Entity Type Modeling for Multi-Document Summarization: Generating Descriptive Summaries of Geo-Located Entities

Ahmet Aker

A thesis submitted in fulfilment of requirements for the degree of
Doctor of Philosophy

to
Department of Computer Science
University of Sheffield

November 2013

Declaration

I hereby declare that this thesis is of my own composition, and that it contains no material previously submitted for the award of any other degree. The work reported in this thesis has been executed by myself, except where due acknowledgement is made in the text.

Ahmet Aker

Abstract

In this work we investigate the application of entity type models in extractive multi-document summarization using the automatic caption generation for images of geo-located entities (e.g. Westminster Abbey, Loch Ness, Eiffel Tower) as an application scenario. Entity type models contain sets of patterns aiming to capture the ways the geo-located entities are described in natural language. They are automatically derived from texts about geo-located entities of the same type (e.g. churches, lakes, towers). We collect texts about geo-located entities from Wikipedia because our investigation show that the information humans associate with entity types positively correlates with the information contained in Wikipedia articles about the same entity types.

We integrate entity type models into a multi-document summarizer and use them to address the two major tasks in extractive multi-document summarization: sentence scoring and summary composition. We experiment with three different representation methods for entity type models: signature words, n-gram language models and dependency patterns. We first propose that entity type models will improve sentence scoring, i.e. they will help to assign higher scores to sentences which are more relevant to the output summary than to those which are not. Secondly, we claim that summary composition can be improved using entity type models.

We follow two different approaches to integrate the entity type models into our multi-document summarizer. In the first approach we use the entity type models in combination with existing standard summarization features to score the sentences. We also manually categorize the set of patterns by the information types they describe and use them to reduce redundancy and to produce better flow within the summary. The second approach aims to eliminate the need for manual intervention and to fully automate the process of summary generation. As in the first approach the sentences are scored using standard summarization features and entity type models.

However, unlike the first approach we fully automate the process of summary composition by simultaneously addressing the redundancy and flow aspects of the summary.

We evaluate the summarizer with integrated entity type models relative to (1) a summarizer using standard text related features commonly used in summarization and (2) the Wikipedia location descriptions. The latter constitute a strong baseline for automated summaries to be evaluated against. The automated summaries are evaluated against human reference summaries using ROUGE and human readability evaluation, as is a common practice in automatic summarization.

Our results show that entity type models significantly improve the quality of output summaries over that of summaries generated using standard summarization features and Wikipedia baseline summaries. The representation of entity type models using dependency patterns is superior to the representations using signature words and n-gram language models.

Acknowledgements

This work would have never been possible without advice, help and support from many excellent people. I was very fortunate to work under the supervision of Prof. Robert Gaizauskas, and I greatly profitted from our many discussions and meetings, his insightful comments on the work and his enthusiasm for it, as well as from his support in all the ways I needed during this PhD. I am very thankful for these. Thanks also to Prof. Mark Sanderson, Dr. Horacio Saggion, Dr. Guy Brown and Dr. Mark Hepple for important comments on my experiments and reading suggestions. I thank Dr. Trevor Cohn for collaboration and many discussions from which I learned a lot. During my PhD I was a member of the NLP lab of the University of Sheffield, where I was fortunate to work with several great people. I thank Dr. Adam Funk, Dr. Kamutha Swampillai and Dr. Mark Greenwood for their practical help and useful discussions. Dr. Georg Struth and Dr. Kirill Bogdanov from the Verification and Testing group of the Computer Science Department helped me understand some math problems and offered solutions I couldn't think of. I am very grateful for this. Outside my lab and the department I was fortunate to work with and learn from several great people. I thank Dr. Xin Fan, Dr. Elena Lloret, Dr. Laura Plaza and Pino diFabrizio for collaborations which were both fruitful and fun.

My PhD was indirectly funded by the TRIPOD project supported by the European Commission under the contract No. 045335, and and I gratefully acknowledge this support. I also would like to thank Mama (Dr. med. Olga Nesic), Baldiz Edina Kurtic, Baldiz Lejla Kurtic, Dr. Mesude Bicak, Dr. Asmaa El Hannani, Dr. Azza Al-Maskari, Dr. Mariam Kiran, Yasin Efe, Murad Abouammoh, Rafet Ongun and Dilan Paranavithana for participating in the manual evaluations and the tedious model summary collection activities. I think without their help, especially of Mama and Baldiz Edina, I would never have finished my experiments. I also would like to thank Babo, Dr. med. Adem Kurtic, for collecting all my publications on his special shelf. To fill this shelf is a challenging goal but the idea of achieving this one day made me always strive to publish more.

None of this work would have been even initiated, let alone completed without help, support and love from my family. I thank my wife, Emina, who has always given me the most of love, help and support. My daughter, Nur, and my son Dzemal have been a source of inspiration and provided more than enough motivation to continue. Finally, thanks to Him, to my mother, my father, my sisters and my brother.

Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

Automatic text summarization aims to represent the topics found in one or more input documents to the user in a condensed form and so to reduce the time effort the user would spend for reading all the documents. As described in Chapter 2 in extractive text summarization this is achieved by selecting a subset of sentences from the document collection which are concatenated to form the summary. The goal is that the summary should contain the most relevant information from the input documents without unnecessary repetitions, and be coherent and readable.

As outlined in Chapter 2 a summary can be generated from a single document or multiple documents. In both cases it is possible to distinguish between generic and query-focused text summarization. In generic text summarization the summary content is determined only based on the content of the input documents. In the case of query-focused text summarization the summarizer is given a natural language query as input which is used by the summarizer to bias its sentence selection towards the pieces of information closely related to that query. The query can take any form. For instance, the query can be an open-ended question about a person such as “Who is X?”, as formulated in the Document Understanding Conferences (DUC), with *X* being the name of a person (Nenkova & McKeown 2011).

To help extract facts about the person one could consider using a person type model. Such a model might capture what facts are typically provided about persons and also the ways these are described in existing texts. When generating a summary about a specific person *X* the person type model could then be used to bias the summarizer’s sentence selection. In addition, the person type model could be used to mark each sentence with the type of information it contains, such as *date of birth*. This would help the summarizer to compose the summary by selecting sentences with unique facts and thus reduce redundancy within the summary. Finally, the model could be used to order the sentences in the summary. For example, a sentence that contains the information about the date of birth of a person could be marked by the model as preceding the sentence containing the date of death of that person. Applying such relationships between the sentences during the summary composition can lead to more coherent summaries and avoid the common problem that the summary reads like a heap of information without any meaningful connection between the sentences.

In this work we use query-focused text summarization to generate a summary for an entity expressed in the query. Instead of a person we use a geo-located entity in the query. Geo-located entities are static features of the built or natural landscape, for example a building, a bridge, a mountain or a river. We create geo-located entity type models to bias the summarizer's sentence selection but also for redundancy reduction and sentence ordering within the summary. Our models are learned off-line from existing texts about different entities of the same type such as *church, river, mountain, etc.* We apply our summarizer to the task of generating captions for images pertaining to geo-located entities.

