) document exploration via related entity recommendation.

In Chapter 3, we proposed different approaches for improving document representation. In the first part, we learned that when representing documents as a mixture of topics, the quality of such topics can be improved by tailoring additional domain-specific similar documents before applying topic modeling algorithms. We called this process Topic Cropping and evaluated the topics in terms of coherence, diversity and relevance. In addition, by integrating the automatic evaluation of topic quality we took a first step towards a self-optimizing process of selecting parameters for topic cropping in different settings (i.e., for document collections in different application domains).

In the second part, we studied another form of document representation, i.e. distributed representation. The distributed representation of documents has quickly established itself as one of the most effective techniques for representing a document in a continuous vector space. In this part, we presented multiplicative tree-structured LSTM networks which are capable of incorporating syntactic and semantic information from the text to the tree-structured LSTM architecture and pointed out the usefulness of our models in various downstream applications. Unlike traditional approaches, the proposed models employ not only word information but also relation information between words. Hence, they are more expressive, as different combination functions can be applied for each word. Experimental

results on common document understanding tasks have demonstrated that the models lead to better document representations.

In the last part, we found out that the distributed representation of documents can be further improved with attention mechanism. We studied this effectiveness in the application of question answering. In particular, we proposed Multihop Attention Networks (MAN) for the answer selection task. Our proposed MANs use multiple vectors which focus on different parts of a question to represent the overall semantics of the question and then apply multiple steps of attention to learn representations for the candidate answers. Such representations are then used to select the most suitable answer for input questions. Furthermore, we showed that sequential attention mechanism can be well adapted for the answer selection task. The mechanism allows local alignment information to be used when computing attention weight for each token in a sequence. Experimental results indicated that MAN outperforms state-of-the-art approaches on popular benchmark question answering datasets. Empirical studies also confirm the effectiveness of sequential attention over other attention mechanisms.

In Chapter 4, we learned that fully understanding document requires context knowledge from the time of document creation and finding information about such context is a tedious and time-consuming task. To study this, we introduced the novel and challenging task of time-aware re-contextualization of context with a gap between creation and reading time. To tackle this task, we presented different query formulation methods for retrieving contextualization candidates and ranking methods taking into account topical and temporal relevance as well as complementarity with respect to the original document. Experimental results have proven that our approach can compute relevant and complementing contextualization information with high precision. In the experiments, hook-based query formulation methods have outperformed document-based ones supporting the validity of our contextualization model, and the pre-dominance of query formulation methods relying on several hooks demonstrates the importance of comprehensive contextualization approaches that go beyond the consideration of individual hooks. Furthermore, our experiments have confirmed that complementarity, which is used in the re-ranking step, plays a significant role in contextualization.

In Chapter 5, we introduced the idea of a contextual relatedness of entities and defined the problem of context-aware entity recommendation for validating the usefulness of contextual relatedness. We then proposed a novel method for tackling the defined problem based on a statistically sound probabilistic model incorporating temporal and topical context via embedding methods. The related entities can help users not only further explore the topics discussed in the documents but also better understand the document contents. We evaluated the proposed approach on a real-world dataset, and the results have revealed considerable improvements of our solution over the states of the art.

By carrying out various studies, proposing different methods to deal with the key challenges, and through extensive evaluations, we have made the following noteworthy contributions for improving document understanding: (i) we have proposed novel methods to enhance document representation, and demonstrated its usefulness in various document

understanding tasks, (ii) we have framed the novel and challenging task of time-aware contextualization and proposed a novel approach for retrieving contextualizing information to support the understanding of documents in presence of wide temporal and contextual gaps, and (iii) we have introduced the notion of contextual entity relatedness and investigated its influence on entity recommendation for supporting document exploration.

## 6.2 Future Research Directions

Building on our observations and findings presented in this thesis, we plan to investigate the following aspects of document understanding in the imminent future.

- **Predicting interesting nuggets in the documents**

  In Chapter 4 and 5, we assumed users will highlight concepts or phrases that they wish to gain contextual insights. Therefore, as future directions we foresee work on automatically identifying interesting phrases (nuggets) that are likely marked by users according to whether they would want to know more about them. This problem is in particular challenging, as we need to understand and model the notion of interestingness, that is, to what factors make a concept be an interesting nugget. For example, the semantics of documents might be an important factor: when users read an article about a movie, they are more likely to browse to an article about an actor than to another movie or the director. The predicted nuggets or hooks can then be used in various applications such as augmenting the document with supplementary information, i.e. contextualization, ad placement and content recommendation.

- **Personalized re-contextualization of documents**

  As we discussed in Chapter 4, due to differences in cultural background or domain expertise, users might require different contextualization needs. Therefore, we believe that the personalization of contextualization approaches is an imperative direction to investigate. To tackle this, we might need to deal with several challenges. The first one is how to collect user information for building the profiles of users. The second challenge is how to integrate this information into the contextualization framework. In the future, we plan to investigate the use of social media and user click logs to gather user information, similar to [YMHH14, BMH+15], for constructing user profile and experiment two ways to leverage this information, either using it as additional features for learning-to-rank algorithms or directly utilizing it in a more complicated learning model.

- **Improving document understanding through multimodel learning**

  In this thesis, we have considered texts only as contextual insights, however, images could be an invaluable source of information for providing additional information to improve document understanding. The images can be integrated into document understanding systems via multimodel learning. One possible direction is to utilize

advanced deep learning architectures such as CNNs or LSTMs to map both texts and images into the same semantic space and then learn a combination function to identify images which provide supplementary information to a given text.

- **Integrating commonsense knowledge to document understanding systems**

  In recent years we can build machines that can accurately translate a text between languages, that can identify whether an object appears in an image, and that are capable of recognizing spoken language at high accuracy levels, but which cannot yet answer higher-level questions related to the contents they have processed. For this, it requires machines to not only discover knowledge from documents or images but also have an ability of reasoning beyond the contents. Building a machine that can read any kind of story or watch a movie of any genre and then can answer simple questions about the plot and the characters still remains an very challenging problem. In the near future, we aim to tackle this problem by leveraging massive external or commonsense knowledge with recent advanced neural networks such as dynamic memory networks [XMS16], and investigate this idea based on the bAbI tasks [WBCM15].

# Bibliography


[ABB00] Orly Alter, Patrick O. Brown, and David Botstein. Singular value decomposition for genome-wide expression data processing and modeling. *Proceedings of the National Academy of Sciences (PNAS)*, 2000.

[AR13] Omri Abend and Ari Rappoport. Universal conceptual cognitive annotation (UCCA). In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 228–238. ACL, 2013.

[AR17] Omri Abend and Ari Rappoport. The state of the art in semantic representation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1)*, pages 77–89. ACL, 2017.

[ATT13] Khaled Hossain Ansary, Anh Tuan Tran, and Nam Khanh Tran. A pipeline tweet contextualization system at inex 2013. In *Working Notes for CLEF 2013 Conference*, 2013.

[BAPM15] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 632–642. EMNLP, 2015.

[BBAW10] Klaus Berberich, Srikanta Bedathur, Omar Alonso, and Gerhard Weikum. A language modeling approach for temporal information needs. In *Proceedings of the 32Nd European Conference on Advances in Information Retrieval*, pages 13–25, 2010.

[BBC+13] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In

*Proceedings of the 7th Linguistic Annotation Workshop & Interoperability with Discourse*, pages 178–186, 2013.

[BBEV12] Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. Developing a large semantically annotated corpus. In *Proceedings of the 8th Language Resources and Evaluation Conference*, pages 3196–3200. LREC, 2012.

[BC08] Michael Bendersky and W. Bruce Croft. Discovering key concepts in verbose queries. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 491–498. SIGIR, 2008.

[BCB15] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings of International Conference of Learning Representations*, 2015.

[BCMT13] Roi Blanco, Berkant Barla Cambazoglu, Peter Mika, and Nicolas Torzec. Entity recommendations in web search. In *Proceedings of the 12th International Semantic Web Conference - Part II*, pages 33–48. ISWC, 2013.

[BCV13] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 1798–1828, 2013.

[BEP+08] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 1247–1250. SIGMOD, 2008.

[BGR+16] Samuel R. Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D. Manning, and Christopher Potts. A fast unified model for parsing and sentence understanding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1)*, pages 1466–1477. ACL, 2016.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006.

[BK15] Jimmy Ba and Diederik Kingma. Adam: A method for stochastic optimization. In *Proceedings of International Conference of Learning Representations*, 2015.

[BMH+15] Bin Bi, Hao Ma, Bo-June (Paul) Hsu, Wei Chu, Kuansan Wang, and Junghoo Cho. Learning to recommend related entities to search users. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 139–148. WSDM, 2015.

[BNJ03] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *The Journal of Machine Learning Research*, pages 993–1022, 2003.

[BNT+14] Kerstin Bischoff, Claudia Niederée, Nam Khanh Tran, Sergej Zerr, Peter Birke, Birke, Kerstin Brückweh, and Wiebke Wiede. Exploring qualitative data for secondary analysis: Challenges, methods, and technologies. In *Digital Hummanities DH2014*, 2014.

[BOM15] Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. Fast and space-efficient entity linking for queries. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 179–188. WSDM, 2015.

[BRVB12] Jason L. G. Braasch, Jean-François Rouet, Nicolas Vibert, and M. Anne Britt. Readers' use of source information in text comprehension. *Memory & Cognition*, pages 450–465, 2012.

[BTB14] Elia Bruni, Nam Khanh Tran, and Marco Baroni. Multimodal distributional semantics. *Journal of Artificial Intelligence Research*, pages 1–47, 2014.