1.1 Application Scenario – Automatic image captioning

The number of images available electronically is growing exponentially with the rapid development of online photo sharing services and increasing prevalence of digital cameras and camera phones. Additionally, many legacy photographs and other images are stored or archived. Effective access to these images is only possible if they are searchable, which presupposes that images are indexed and easily identifiable. However, typically, only limited textual information is available with each image, usually in form of a set of keywords assumed to describe an image. Alternatively, it could be that no textual information is provided, but the images are only tagged with geo-coordinates and compass information. Such a small or non-existent amount of textual information associated with an image is of limited usefulness for image indexing, organization and search. What would be useful is a means to automatically generate or augment captions for images on the basis of minimal input information. The generated captions could then be used for indexing purposes.

Automatic image captioning is a challenging task not least because it is not straightforward to decide what to include in a caption. One can capture any kind of object in the universe with an image, so the content of an image can be virtually anything we can see (abstract objects made by photo-montage are left out of consideration). However, most objects are multi-faceted and it is not clear which aspects of the image the caption should address. For example, if we take an

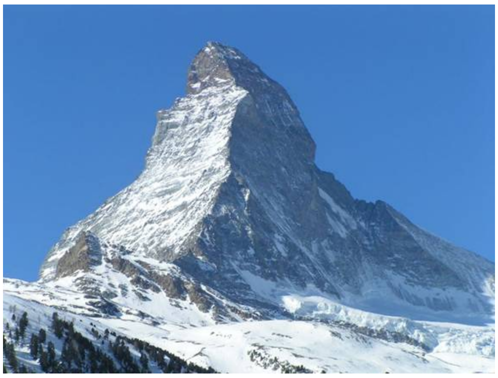


Figure 1.1: Image of Matterhorn taken from http://de.m.wikipedia.org.

image of the *Matterhorn* (Figure 1.1), one could say that the image shows "a mountain covered with snow" or "Matterhorn" if it is known that it is Matterhorn. Alternatively, an interpretative description could be given, e.g. one could say that the image shows challenge, difficulty etc. To make the interpretation the person writing the description needs more knowledge about *Matterhorn*. Therefore, such descriptions can vary from individual to individual and depending on the task of use of the description as each individual will have different knowledge about the object(s) shown in the image and a different interpretation of this knowledge. In addition, the task or context of use will also have impact on the resulting descriptions.

To gain insight into what types of descriptions humans associate with images, a substantial number of investigations into how to classify images or image contents in categories have been carried out. As a result, many classification schemas or analyses dedicated to categorization of image-related information have been proposed (e.g. Armitage & Enser (1997), Eakins (1998), Jörgensen (1998), Jaimes & Chang (2000), etc.). However, the classification schema proposed by Panofsky (1972) and modified by Shatford (1986) shown in matrix in Table 1.1 has been used as the direct or indirect basis for all these works in the field of image classification.

Attempts towards automatic generation of image captions, which mostly address the *what* and *who* facets of the Panofsky-Shatford matrix have been previously reported. They can be divided into *text-based* and *content-based* image caption generation approaches. Text-based approaches generate image captions solely using texts related to the image and do not take image features such as colour, texture, position, etc. into consideration. The content-based methods take image-related texts as well as the image features as input and output image captions based on these two input resources.

Table 1.1: The Panofsky-Shatford mode/facet matrix, Shatford (1986)

Facets \ Modes	Specific Of	Generic Of	About
WHO?	Individually named persons, animals, things,...	Kinds of persons, animals, things	Mythical beings (Generic/Specific), Abstractions manifested or symbolized by objects or beings
WHAT?	Individually named events	Actions, conditions	Emotions, Abstractions manifected by actions, events
WHERE?	Individually named geographic location	Kind of place geographic or architectural	Places symbolized (Generic/Specific), Abstractions manifested by locale
WHEN?	Linear times; dates or periods	Cyclical time; seasons, time of day	Emotions or abstractions symbolized by or manifested by time

To our knowledge the work of Deschacht & Moens (2007) is the only text-based approach in image captioning. The authors automatically generate image captions using associated text such as existing image captions, video transcripts or surrounding text in web pages. They try to identify entities (names of persons and objects) shown in the image. To do this they first detect persons and objects in the associated text by applying automatic named entity recognition. Then they rank the identified entities (person and object names) by assigning them salience weights (the importance of an entity in a text based on word statistics) and visualness measures (how likely an entity will be present in the image). Entities above a threshold of these measures are taken as captions for the image.

In contrast to Deschacht & Moens (2007), several different content-based approaches (Mori et al. 2000, Barnard & Forsyth 2001, Duygulu et al. 2002, Barnard et al. 2003, Pan et al. 2004, Feng & Lapata 2008, Farhadi et al. 2009, Feng & Lapata 2010*c*,*b*,*a*, Farhadi et al. 2010, Yao et al. 2010) use a combination of text resources related to the image and low level image features to generate captions. Although there are differences in problem formulation and application, the common idea presented by these studies is (1) to relate words or greater units such as sentences from the immediate textual context of an image to features or attributes extracted from the image and (2) use the high co-relating text units as description of the image content.

The main drawback of these approaches is that they rely on texts associated with images. However, the associated text may have little semantic agreement with the content of the image, which can result in captions which do not describe the image at all (Marsh & White 2003). Using these "wrong" captions for indexing purposes, for instance, can, according to Purves et al. (2008), be misleading to image retrieval. More importantly, these approaches assume that there exists a text associated with an image. This could be the case if the image has been obtained from a document, for example, or has some existing description or caption describing its content. However, this need not be the case. In the case where there is no document associated with the image or no immediate text that describes its content exists, these techniques are not applicable. Captioning images with little or no associated text information is precisely the scenario which we are concerned with in this work.

1.2 Aims and Scope

We use query-focused, extractive multi-document summarization to generate captions or summaries for geo-located entities. Extractive multi-document summarization aims to present the most important parts of multiple documents to the user in a condensed form by identifying the most relevant sentences from these documents and presenting them in the same form which they have in the original documents (Jones 1999, Mani 2001).

In our system the documents to be summarized are web-documents retrieved using the name of the geo-located entity shown in image (e.g. *Eiffel Tower*) as a query. The resulting image caption has the form of a short description or summary of the web-documents about the place in the image, which distinguishes it from captions in form of lists of keywords generated in much previous work (e.g. (Duygulu et al. 2002, Barnard et al. 2003, Pan et al. 2004, Farhadi et al. 2009)). Therefore, in the remainder of the thesis we use the terms caption, (image) description and summary interchangeably.

Extractive multi-document summarization is presented with several challenges. First of all, it is necessary to distinguish between summary-relevant and summary-irrelevant sentences. This is referred to as *sentence scoring* or *sentence ranking*. Summary-relevant sentences are those which are candidates for inclusion in the final summary and thus should be ranked or scored by the summarizer more highly than the summary-irrelevant sentences. Once the sentences are scored there is the challenge of composing the final summary from these sentences, so that 1) the summary is informative, i.e. contains the most relevant pieces of information without exceeding a predefined length, 2) does not contain redundant information and 3) is fluent to read. Constructing such a summary from a subset of scored sentences will be referred to as *summary composition*.

While related work on automatic summarization usually tackles these challenges independently, in this work we address all jointly. Therefore, the first aim of this work is to investigate how sentence scoring can be improved using models of how people organize and describe knowledge about geo-located entities in their environment. The second goal is then to address the challenges of summary composition by integrating this knowledge. To this end we outline a framework which jointly addresses the maximisation of summary informativeness, redundancy reduction and improving the sentence ordering within a summary.

1.2.1 Sentence Scoring

Previous work has identified several text-based features which are commonly used in sentence scoring (see Chapter 2). These features are "universal" text features, in the sense that they capture a topic and other general properties of a text independently of what a text is about, and what kind of issue the summary should address. They may work well in some domains or genres, but not in others. For example, the *sentence position* feature indicates the position of the sentence within its document, so that e.g. the first sentence in the document gets the highest score, and the score decreases towards the end of the document. This feature has been found useful in the news genre. For news articles the first sentences in the article are worth including in the summary because they usually summarize the entire article (Baxendale 1958, Kupiec et al. 1995, Teufel & Moens 1997). However, Kim et al. (2007) note that this feature was not useful for scoring sentences in biomedical research papers. What may be useful in every domain is to capture how people think about the entities, events and general topics of this domain. This involves identifying the types of information people associate with the topics of the domain and scoring the sentences which address them more highly. Unlike the direct text features, this involves a level of abstraction above the text, as sentences need to be categorized according to the information types they address. However, by doing so the foci of interest within a domain can be captured and addressed in the summaries, which may improve their quality.

For this reason, in addition to using the features commonly used for sentence scoring in previous work, our multi-document summarizer biases the sentence scoring according to an *entity type model*. Using entity type models in sentence scoring is central to our approach. It derives from the fact that humans can categorize things they see in their environment. Cognitive psychology has offered several theories and substantial empirical evidence for existence of categories or concepts and an explanation of what constitutes them (Eysenck & Keane 2005). These theories agree that concepts are characterized by sets of attributes, although they differ in whether a set of attributes is necessary and sufficient to define a concept (defining-attributes theories) or whether the concepts are more fuzzy in their specification in terms of attributes (prototype theories), so that some instances are more representative of a concept than others.

If humans use concepts to organize the knowledge about the world, then we assume that they will have ways to describe these concepts in natural language. We argue that to build a good summary about a geo-located entity (e.g. Eiffel Tower, Westminster Abbey, etc.), we need to select sentences which address the attributes specific to the concept this entity can be categorized into (e.g. tower, church, etc.). This can be achieved if the sentence selection is biased according to an entity type model.

Entity type models are automatically derived from texts describing entities of the same type and contain sets of patterns aiming to capture the ways the entities are described in natural language. We investigate whether entity type models can help our summarization system to perform better sentence scoring.

1.2.2 Summary Composition

We investigate two different ways of composing the final summary from scored sentences: (1) learning a model based on the actual automatic summaries that discriminates between good and bad summaries and (2) integrating summary informativeness, redundancy reduction and summary fluency into the composition process.

For learning the model to discriminate between good and bad summaries, i.e. the prediction model, we investigate the idea of integrating the training of the model and the summary composition within a single framework. The advantage of having such a framework is that the prediction model for distinguishing between good and bad summaries is trained on the actual outputs of the summarizer. Thus the prediction model is trained to optimize the actual summary quality. In this way our work departs form related work where there is no connection between the training and summary composition. In this case the prediction model does not optimize the summary quality but some other peripheral objective, so this does not guarantee that the prediction model will successfully distinguish between good and bad summaries.

Finally, we investigate whether we can formulate the summary composition as a search method that finds the subset of scored sentences in the input documents which combined lead to the most informative, least redundant and most fluent summary.

1.3 Hypotheses and Contributions

This work aims to investigate the ways in which entity type models can be integrated into a multi-document summarization system and to propose a framework which tackles summarization challenges in an integrated manner, with training and search tasks addressed simultaneously. Several hypotheses underlie this work.

1. **Geo-located entity types are characterized by sets of attributes, some of which are type specific and others are shared between types:** Although this seems evident based on the concept definitions made by theories mentioned above, the evidence for categorization of entities into types in these theories comes from domains such as furniture or colours (Rosch 1999). Geo-located entity types such as *mountain, lake, etc.* were investigated by Smith & Mark (2001). The authors analyse how non-experts conceptualize geo-located entities by asking them to suggest entity types after seeing an geo-located entity-related phrase such as *natural Earth formation*. With this study the authors demonstrate that entity types of the natural landscape form a part of human cognitive inventory about geo-located entities. However, they do not investigate what characteristics or attributes humans associate with different entity types. We aim to investigate how humans characterize geo-located entity types in terms of sets of attributes, which may be specific to single entity types (e.g. church), or shared between entity types of similar function (e.g. church and temple).
2. **Entity type models can be derived from existing text resources:** We claim that relevant attributes which characterize entity types are present in existing online collections of text resources, so that entity type models can be derived from these text resources.
3. **Entity type modeling improves sentence scoring:** We claim that incorporating entity type models into a summarizer's sentence scoring will significantly improve summary quality over using standard text related features. However, whether this is the case might also depend on the way entity type models are derived and represented. To test this hypothesis, we evaluate three methods for approximating entity type models: a) signature words, b) n-gram language models and c) dependency patterns.
4. **Entity type modeling can be used for redundancy reduction and improving sentence ordering within a summary:** We hypothesize that it is not only the attributes which characterize an entity type, but also the ordering of these attributes, which is relevant when generating a description of an entity. Therefore, we claim that entity type models can be

used to automatize information ordering within a summary and improve summary coherence. Furthermore, we claim that entity type models can be used to control information duplication and so reduce redundancy within a summary.

5. **Having a model trained on actual outputs of the summarizer improves the summary quality over methods in which such a model is not trained on the automatic summaries:** When there is no connection between training of a model for predicting good and bad summaries and the actual summarizer's output, it is not guaranteed that the model later performs the prediction correctly. By carrying out summary composition with the training of the model intact, we ensure that the model is learned in such a way that the best scoring *whole summary* under the model has a high score under an evaluation metric. Therefore, we hypothesize that these summaries will have higher quality than summaries generated by methods in which training and summary composition are two independent steps.

By testing these hypotheses this work makes a number of novel contributions.

It introduces the idea of entity type modeling to automatic multi-document summarization by investigating whether entity type models can improve both sentence scoring and summary composition. To do this the work contributes a detailed account of how entity type models for summarization can be derived, starting from the description of how humans conceptualize geo-located entities and the investigation of the extent to which this information is retrievable from existing text resources, and ending with the evaluation of three different representation methods for entity type models.

Given that conceptualization is a general feature of human thinking, the idea of entity type modeling is not limited to geo-located entity description generation, but also applies to other domains and genres. Therefore, our technique is suitable not only for image captioning but in any application context that requires summary descriptions of instances of entity classes, where the instance is to be characterized in terms of the features typically mentioned in describing members of a class.

A further contribution of this work is that it aims to address automatic summarization challenges in a single integrated framework. We offer entity type modeling as a method for improving sentence scoring and also use it to reduce redundancy, maximize informativeness and control sentence ordering within the output summaries. Furthermore, our approach unifies training and summary composition and proposes a framework which can be used for generation of optimized summaries.