[BYB17] Sebastian Brarda, Philip Yeres, and Samuel Bowman. Sequential attention: A context-aware alignment function for machine reading. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 75–80, 2017.

[BYRB17] Sebastian Brarda, Philip Yeres, and Samuel R. Bowman. Sequential attention: A context-aware alignment function for machine reading. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 75–80, 2017.

[CBM16] Danqi Chen, Jason Bolton, and Christopher D. Manning. A thorough examination of the cnn/daily mail reading comprehension task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2358–2367, 2016.

[CFWB17] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, 2017.

[CG14] Gordon V. Cormack and Maura R. Grossman. Evaluation of machine-learning protocols for technology-assisted review in electronic discovery. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 153–162. SIGIR, 2014.

[CHH+17] Qin Chen, Qinmin Hu, Jimmy Xiangji Huang, Liang He, and Weijie An. Enhancing recurrent neural networks with positional attention for question answering. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 993–996, 2017.

[CHL05] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, pages 539–546, 2005.

[CK12] David Carmel and Oren Kurland. Query performance prediction for ir. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1196–1197. SIGIR, 2012.

[CKC+08] Charles L.A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 659–666. SIGIR, 2008.

[CL14] Zhiyuan Chen and Bing Liu. Mining topics in documents: Standing on the shoulders of big data. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1116–1125, 2014.

[CM14] Danqi Chen and Christopher Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 740–750. EMNLP, 2014.

[CTKN14] Andrea Ceroni, Nam Khanh Tran, Nattiya Kanhabua, and Claudia Niederée. Bridging temporal context gaps using time-aware recontextualization. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1127–1130. SIGIR, 2014.

[CTZC02] Steve Cronen-Townsend, Yun Zhou, and W. Bruce Croft. Predicting query performance. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 299–306. SIGIR, 2002.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Mach. Learn.*, pages 273–297, 1995.

[CV07] Rudi L. Cilibrasi and Paul M. B. Vitanyi. The google similarity distance. *IEEE Transactions on Knowledge and Data Engineering*, pages 370–383, 2007.

[CvMG+14] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734. EMNLP, 2014.

[CWB+11] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, pages 2493–2537, 2011.

[CYgL18] Jihun Choi, Kang Min Yoo, and Sang goo Lee. Learning to compose task-specific tree structures. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI-18)*. AAAI, 2018.

[CYT10] David Carmel and Elad Yom-Tov. Estimating the query difficulty for information retrieval. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 911–911. SIGIR, 2010.

[CZL+17] Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang, and Diana Inkpen. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1)*, pages 1657–1668. ACL, 2017.

[DBK+97] Harris Drucker, Christopher J. C. Burges, Linda Kaufman, Alex J. Smola, and Vladimir Vapnik. Support vector regression machines. In *Advances in Neural Information Processing Systems 9*. NIPS, 1997.

[DBL+15] Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah Smith. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, pages 334–343. ACL, 2015.

[DCZ+10] Anlei Dong, Yi Chang, Zhaohui Zheng, Gilad Mishne, Jing Bai, Ruiqiang Zhang, Karolina Buchner, Ciya Liao, and Fernando Diaz. Towards recency ranking in web search. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, pages 11–20. WSDM, 2010.

[DD95] R Desimone and J Duncan. Neural mechanisms of selective visual attention. *Annu Rev Neurosci*, pages 193–222, 1995.

[DDF+90] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of The American Society for Information Science*, pages 391–407, 1990.

[DHS11] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

[DOL15] Andrew M. Dai, Christopher Olah, and Quoc V. Le. Document embedding with paragraph vectors. In *NIPS Deep Learning Workshop*, 2015.

[DSD11] Na Dai, Milad Shokouhi, and Brian D. Davison. Learning to rank for freshness and relevance. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 95–104. SIGIR, 2011.

[dSTXZ16] Cicero dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. Attentive pooling networks. In *CoRR, abs/1602.03609*, 2016.

[DSZ12] Fan Deng, Stefan Siersdorfer, and Sergej Zerr. Efficient jaccard-based diversity analysis of large document collections. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, pages 1402–1411. CIKM, 2012.

[DWT+14] Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. Adaptive recursive neural network for target-dependent twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2)*, pages 49–54. ACL, 2014.

[Elm90] Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, pages 179–211, 1990.

[FDSC16] Jeffrey Flanigan, Chris Dyer, Noah A. Smith, and Jaime Carbonell. CMU at SemEval-2016 task 8: Graph-based AMR parsing with infinite ramp loss. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1202–1206. SemEval, 2016.

[Fir57] J. R. Firth. A synopsis of linguistic theory 1930-55. pages 1–32, 1957.

[FKNV07] Stefan L. Frank, Mathieu Koppen, Leo G.M. Noordman, and Wietske Vonk. Modeling multiple levels of text representation. In F. Schmalhofer & C. A. Perfetti (Eds.)*, Higher level language processes in the brain: Inference and comprehension processes*, pages 133–157, 2007.

[FMA15] Besnik Fetahu, Katja Markert, and Avishek Anand. Automated news suggestions for populating wikipedia entity pages. In *Proceedings of the 24th*

*ACM International on Conference on Information and Knowledge Management*, pages 323–332. CIKM, 2015.

[FTC+14] Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A. Smith. A discriminative graph-based parser for the abstract meaning representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1)*, pages 1426–1436. ACL, 2014.

[FXG+15] Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, and Bowen Zhou. Applying deep learning to answer selection: A study and an open task. In *Workshop on Automatic Speech Recognition and Understanding*, pages 813–820, 2015.

[GLD12] Wei Gao, Peng Li, and Kareem Darwish. Joint topic modeling for event summarization across news and social media streams. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, pages 1173–1182. CIKM, 2012.

[Gol16] Yoav Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, pages 345–420, 2016.

[Got16] Gregory Goth. Deep or shallow, nlp is breaking out. *Communications of the ACM*, pages 13–16, 2016.

[GPS99] Gene Golovchinsky, Morgan N. Price, and Bill N. Schilit. From reading to retrieval: Freeform ink annotations as queries. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 19–25. SIGIR, 1999.

[GR12] John Gantz and David Reinsel. The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east. Technical report, IDC, 5 Speen Street, Framingham, MA 01701 USA, 2012.

[Gra12] Alex Graves. *Supervised Sequence Labelling with Recurrent Neural Networks*. Studies in Computational Intelligence. 2012.

[GS04] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, pages 5228–5235, 2004.

[Har54] Zellig Harris. Distributional structure. *Word*, pages 146–162, 1954.

[HCMB03] Monika Henzinger, Bay-Wei Chang, Brian Milch, and Sergey Brin. Query-free news search. In *Proceedings of the 12th International Conference on World Wide Web*, pages 1–10. WWW, 2003.

[HD10] Liangjie Hong and Brian D. Davison. Empirical study of topic modeling in twitter. In *Proceedings of the First Workshop on Social Media Analytics*, pages 80–88. SOMA, 2010.

[HdRS+11] Jiyin He, Maarten de Rijke, Merlijn Sevenster, Rob van Ommering, and Yuechen Qian. Generating links to background knowledge: A case study using narrative radiology reports. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, pages 1867–1876. CIKM, 2011.

[HHdJ08] Claudia Hauff, Djoerd Hiemstra, and Franciska de Jong. A survey of pre-retrieval query performance predictors. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pages 1419–1420. CIKM, 2008.

[HKG+15] Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, pages 1693–1701, 2015.

[HLLC14] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. Convolutional neural network architectures for matching natural language sentences. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, pages 2042–2050, 2014.

[HO04] Ben He and Iadh Ounis. Inferring query performance using pre-retrieval predictors. In *Proceedings of Symposium on String Processing and Information Retrieval*, pages 43–54. SPIRE, 2004.

[Hof99] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 50–57. SIGIR, 1999.

[HPM13] Christine Howes, Matthew Purver, and Rose McCabe. Investigating topic modelling for therapy dialogue analysis. In *Proceedings of IWCS 2013 Workshop on Computational Semantics in Clinical Text (CSCT)*, pages 7–16, 2013.

[HPQ17] Minghao Hu, Yuxing Peng, and Xipeng Qiu. Reinforced mnemonic reader for machine comprehension. *arXiv preprint arXiv:1705.02798*, 2017.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, pages 1735–1780, 1997.

[HS10] Michael Heilman and Noah A. Smith. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1011–1019, 2010.

[HSN+12] Johannes Hoffart, Stephan Seufert, Dat Ba Nguyen, Martin Theobald, and Gerhard Weikum. Kore: Keyphrase overlap relatedness for entity disambiguation. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, pages 545–554. CIKM, 2012.

[HYB+11] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. EMNLP, 2011.

[II08] Aminul Islam and Diana Inkpen. Semantic text similarity using corpus-based word similarity and string similarity. *ACM Trans. Knowl. Discov. Data*, pages 10:1–10:25, 2008.

[IKN98] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 1254–1259, 1998.

[JD07] Rosie Jones and Fernando Diaz. Temporal profiles of queries. *ACM Trans. Inf. Syst.*, 2007.

[JLZ+11] Ou Jin, Nathan N. Liu, Kai Zhao, Yong Yu, and Qiang Yang. Transferring topical knowledge from auxiliary long texts for short text clustering. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, pages 775–784. CIKM, 2011.

[KB15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *The 3rd International Conference on Learning Representations*. ICLR, 2015.

[KBM11] Nattiya Kanhabua, Roi Blanco, and Michael Matthews. Ranking related news predictions. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 755–764. SIGIR, 2011.