Finally, the summarization framework developed in this work presents a working summarization system which can be used for automatic image captioning. As such it can be turned into an application for smart phones for example, which automatically generates image descriptions for an image of any geo-located entity taken using the phone. As smart phones are equipped with GPS and compass, the geo-referencing information necessary to generate image descriptions using our system is provided. The application would be interesting for all users seeking to find information about a geo-located entity just by taking its picture.

Furthermore, our system can be used for caption generation or caption augmenting for many pictures archived electronically. As previous work suggests, having more information about the content, i.e. short description, of the place in picture would improve the image's retrievability. In our earlier work we have shown that this is indeed the case (Aker, Fan, Sanderson & Gaizauskas 2012).

1.4 Thesis Structure

This thesis is organized as follows:

In Chapter 2 (General Overview of Automatic Summarization) we review related work in automatic document summarization. The review addresses the two challenges in automatic text summarization: 1) the identification of summary-worthy sentences (sentence scoring) and 2) the generation of informative and readable summaries (summary composition). For sentence scoring, we discuss a number of text related features used in previous work, against which we evaluate the performance of our summaries biased by entity type models. For summary composition, we introduce several previous approaches to informativeness maximization, redundancy reduction and achieving summary coherence. Furthermore, we summarize different approaches for evaluation of automatically created summaries.

In Chapter 3 (Overview of Methodology and Resources) we outline the steps undertaken to address our research questions and the resources needed for summary generation and evaluation. We proceed in four steps: 1) collecting attributes associated with geo-located entity types; 2) analysing existing text resources to establish whether, and if so how these attributes are described in existing texts; 3) exploring methods of geo-located entity type model representation and their exploitation in summarization; and 4) evaluating our summarization techniques. The resources we describe in this chapter are our summarization system, the baseline summaries and the input web-documents from which we generate the automated summaries.

Chapter 4 (How Humans describe Geo-located Entity Types) describes an online user survey which aims to understand what information types (attributes) humans associate with geo-located entity types. This chapters aims to test our first hypothesis that geo-located entity types are characterized by sets of attributes, some of which are entity type specific and others are shared between entity types.

In Chapter 5 (Building Corpora for Geo-located Entity Types) we compare the information contained in Wikipedia articles about geo-located entities to the attributes humans associate with entities identified in Chapter 4. By doing so we test our second hypothesis that entity type models can be derived from existing text resources. This justifies the use of Wikipedia as a resource containing existing geo-located entity descriptions for deriving entity type models. Furthermore, we describe and evaluate a method for the automatic collection of articles about geo-located entity types (so called entity type corpora), from which we derive entity type models.

Chapter 6 (Model Summaries) describes a set of human generated model summaries of geo-located image descriptions. These model summaries are used for computing the ROUGE evaluation metric and are also used in training the summarizer. This data set contains 937 model summaries for 307 different geo-located entities. The model summaries are evaluated based on manual readability evaluation, and are comparable in quality to those reported by the Document Understanding Conference (DUC)[1], a standard challenge for summarization assessment.

Chapter 7 (Entity Type Models for Multi-Document Summarization) introduces the methods of entity type modeling and evaluates the usefulness of entity type modeling in sentence scoring and summary composition. The chapter explains signature words, language models and relational dependency patterns, which we use to represent entity type models derived from entity type corpora. Furthermore, it describes how these models are used by our summarization system to bias sentence scoring and to decide which information types should be included in the summary and in which order, addressing redundancy and sentence ordering. We evaluate the automatically generated summaries obtained by different entity type models using ROUGE and also a human readability evaluation and discuss the results.

In Chapter 8 (Addressing the Challenges of Summary Composition) the fully automatic system for summary generation is presented, with particular emphasis on the challenges of summary composition: redundancy reduction and sentence ordering. We describe a novel framework that

[1] http://duc.nist.gov/

integrates all summarization challenges when producing automated summaries. In this framework the entity type models are used for scoring the sentence, to reduce the redundancy and to derive information order between the sentences in the summary and improve its fluency.

Finally, in Chapter 9 (Conclusion and Future Work) we conclude with a summary and discuss the results of our experiments in relation to our research questions and hypotheses. We also further discuss our contribution to related work in summarization and image captioning and outline possible future directions.

Table 1.2 shows a list of papers resulted from the different chapters of the thesis.

Table 1.2: Papers published within the thesis.

Chapter 4 (How Humans describe Geo-located Entity Types)	Aker, A. and Gaizauskas, R. (2011), Understanding the types of information humans associate with geographic objects, in Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM), pp. 1929-1932.
Chapter 4 (How Humans describe Geo-located Entity Types)	Aker, A, Plaza, L. and Lloret, E. (2013): Do humans have conceptual models about Geographic Objects? A user study. Journal of the American Society for Information Science and Technology (JASIST). ISSN: 1532-2890.
Chapter 5 (Building Corpora for Geo-located Entity Types)	Gornostay, T. and Aker, A. (2009), Development and implementation of multilingual object type toponym-referenced text corpora for optimizing automatic image description, in Proceedings of the 15th International Conference on Computational Linguistics , May 27-31, Bekasovo, Russia.
Chapter 6 (Model Summaries)	Aker, A. and Gaizauskas, R. (2010), Model summaries for location-related images, Proceedings of the International Conference on Language Resources and Evaluation (LREC).
Chapter 7 (Entity Type Models for Multi-Document Summarization)	Aker, A. and Gaizauskas, R. (2009), Summary generation for toponym-referenced images using object type language models, Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP) pp. 6-11.
Chapter 7 (Entity Type Models for Multi-Document Summarization)	Aker, A. and Gaizauskas, R. (2010), Generating image descriptions using dependency relational patterns, in Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 1250-1258.
Chapter 8 (Addressing the Challenges of Summary Composition)	Aker, A., Cohn, T. and Gaizauskas, R. (2010), Multi-document summarization using A* search and discriminative training, in Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, pp. 482-491.
Chapter 8 (Addressing the Challenges of Summary Composition)	Aker, A., Cohn, T. and Gaizauskas, R. (2012), Redundancy reduction for multi-document summaries using A* search and discriminative training. Workshop on Automatic Text Summarization of the Future, Spain, 2012.

CHAPTER 2

General Overview of Automatic Summarization

In this chapter we review related work in automatic document summarization. Although our work focuses on query-focused, extractive multi-document summarization, a number of proposed methods and ideas which we explore in this work have been developed within other summarization paradigms, e.g. single document summarization or abstractive summarization. Therefore, the chapter first briefly introduces the different summarization approaches and paradigms used in previous related work (Section 2.1). Next, the review focuses on summarization techniques (Section 2.2). We distinguish between techniques which perform the summarization without transforming the document structure into another representation (Section 2.2.1) and approaches which first apply a structure transformation of the document and then perform summary generation based on the new representation of the input document (Section 2.2.2). Finally, Section 2.3 describes the ways in which automatically generated summaries can be evaluated.