[KFN09] Ralf Krestel, Peter Fankhauser, and Wolfgang Nejdl. Latent dirichlet allocation for tag recommendation. In *Proceedings of the Third ACM Conference on Recommender Systems*, pages 61–68. RecSys, 2009.

[KGB14] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1)*, pages 655–665. ACL, 2014.

[Kim14] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751. EMNLP, 2014.

[Kin98] Walter Kintsch. *Comprehension: A paradigm for cognition*. New York: Cambridge University Press, 1998.

[KIO+16] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1378–1387, 2016.

[KKN+16] Nattiya Kanhabua, Philipp Kemkes, Wolfgang Nejdl, Tu Ngoc Nguyen, Felipe Reis, and Nam Khanh Tran. How to search the internet archive without indexing it. In *Proceedings of the 20th International Conference on Theory and Practice of Digital Libraries*, pages 147–160. TPDL, 2016.

[KL80] V. Klema and A. Laub. The singular value decomposition: Its computation and some applications. *IEEE Transactions on Automatic Control*, pages 164–176, 1980.

[KM03] Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430. ACL, 2003.

[KN10] Nattiya Kanhabua and Kjetil Nørvåg. Determining time of queries for reranking search results. In *Proceedings of the 14th European Conference on Research and Advanced Technology for Digital Libraries*, pages 261–272, 2010.

[KN11] Nattiya Kanhabua and Kjetil Nørvåg. Time-based query performance predictors. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1181–1182. SIGIR, 2011.

[KOM03] Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 48–54, 2003.

[KSH17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, pages 84–90, 2017.

[KSKW15] Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. From word embeddings to document distances. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, pages 957–966. ICML, 2015.

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, pages 436 EP –, 2015.

[LCKC09] Chia-Jung Lee, Ruey-Cheng Chen, Shao-Hang Kao, and Pu-Jen Cheng. A term dependency-based approach for query terms ranking. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, pages 1267–1276. CIKM, 2009.

[LFdS+17] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. In *International Conference on Learning Representations 2017 (Conference Track)*, 2017.

[LFT+15] Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman M. Sadeh, and Noah A. Smith. Toward abstractive summarization using semantic representations. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*. NAACL, 2015.

[LKF10] Yann LeCun, Koray Kavukcuoglu, and Clement Farabet. Convolutional networks and applications in vision. In *ISCAS 2010 - IEEE International Symposium on Circuits and Systems: Nano-Bio Circuit Fabrics and Systems*, pages 253–256, 2010.

[LLJH15] Jiwei Li, Minh-Thang Luong, Dan Jurafsky, and Eduard Hovy. When are tree structures necessary for deep learning of representations? In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2304–2314. EMNLP, 2015.

[LM14] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, pages II–1188–II–1196, 2014.

[LQZ+16] Pengfei Liu, Xipeng Qiu, Yaqian Zhou, Jifan Chen, and Xuanjing Huang. Modelling interaction of sentence pair with coupled-lstms. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1703–1712. EMNLP, 2016.

[LSLW16] Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. Learning natural language inference using bidirectional lstm model and inner-attention. *CoRR*, 2016.

[LWRM14] Cheng Li, Yue Wang, Paul Resnick, and Qiaozhu Mei. Req-rec: High recall retrieval with query pooling and interactive classification. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 163–172. SIGIR, 2014.

[Man15] Christopher D. Manning. Computational linguistics and deep learning. *Computational Linguistics*, pages 701–707, 2015.

[MB16a] Arindam Mitra and Chitta Baral. Addressing a question answering challenge by combining statistical methods with inductive rule learning and reasoning. In *Proceedings of the 13th AAAI Conference on Artificial Intelligence*, pages 2779–2785. AAAI, 2016.

[MB16b] Makoto Miwa and Mohit Bansal. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1)*, pages 1105–1116. ACL, 2016.

[MC07] Rada Mihalcea and Andras Csomai. Wikify!: Linking documents to encyclopedic knowledge. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, pages 233–242. CIKM, 2007.

[MC13] K. Tamsin Maxwell and W. Bruce Croft. Compact query term selection using topically related text. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 583–592. SIGIR, 2013.

[McC02] Andrew Kachites McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[MCCD13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Proceedings of the International Conference on Learning Representations 2013: Workshop Track*. ICLR, 2013.

[MCY17] Jean Maillard, Stephen Clark, and Dani Yogatama. Jointly learning sentence embeddings and syntax with unsupervised tree-LSTMs. *CoRR*, 2017.

[MDM07] Donald Metzler, Susan Dumais, and Christopher Meek. Similarity measures for short segments of text. In *Proceedings of the 29th European Conference on IR Research*, pages 16–27, 2007.

[Mil95] George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, pages 39–41, 1995.

[MKB+10] Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In *INTERSPEECH*, pages 1045–1048, 2010.

[MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[MRW04] M. Marchington, J. Rubery, and H. Willmott. Changing organizational forms and the re-shaping of work : Case study interviews, 1999-2002, 2004.

[MS99] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[MSB+14] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, 2014.

[MSC+13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, pages 3111–3119, 2013.

[MT05] Josiane Mothe and Ludovic Tanguy. Linguistic features to predict query difficulty. In *Proceedings of the ACM Conference on Research and Development in Information Retrieval*. SIGIR, 2005.

[MW08a] David Milne and Ian H. Witten. An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In *Proceeding of AAAI Workshop on Wikipedia and Artificial Intelligence*, pages 25–30, 2008.

[MW08b] David Milne and Ian H. Witten. Learning to link with wikipedia. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pages 509–518. CIKM, 2008.

[NASW09] David Newman, Arthur Asuncion, Padhraic Smyth, and Max Welling. Distributed algorithms for topic models. *Journal of Machine Learning Research*, pages 1801–1828, 2009.

[NAT+17] Dat Ba Nguyen, Abdalghani Abujabal, Nam Khanh Tran, Martin Theobald, and Gerhard Weikum. Query-driven on-the-fly knowledge base construction. *Proc. VLDB Endow.*, pages 66–79, 2017.

[NBB11] David Newman, Edwin V. Bonilla, and Wray Buntine. Improving topic coherence with regularized topic models. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, pages 496–504. NIPS, 2011.

[NBD14] Thanapon Noraset, Chandra Bhagavatula, and Doug Downey. Adding high-precision links to wikipedia. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 651–656. EMNLP, 2014.

[NLGB10] David Newman, Jey Han Lau, Karl Grieser, and Timothy Baldwin. Automatic evaluation of topic coherence. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 100–108. HLT-NAACL, 2010.

[PC98] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 275–281, 1998.

[PGK05] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31:71–106, 2005.

[PGKT06] Matthew Purver, Thomas L. Griffiths, Konrad P. Körding, and Joshua B. Tenenbaum. Unsupervised topic modelling for multi-party spoken discourse. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 17–24. ACL, 2006.

[PKVM17] D. Papadimitriou, G. Koutrika, Y. Velegrakis, and J. Mylopoulos. Finding related forum posts through content similarity over intention-based segmentation. *IEEE Transactions on Knowledge & Data Engineering*, pages 1860–1873, 2017.

[PNH08] Xuan-Hieu Phan, Le-Minh Nguyen, and Susumu Horiguchi. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th International Conference on World Wide Web*, pages 91–100. WWW, 2008.

[PPQ+17] Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wentau Yih. Cross-sentence n-ary relation extraction with graph LSTMs. *Transactions of the Association for Computational Linguistics*, pages 101–115, 2017.

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[PXS18] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*, 2018.

[RCW15] Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, 2015.

[RFM10] Jacob Ratkiewicz, Alessandro Flammini, and Filippo Menczer. Traffic in social media i: Paths through information networks. In *Proceedings of the 2010 IEEE Second International Conference on Social Computing*, pages 452–458. SOCIALCOM, 2010.

[RGH+15] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiský, and Phil Blunsom. Reasoning about entailment with neural attention. In *arXiv preprint arXiv:1509.06664*, 2015.

[RHL16] Jinfeng Rao, Hua He, and Jimmy Lin. Noise-contrastive estimation for answer selection with deep neural networks. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 1913–1916, 2016.

[RK14] Fiana Raiber and Oren Kurland. Query-performance prediction: Setting the expectations straight. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 13–22. SIGIR, 2014.

[RWJ+95] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. Okapi at trec-3. *Nist Special Publication Sp*, page 109, 1995.

[SBMAY13] Richard Socher, John Bauer, Christopher D. Manning, and Ng Andrew Y. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1)*, pages 455–465. ACL, 2013.

[SHMN12] Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211. EMNLP-CoNLL, 2012.

[SLNM11] Richard Socher, Cliff C. Lin, Andrew Y. Ng, and Christopher D. Manning. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on Machine Learning*, pages 129–136. ICML, 2011.

[SM13] Aliaksei Severyn and Alessandro Moschitti. Automatic feature engineering for answer selection and extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 458–467, 2013.

[SM15] Aliaksei Severyn and Alessandro Moschitti. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 373–382, 2015.

[SMH11] Ilya Sutskever, James Martens, and Geoffrey Hinton. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1017–1024. ICML, 2011.

[SP06] Michael Strube and Simone Paolo Ponzetto. Wikirelate! computing semantic relatedness using wikipedia. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2*, pages 1419–1424. AAAI, 2006.

[SPW+13] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642. ACL, 2013.

[SVL14] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, pages 3104–3112, 2014.

[TC18] Nam Khanh Tran and Weiwei Cheng. Multiplicative tree-structured long short-term memory networks for semantic representations. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 276–286, 2018.