2.1 Automatic Document Summarization

Automatic document summarization aims to present the most important parts of a document to the user in a condensed form (Jones 1999, Mani 2001). Two different approaches to automatic document summarization have been developed: *extractive* and *abstractive*. In extractive document summarization the most important sentences from the input document are taken as the condensed form of the document and presented to the user in the order they occur in the original document until a stipulated summary length or compression ratio is reaches. The compression ratio indicates the number of sentences or words, relative to the number of complete sentences or words in the text, that the summary should contain. By contrast abstractive approaches aim to rephrase the content identified as relevant in fewer words than the original text.

Extractive and abstractive summaries can be *indicative* or *informative* (Mani 2001, Nenkova & McKeown 2011). Indicative summaries give only an indication about the topic in the input document, so that the reader can decide whether or not to read the entire document based on the indications. Informative summaries aim to capture all the topics presented in the document, so that in an ideal case the reader does not need to read the original documents.

An automated summary can be generated from a single document or from multiple documents. *Single document summarization* can identify sentences relevant to the topic of document or relevant to a given query and present them to the user in a condensed form (Mani 2001). The aim of the *multi-document summarization* is to summarize multiple documents to enable a reader to get a quick overview of their content.

Summary generation in both single and multi-document summarization tasks involves different stages. Jones (1993), for instance, describes a two step approach as: (1) source text representation and (2) summary generation. In step (1) the input text(s) or source is pre-processed and assigned scores. These scores form the basis for deciding which parts of the text(s) are relevant to be included in the automated summary. We refer to this process as *Sentence scoring* or *Sentence ranking*. In step (2) the scores are used to distinguish between relevant and less relevant information and a summary is constructed from the relevant information. We call this stage *Summary composition*.

Summary composition is presented with several challenges, which seem to be more difficult to tackle in multi-document summarization than in single document summarization. As Goldstein et al. (2000) argue, an ideal multi-document summary would be one that contains the main topic shared by all the documents only once. In addition it should also include the information relevant to this main topic which is unique to each document. To achieve this is, according to Goldstein et al. (2000), more challenging than single document summarization.

One of the challenges is the high degree of redundancy – repetition of same information – in information contained in different documents related to the same topic. Multiple documents about the same topic are likely to repeat information, whereas a single document will only contain the topic relevant information once and support it with some background knowledge. Furthermore, if the multiple documents report about related events happened in different time spans, then the ordering of the information becomes an additional problem for a multi-document summarizer. The releasing time of the documents can also cause the problem of information overwriting, i.e. information in earlier released documents may become less important or incorrect. This is less likely to occur in single document summarization.

Finally, a multi-document summary must be coherent and cohesive. Both coherence and cohesion refer to how well-structured and well-organized a description is. A well-structured and well-organized description is the one that does not just contain a heap of related information, but is built from sentence to sentence to a coherent body of information about a topic. In single document summarization the information ordering from the original input document can be taken as a guide to order the selected pieces of information in the final summary. The sentences in the output summary are usually presented in the same order they occur in the input document in order to have a more coherent and cohesive summary. However, if a summary is to be generated from multiple documents, this is no longer possible, so that information ordering for the automated output summary presents a significant challenge.

2.2 Summarization Techniques

In this section we describe different techniques used for summary generation. We distinguish between techniques which do not transform the document into another representation (Section 2.2.1) and methods which require such a transformation as a first step for generating the summary (Section 2.2.2).

2.2.1 Non-Transformational Techniques

Non-transformational techniques are those which do not transfer the document content or structure into another representation but directly work on the document content as it is input to the summarizer. Such methods treat each sentence separately and compose the summary by extracting a subset of sentences from the input document.

As the first step the non-transformational techniques require feature extraction from the input sentences. In the next step for each sentence its feature values are combined to compute a sentence score. In the final step the sentence scores are used in summary composition, i.e. selecting a subset of sentences from the input document. In this final step it is also important to ensure that the resulting summary does not have redundant content and is coherent.

In this section we first discuss a number of features used in previous work. Later in our experimental chapters we compare the performance of our summaries biased by entity type models against summaries produced using only these features. We also review different ways related work has performed feature combination to compute sentence scores. Finally, we describe related work in summary composition and discuss how they solve the redundancy reduction and summary coherence problems that subsequently we address in a single integrated summarization framework. The methods we will describe are used both in single and multi-document extractive summarization.

Features for Sentence Scoring

Word Frequency The work of Luhn (1958) was pioneering in the area of text summarization. Luhn automatically generates literature abstracts using a single feature, word frequency. He argues that the relevant information in a text document is expressed by repetition of certain words that are indicators for the topic in the document. These topic bearing words are referred to as *significant words*. Luhn uses two different boundary values to distinguish between significant and non-significant words, so called high frequency and low frequency borders. Any word whose word frequency in the document is higher or lower than these border values is regarded as non-significant, and the words whose values are between the border lines are the significant ones. Luhn uses the significant and non-significant words for scoring sentences, i.e. any sentence in the document containing significant words is scored higher than a sentence containing non-significant words. The extracts are generated by selecting the sentences with highest scores.

By setting the lower and upper borders Luhn (1958) aims to avoid considering very frequent words (e.g. non-content words such as *the*, *a*, etc.) or very rare words in sentence scoring. Both the lower and the upper border are pre-defined based on observation of word frequency distributions in the input documents.

Sentence Position The position of each sentence in the document is a further sentence scoring feature which was introduced early in text summarization research by Baxendale (1958) and continues to be widely used in more recent studies. Baxendale (1958) regards the first and the last sentences of a paragraph as summary relevant. He reports a survey where 200 paragraphs were manually analyzed and concludes that in 85% of the paragraphs the topic sentence (sentence that is capturing the topic) was in the first position and in 7% of the cases in the final position. Edmundson (1969) who works with scientific documents uses sentence position in the same way as Baxendale (1958) and introduces further positions in the document in which the sentences are regarded as more salient than others. Specifically, he assigns extra rewards to sentences which occur under certain headings such as introduction, purpose or conclusion. In both studies the common hypothesis is that sentences occurring in the specified positions are more likely to summarize the topic of the document and are thus relevant for inclusion in an automated summary.

Lin & Hovy (1997), however, argue that documents in different genres and domains have different structures, so that it could be difficult to find the relevant sentences in the documents with predefined positions. Using a training set containing documents with topic keywords and manually generated extracts, the authors automatically learn the positions of relevant sentences. For each sentence, a *sentence position yield* is computed which holds information about how many distinct words the sentence shares with the topic words and the words from the manually generated extracts. Then for each document an *optimal position policy* (OPP) is generated that combines the information about the paragraph number (P_m) and the position of a sentence (S_n) within this paragraph (P_m, S_n). The same OPPs resulting from different documents are added together, and each OPP is sorted according to the sentence yield score. The authors compare the generated OPPs with OPPs obtained from unseen data from the same genre and domain. They also used the trained OPPs to generate extracts from the unseen data and compare them with the extracts from the training data. Both comparisons indicate high correlations, so the authors argue that OPPs are a way of dynamically learning the salient sentence positions in documents regardless of what genre and topic the documents are from. The authors integrated the idea of OPP into their summarization system SUMMARIST (Hovy & Lin 1998).

Cue Phrases/Words Edmundson (1969) uses different sets of cue words (*bonus words* that are positively relevant, *stigma words* that are negatively relevant and *null words* that are irrelevant) to assign scores to sentences. The sentence score is computed by a weighted combination of bonus, stigma or null words scores appearing in the sentence. Some example cue words used by the author are *significant*, *impossible* and *hardly*.

Units larger than words are investigated by Pollock & Zamora (1975). They use phrases as cues to perform sentence rejection and selection. The authors use a hand-crafted phrase list that is matched against the input document sentences extracted from scientific papers. Sentences which match positive marked phrases such as *our work, reported here* or *this study, present work* are kept for further processing, and sentences which match negative marked phrases such as *previously* or *previous work* are deleted.

Title Words Edmundson (1969) also introduces the idea of using title words as a sentence scoring feature, scoring sentences that contain words from the title and subsection headings more highly than sentences that do not include these terms. The rationale behind this is the hypothesis that writers reuse words from the title and subsection headings in subsequent sentences when they write their articles.

Recently reported analyses of relatedness between titles and content in news articles confirm this hypothesis, suggesting that title words are indeed a useful feature for text summarization. Lopez et al. (2011) analyzed 300 titles of news articles and showed that 66% of the title words occur in the articles. A related investigation was performed by Aker, Kanoulas & Gaizauskas (2012) aiming to collect comparable corpora in different languages for statistical machine translation. The authors use titles to compare news articles in different languages, translating them into English to assess their comparability, i.e. whether or not they are about the same topic. They perform this comparison on titles only, as well as on the entire news articles. Their results suggest that comparability between the titles entails comparability between documents, leading to the conclusion that titles are representative of the entire news document content.

Key Phrases Kutlu et al. (2010) experiment with generic single document summarization using key phrases. The authors work with Turkish texts and extract the base noun phrases which contain head noun and zero or more pre-modifying adjectives and nouns from the input document as key phrases. To obtain the key phrases the base noun phrases are extracted through a noun phrase extraction toolkit reported in Kalaycilar & Cicekli (2008) and then scored and filtered based on frequency in the document and length in words. Sentences are scored according to the following equation:

$$Score_S = \frac{|KPs\text{found in}S|}{|nouns\text{found in}S|} \tag{2.1}$$

where *S* is a sentence in the input document, *KP* is a key phrase. The denominator counts the total number of nouns in the sentence *S*. For scoring the sentence only the 10 most frequent *KPs* extracted from the entire document are used. By limiting the number of key phrases to a small number the authors ensure that they score only the topic related sentences higher than the topic unrelated ones.

Centroid of Document Cluster In the centroid method each document in a set or cluster of documents is represented by a vector of terms, which is also called the vector space model of a document (Salton et al. 1975). The vectors do not contain the actual terms but numerical values representing the significance of terms in the document. The common representation of terms within the vector space model is the *tf*idf* measure (Manning et al. 2008). To compute *tf*idf* the following equation is used:

$$(tf * idf)_{i,j} = tf_{i,j} * idf_i \tag{2.2}$$

where $(tf * idf)_{i,j}$ is the $tf * idf$ value of word i in a document j. The term frequency tf_{ij} is the frequency count of a word i occurring in the document j.

The inverse document frequency, idf_i, is a measure of a term *i*'s distribution over a set of documents (Salton & Buckley 1988). It is computed as:

$$idf_i = \log \frac{|D|}{|\{d : w_i \in D\}|} \tag{2.3}$$

where $|D|$ represents the corpus size (number of documents in the corpus D), and the denominator is the number of documents where word i occurs.

After representing each document d with a vector **d**, the centroid of the document cluster D can be computed as follows:

$$C = \frac{1}{|D|} \sum_{d \in D} \mathbf{d} \tag{2.4}$$

As shown in Equation 2.4 the centroid of a document cluster is the sum of individual document vectors divided by the total number of documents in the cluster.

Radev et al. (2004) use the cosine metric (see below) and compares every sentence in the input documents to the centroid. The resulting similarity score is used to score the sentence.

Query-focused or Query-based Method In the query-focused or query-based method a query is input along with the documents to be summarized to the summarizer. The query represents the user's interest in the input documents expressed through, e.g. a list of words. Each sentence in the input documents is scored according to its similarity to the query (see e.g. Saggion & Gaizauskas (2004)). One of the most widely used similarity metrics between the query and a sentence is the *cosine similarity* measure introduced by Salton & Lesk (1968). The cosine similarity requires that both the query and the sentence are represented as vectors and gives the cosine angle between these two vectors. If the angle is *1*, the query and the sentence are identical, while *0* indicates that these two text units have zero words in common. Similarly to the centroid method described earlier, query and sentence can be represented as vectors of *tf*idf* term weights. Equation 2.5 shows the cosine similarity calculation between a query q and a sentence *S*.

$$cos(q,S) = \frac{\sum_{i}(tf * idf)_{i,q} * (tf * idf)_{i,S}}{\sqrt{\sum_{i \in m}((tf * idf)_{i,q})^2} * \sqrt{\sum_{i \in n}((tf * idf)_{i,S})^2}} \tag{2.5}$$

The computation of vector similarities is affected by the way the vector space model is constructed. If, for instance, words are taken as they occur in the input texts, then this will have an impact on the similarity results, as morphological variants of the words will be regarded as dissimilar. To avoid this problem stems or lemmas (e.g. *go* is the lemma of *going*) of the words can be taken or knowledge from different thesauri can be applied to link words based on different relations such as synonymy. Linking words has been proposed for highlighting similarities between syntactically different but semantically equivalent text units (Resnik 1995, Jiang & Conrath 1997, Lin 1998, Hatzivassiloglou et al. 2001, Banerjee & Pedersen 2003, Gurevych & Strube 2004, Erkan & Radev 2004, Mihalcea et al. 2006).

Feature Combination for Sentence Scoring

The features are used to score sentences in the input text(s), and the scores are used to decide which sentences to include in the final summary. However, if more than one feature is used to score sentences, the question arises as to how to combine the different scores returned by the different features to compute a final score.

In the literature the linear combination of all feature scores is the most widely used approach for this purpose (Edmundson 1969, Radev et al. 2001, Saggion & Gaizauskas 2004, Zajic et al. 2005). Edmundson (1969), for instance, uses the following formula to compute a final score for each sentence.

$$Score(S) = w_1 * location + w_2 * cue + w_3 * key + w_4 * title \tag{2.6}$$

The symbols w_1 to w_4 are the weights for the four sentence features: location, cue, key and title. They express how strongly a single feature contributes to the final score. Edmundson (1969) sets the values of the feature weights manually. However, choosing the weight values manually, has the disadvantage that other possible combinations of weights which may lead to better results are not tested. To achieve the best summary results all combinations of the weights have to be tested. However, manual value selection for the weights is a time consuming task and makes it almost impossible to cover all possible value combinations and select the best one. Therefore, attempts to determine the contribution of scoring features to the final sentence score automatically have been explored.

Kupiec et al. (1995), for instance, experimented with classification machine learning methods and used a Naive Bayesian classifier to learn the combination of their sets of features. They showed that combining all features using the classifier resulted in better results compared to using the features separately. Following Kupiec et al. (1995), other machine learning approaches such as Decision Trees, Hidden Markov Models (HMM), Neural Networks, Support Vector Machines (SVM) etc. have been investigated to classify a sentence as summary-worthy or not (Chuang & Yang 2000, Mani & Bloedorn 1998, Zhou & Hovy 2003, Hirao et al. 2002). Summary-worthy sentences are then used to generate the final summary.