[TCKN15a] Nam Khanh Tran, Andrea Ceroni, Nattiya Kanhabua, and Claudia Niederée. Back to the past: Supporting interpretations of forgotten stories by time-aware re-contextualization. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 339–348. WSDM, 2015.

[TCKN15b] Nam Khanh Tran, Andrea Ceroni, Nattiya Kanhabua, and Claudia Niederée. Time-travel translator: Automatically contextualizing news articles. In *Proceedings of the 24th International Conference on World Wide Web*, pages 247–250. WWW Companion, 2015.

[TdRW11] Manos Tsagkias, Maarten de Rijke, and Wouter Weerkamp. Linking online news and social media. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, pages 565–574. WSDM, 2011.

[TdSXZ16] Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. Improved representation learning for question answer matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 464–473, 2016.

[TEPW11] Tran Anh Tuan, Shady Elbassuoni, Nicoleta Preda, and Gerhard Weikum. Cate: Context-aware timeline for entity illustration. In *Proceedings of the 20th International Conference Companion on World Wide Web*, pages 269–272. WWW, 2011.

[TN18a] Nam Khanh Tran and Claudia Niederée. A neural network-based framework for non-factoid question answering. In *Companion Proceedings of the The Web Conference 2018*, pages 1979–1983, 2018.

[TN18b] Nam Khanh Tran and Claudia Niedereée. Multihop attention networks for question answer matching. In *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 325–334, 2018.

[TNK+15] Tuan A. Tran, Claudia Niederee, Nattiya Kanhabua, Ujwal Gadiraju, and Avishek Anand. Balancing novelty and salience: Adaptive learning to rank entities for timeline summarization of high-impact events. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1201–1210. CIKM, 2015.

[TP10] Peter D. Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, pages 141–188, 2010.

[Tra14] Nam Khanh Tran. Time-aware topic-based contextualization. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 15–20. WWW Companion, 2014.

[TSM15] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for*

*Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 1556–ĂŞ1566. ACL, 2015.

[TSO+16] Sho Takase, Jun Suzuki, Naoaki Okazaki, Tsutomu Hirao, and Masaaki Nagata. Neural headline generation on abstract meaning representation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1054–1059. EMNLP, 2016.

[TTN17] Nam Khanh Tran, Tuan Tran, and Claudia Niederée. Beyond time: Dynamic context-aware entity recommendation. In *The Semantic Web - 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 - June 1, 2017, Proceedings, Part I*, pages 353–368, 2017.

[TTT+13] Giang Binh Tran, Tuan A. Tran, Nam-Khanh Tran, Mohammad Alrifai, and Nattiya Kanhabua. Leveraging learning to rank in an optimization framework for timeline summarization. In *SIGIR 2013 Workshop on Time-aware Information Access (TAIA'2013)*, 2013.

[TTTHJ15] Tuan Tran, Nam Khanh Tran, Asmelash Teka Hadgu, and Robert Jäschke. Semantic annotation for microblog topics using wikipedia temporal information. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 97–106. EMNLP, 2015.

[Tur00] Peter D. Turney. Learning algorithms for keyphrase extraction. *Information Retrieval*, pages 303–336, 2000.

[TXZ15] Ming Tan, Bing Xiang, and Bowen Zhou. Lstm-based deep learning models for non-factoid answer selection. *CoRR*, abs/1511.04108, 2015.

[TZB+13a] Nam Khanh Tran, Sergej Zerr, Kerstin Bischoff, Claudia Niederée, and Ralf Krestel. "gute arbeit": Topic exploration and analysis challenges for the corpora of german qualitative studies. In *Exploration, Navigation and Retrieval of Information in Cultural Heritage (ENRICH), Workshop at SIGIR '13*, pages 15–22. SIGIR, 2013.

[TZB+13b] Nam Khanh Tran, Sergej Zerr, Kerstin Bischoff, Claudia Niederée, and Ralf Krestel. Topic cropping: Leveraging latent topics for the analysis of small corpora. In *Proceedings of the International Conference on Theory and Practice of Digital Libraries*, pages 297–308. TPDL, 2013.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 6000–6010. 2017.

[vTP+13] Tadej Štajner, Bart Thomee, Ana-Maria Popescu, Marco Pennacchiotti, and Alejandro Jaimes. Automatic selection of social media responses to news. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 50–58. KDD, 2013.

[WBC+16] Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M. Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. In *Proceedings of International Conference of Learning Representations*, 2016.

[WBCM15] Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. *CoRR*, 2015.

[WJ17] Shuohang Wang and Jing Jiang. A compare-aggregate model for matching text sequences. In *Proceedings of 5th the International Conference on Learning Representations*, 2017.

[WLZ16] Bingning Wang, Kang Liu, and Jun Zhao. Inner attention based recurrent neural networks for answer selection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1288–1297, 2016.

[WM10] Mengqiu Wang and Christopher Manning. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1164–1172, 2010.

[WM12] Sida Wang and Christopher Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 90–94, 2012.