A classifier-based approach only decides if a sentence is summary-worthy or not. However, since there might be more summary-worthy sentences than the final summary should contain, it is difficult to decide which ones to include in the final summary. To avoid this problem, Learning To Rank (LTR) methods have been applied to the problem of combining features (Amini et al. 2005, Amini & Usunier 2009, Fisher & Roark 2006, Toutanova et al. 2007, Wang et al. 2007, Metzler & Kanungo 2008). In LTR sentences are ranked in, e.g., descending order so that the top ranking sentence is regarded as the most summary-relevant one and the sentence at the bottom is considered as least summary-relevant. A summary can be then constructed by selecting the n top ranking sentences.

Another approach to sentence ranking is to use regression methods, as investigated by Ouyang et al. (2011). As a regression model they used Support Vector Regression (SVR) (Vapnik 2000) and showed that it outperformed classification (SVM (Vapnik 2000)) and LTR (ranking SVM (Joachims 2002*b*)) methods.

Summary Composition

In the summary composition phase a subset of sentences from the input documents are concatenated to form a summary. In this phase the sentence score computed as described in the previous section is used to bias the sentence selection towards sentences with high scores. However, this step has to also ensure that while performing the sentence selection the final summary is (1) the most informative summary, (2) that has least redundancy and (3) is most fluent to read. In this section we sketch each of these challenges and will revisit them in Chapter 8.

Finding the most informative summary A summary must be shorter than the input text(s). How short a summary should be is usually controlled by the number of words or sentences the summary should contain (Paice 1980) or by the compression rate (Radev et al. 2004). The compression rate, expressed as a percentage, specifies what proportion of the input text(s) should be used in the summary.

These constraints on the summary length mean that typically not all sentences identified as summary-worthy based on their scores can be included in the summary. In selecting the sentences which should be included in the output summary, the first challenge is to create the most informative summary, i.e. a summary which contains the most relevant pieces of information without exceeding a predefined length.

The entire task of summary composition and thus also its subtask of maximizing summary informativeness can be regarded as a search problem, in which the best sentences, in this case the sentences which maximize the summary informativeness, are selected from the entire set of summary-worthy candidate sentences. Related work has addressed the search problem using *non-inference* and *inference* techniques.

The non-inference techniques assume that including sentences with highest scores will solve the problem of summary informativeness. Therefore, the simplest and the most widely used non-inference approach to finding the most informative summary is to perform the search using a greedy algorithm which selects each sentence in a decreasing order of sentence score until the desired summary length is reached (see e.g. (Paice 1980, Saggion & Gaizauskas 2004, Saggion 2005)). Alternatively, some studies use heuristic strategies and prefer sentences based on position in document or lexical clues (Edmundson 1969, Brandow et al. 1995, Hearst 1997, Ouyang et al. 2011). A third group of non-inference approaches applies genetic algorithms to the search problem (Orasan 2003, Alfonseca & Rodríguez 2003, Liu et al. 2006, Riedhammer et al. 2008). All non-inference approaches only ensure that highest scoring sentences are included in the summary. However, by doing this it is not guaranteed they will also lead to the best summary as this strategy may lead to highly redundant summaries particularly in multi-document summarization.

In the inference-based approach more sophisticated techniques are investigated to address the search problem. McDonald (2007), for instance, addresses the search problem using Integer Linear Programming (ILP). In his ILP problem formulation he adopts the idea of Maximal Marginal

Relevance ((Carbonell & Goldstein 1998), see below)) to maximize the amount of relevant information in the summary and at the same time to reduce the redundancy within it. Others have also addressed the search problem using a variation of ILP (Gillick & Favre 2009, Gillick et al. 2009), as well as using different approaches such as stack decoding algorithms (Yih et al. 2007) and submodular set function optimisation (Lin & Bilmes 2010). McDonald (2007) shows that his inference-based approach performs significantly better than a non-inference approach where the summary composition is performed using a greedy algorithm.

Redundancy Detection Excluding redundant information from the summary can make the summary not only more informative but also can improve its readability. To perform redundancy detection different approaches have been investigated. The most common method is the Maximum Marginal Relevance (MMR) method (Carbonell & Goldstein 1998). The MMR method originates in Information Retrieval (IR). It takes into account the diversity of information with respect to a topic (query), i.e. it measures the information gain a new document brings to the topic with respect to what has already been seen. The computation of MMR is shown in Equation 2.7:

$$MMR = \arg \max_{D_i \in R \setminus S} [\lambda(Sim_1(D_i, Q)) - (1 - \lambda) \max_{D_j \in S}(Sim_2(D_i, D_j))] \tag{2.7}$$

where *S* is the set of already seen documents and *R* $\setminus$ *S* the set of documents not seen yet, Q is the query to an IR system. λ is the controlling weight between the two summations/substractions and its value can be either set experimentally or learned using machine learning techniques. The terms Sim_1 and Sim_2 are the similarity metrics. Various similarity metrics can be used here, e.g. cosine similarity. Furthermore, Sim_1 and Sim_2 can be both same or not. Carbonell & Goldstein (1998) use MMR in text summarization and treat the sentences in the input text as documents in an IR setting. For each unseen sentence the authors compute the similarity between the current sentence and a user generated query and already seen sentences (sentences in the summary). For both Sim_1 and Sim_2 the authors use the cosine similarity. The selected sentences in the summary are presented in the same order as they appear in the input text. The authors report that MMR works better for long documents as they contain apart from the topic relevant sentences, also many topic irrelevant ones. The authors achieved the highest score among 15 different systems in the SUMMAC summarization competition (Mani et al. 1999).

In Saggion & Gaizauskas (2004) and Saggion (2005) redundancy detection is performed before sentences are selected into the summary. The authors compute a similarity metric for each sentence which decides whether a sentence is distinct enough from the sentences already selected to be included in the summary or not. They use the following formula to compute the similarity between two sentences:

$$NGramSim(S_1, S_2, n) = \sum_{i=1}^{n} w_i * \frac{ngram(S_1, i) \bigcap ngram(S_2, i)}{ngram(S_1, i) \bigcup ngram(S_2, i)} \tag{2.8}$$

where n specifies maximum size of the n-grams to be considered, ngram(S_X, i) is the set of i-grams in sentence S_x and w_i is the weight associated with i-gram similarity. Two sentences are similar if *NGramSim(S_1, S_2, n)* $> \alpha$, where α is a threshold which can be varied from 0 to 1. With this approach the authors can calculate how many n-grams the current sentence has in common with the previous selected ones.

Lloret et al. (2008) use textual entailment to automatically identify sentences within a document which can be inferred from others and remove them from the document. A textual entailment relation holds whenever the truth of one text fragment follows from another text (Glickman 2006). After the entailment step the authors apply a word frequency feature to score the remaining sentences within the document and extract a summary using the top ranked sentences. They experiment with DUC summarization data sets and report high improvements in summary quality compared to other systems which are also run on the same data sets.