[WN15] Di Wang and Eric Nyberg. A long short-term memory model for answer sentence selection in question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 707–712, 2015.

[WPF+99] Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. Kea: Practical automatic keyphrase extraction. In *Proceedings of the Fourth ACM Conference on Digital Libraries*, pages 254–255. DL, 1999.

[WRS+16] Aaron Steven White, Drew Reisinger, Keisuke Sakaguchi, Tim Vieira, Sheng Zhang, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme.

Universal decompositional semantics on universal dependencies. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1713–1723. EMNLP, 2016.

[WSM07] Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. What is the Jeopardy model? a quasi-synchronous grammar for QA. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 22–32, 2007.

[WT15] Ellery Wulczyn and Dario Taraborelli. Wikipedia clickstream. *Figshare*, 2015.

[WZQ+10] Yafang Wang, Mingjie Zhu, Lizhen Qu, Marc Spaniol, and Gerhard Weikum. Timely yago: Harvesting, querying, and visualizing temporal knowledge from wikipedia. In *Proceedings of the 13th International Conference on Extending Database Technology*, pages 697–700. EDBT, 2010.

[XDYY08] Gui-Rong Xue, Wenyuan Dai, Qiang Yang, and Yong Yu. Topic-bridged plsa for cross-domain text classification. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 627–634. SIGIR, 2008.

[XMS16] Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic memory networks for visual and textual question answering. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, pages 2397–2406, 2016.

[XZS16] Caiming Xiong, Victor Zhong, and Richard Socher. Dynamic coattention networks for question answering. *CoRR*, 2016.

[YAGC16] Liu Yang, Qingyao Ai, Jiafeng Guo, and W. Bruce Croft. anmm: Ranking short answer texts with attention-based neural matching model. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 287–296, 2016.

[YBD+09] Yin Yang, Nilesh Bansal, Wisam Dakka, Panagiotis Ipeirotis, Nick Koudas, and Dimitris Papadias. Query by document. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 34–43. WSDM, 2009.

[YBD+17] Dani Yogatama, Phil Blunsom, Chris Dyer, Edward Grefenstette, and Wang Ling. Learning to compose words into sentences with reinforcement learning. In *Proceedings of the 5th International Conference on Learning Representations*. ICLR, 2017.

[YCMP13] Wen-tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. Question answering using enhanced lexical semantic models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1744–1753, 2013.

[YHBP14] Lei Yu, Karl M. Hermann, Phil Blunsom, and Stephen Pulman. Deep learning for answer sentence selection. In *NIPS Deep Learning Workshop*, 2014.

[YMHH14] Xiao Yu, Hao Ma, Bo-June (Paul) Hsu, and Jiawei Han. On building entity recommender systems using user click log and freebase knowledge. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, pages 263–272. WSDM, 2014.

[YMM09] Limin Yao, David Mimno, and Andrew McCallum. Efficient methods for topic model inference on streaming document collections. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 937–946. KDD, 2009.

[YSXZ16] Wenpeng Yin, Hinrich Schutze, Bing Xiang, and Bowen Zhou. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics*, pages 259–272, 2016.

[YVDCBC13] Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. Answer extraction as sequence tagging with tree edit distance. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 858–867, 2013.

[YYM15] Yi Yang, Wen-tau Yih, and Christopher Meek. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, 2015.

[ZCM02] Yi Zhang, Jamie Callan, and Thomas Minka. Novelty and redundancy detection in adaptive filtering. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 81–88. SIGIR, 2002.

[ZHMP08] Xiaodan Zhu, Xuming He, Cosmin Munteanu, and Gerald Penn. Using latent dirichlet allocation to incorporate domain knowledge for topic transition detection. In *Proceedings of the 9th Annual Conference of the International Speech Communication Association*, pages 2443–2445, 2008.

[ZLG+14] Yadong Zhu, Yanyan Lan, Jiafeng Guo, Xueqi Cheng, and Shuzi Niu. Learning for search result diversification. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 293–302. SIGIR, 2014.

[ZLP15] Han Zhao, Zhengdong Lu, and Pascal Poupart. Self-adaptive hierarchical sentence model. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 4069–4076. IJCAI, 2015.

[ZRZ16] Lei Zhang, Achim Rettinger, and Ji Zhang. A probabilistic model for time-aware entity recommendation. In *Proceedings of the 15th International Semantic Web Conference*. ISWC, 2016.

[ZSG15] Xiaodan Zhu, Parinaz Sobhani, and Hongyu Guo. Long short-term memory over recursive structures. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1604–1612. ICML, 2015.

[ZTBN13] Sergej Zerr, Nam Khanh Tran, Kerstin Bischoff, and Claudia Niederée. Sentiment analysis and opinion mining in collections of qualitative data. In *Proceedings of the 1st International Workshop on Archiving Community Memories at iPRESS 2013*, 2013.