Summary Coherence The third challenging task of summary composition is achieving the coherence of a human authored summary. To produce coherent summaries it is important to ensure that the information flow within a summary follows a logical order. This is especially important when the pieces of information included in the summary come from multiple documents.

In early works, chronological ordering was investigated for sentence ordering (McKeown et al. 1999, Lin & Hovy 2001, Radev et al. 2004). In the chronological approach sentences within a summary are ordered according to the publication date of the input documents, so that the sentences coming from the documents published earlier occur earlier in the summary than the sentences which come from documents published later. Barzilay et al. (2002) also use chronological sentence ordering. They first extract groups of sub-topics from the input news documents and then arrange the sentence within these groups chronologically. However, they show that ordering sentences chronologically does not necessarily lead to coherent summaries.

Lapata (2003) and Soricut & Marcu (2006) experiment with probabilistic models to learn the order of the sentences in a single text document. In Lapata (2003) shallow features such as word frequency are extracted from each sentence in the document. These features are used to compute the likelihood that a sentence follows a preceding one. Soricut and Marcu use the IBM Model 1 (Brown et al. 1993) to model the local coherence between two adjacent sentences. Another probabilistic information ordering approach is described by Barzilay & Lapata (2005, 2008). Their approach is motivated by Centering Theory (Grosz et al. 1995), which states that adjacent sentences in a text are likely to focus on the same entities. In their approach single documents are used to learn sentence ordering based on the entity shifts between the adjacent sentences. The sentences in each document are mapped into a grid. The rows within the grid represent the sentences and the column the "discourse entities" identified within the document sentences. These discourse entities are phrases marked as subject (*S*), object (*O*) and neither subject nor object (*X*) of each sentence. For mapping each sentence in a document to the grid, the list of discourse entities extracted from that document is examined and each entity occurrence within the sentence is checked. If the sentence contains the entity, then the column that crosses with the row of the sentence is marked with the entity type (*S/O/X*) or with a gap (-) indicating the absence of an entity within that sentence. Using these grids the authors learn a probabilistic model that describes the entity transitions within a single document. Elsner & Charniak (2011) extend the entity model of Barzilay and Lapata by adding additional features which better capture the likelihood that an entity will be mentioned in following sentences. The authors report better results with the extended model compared to the original entity model set-up.

The work reported by Lin et al. (2011) also performs sentence ordering based on knowledge extracted from the adjacent sentences in a single document. However, instead of using entity flow, Lin et al. (2011) use discourse relation transition between the sentences. The discourse relations are *Temporal, Contingency, Comparison,* and *Expansion*. Using this idea the authors achieve better results than those reported in Barzilay & Lapata (2005, 2008).

More recently, Bollegala et al. (2012) use feature sets or experts such as *chronology, probabilistic, topical-closeness, precedence* and *succession* to perform sentence ordering where the sentences come from multiple documents. They use manually ordered model summaries to learn the optimal combination of the different experts and use the learned model to order sentences coming from multiple documents.

Louis & Nenkova (2012) use a coherence model based on syntactic patterns to capture the intentional structure of text. They showed that their model is able to predict the coherence of abstract, introduction, and related work sections of academic conference articles.

These approaches concentrate on the task of reordering the sentences in a single document after its sentence order has been permuted. However, this is different from the sentence ordering task in text summarization, where the challenge is to find the best sentence order when only a subset of the sentences in the document is used. Thus, the assumption that the number of sentences is equal at all times de-links the sentence ordering task from the sentence extraction one that aims to identify only a subset of sentences to use in the final summary. This assumption makes the reported approaches not directly applicable to automatic single document summarization and even less applicable when the sentences are extracted from multiple documents in a multi-document summarization setting. Therefore, although it is claimed that the reported studies can offer useful insight into information ordering in automated summaries, summarization-specific methods are needed to address this task.

For ordering the sentences in a single document summary the sentences can be presented in the same order as they appear in the input document (Carbonell & Goldstein 1998). This original order is more likely to produce a fluent summary than any other ordering of the summary sentences. In a multi-document summarization scenario, Barzilay & Lee (2004) apply Hidden Markov Models (HMMs) to address sentence ordering within a summary. The states within the HMM are topics, represented as n-gram language models, generated from topic specific texts. The topics are clusters of sentences taken from the input documents. For clustering the sentences the cosine similarity is used. The authors use transition probabilities between the states for sentence ordering. A similar HMM approach is followed by Fung & Ngai (2006). However, determining the order of the summary sentences based on the order of the input topics can be unreliable if the input texts do not follow a common particular topic transition.

Rhetorical Structure Theory (RST) Mann & Thompson (1988) relations can also be used to perform sentence ordering. The $ELABORATION$ relation between the sentences is used to signal coherence (cf. e.g. Hovy (1988)). However, related work has argued that this relationship is not suitable for coherence and must be replaced by entity-focused approaches (Walker et al. 1998).

2.2.2 Transformational Techniques

Transformational techniques transfer the document content into another representation and use the new representation to extract the summary. In this section we describe different techniques which perform summary composition by using document structure transformation.

Latent Semantic Analysis (LSA)

Latent Semantic Analysis transfers the input document into a matrix representation. It uses the matrix to capture the similarity of meaning of words and sentences. The columns of the matrix are the sentences from the input text and the rows the words. The size of the matrix depends on the size of the input text. The more sentences and distinct words the input text has, the greater is the dimensionality of the matrix. A matrix with a large dimensionality implies also that the matrix has a lot of noise or less summary worthy sentences which can be discarded from the matrix. To keep the matrix small and to implicitly reduce noise, LSA applies Singular Value Decomposition (SVD) analysis:

$$A = U\Sigma V^T \tag{2.9}$$

where *A* is an *n x m* matrix representing the input text as noted above. The cells of the matrix contain feature scores, e.g. how many times a word occurs in a sentence. *U* is a *n x n* matrix of singular left vectors arranged right to left from largest corresponding eigenvalue to the least (i.e. the columns of *U* are the orthonormal eigenvectors of AA^T). This *U* matrix can also be interpreted as the topic matrix with the columns representing the topics. The Σ matrix is an *n x m* diagonal matrix containing the square roots of eigenvalues from *U* in descending order. These values are the weights for the topics in matrix *U*. The matrix V^T is a *m x m* matrix of right singular vectors arranged similar to *U* from right to left according to the eigenvalues (i.e. the columns of V are orthonormal eigenvectors of A^TA). In this matrix the sentences from *A* are represented in the rows using the topics from *U*.

A less important topic has a smaller weight in the Σ matrix. If the goal is to discard *k* topics, one can reduce the dimension of the matrixes by deleting the last *k* rows from *U*, the last *k* columns from Σ and the last *k* rows from the V^T matrix. A step-by-step guide for computing SVD and performing dimension reduction is given by Baker (2005).

Gong & Liu (2001) use the V^T matrix and extract sentences from it. The matrix V^T indicates how well a sentence covers important words, i.e. topic words. The sentence that is best is in the first row of the matrix. The last row of the matrix contains the sentence that covers the least important words. In Gong & Liu (2001) sentences are selected from top to bottom until the compression rate of the summary is violated. The authors select a sentence per topic, however, in some cases more than one sentence may be required to cover all information belonging to that topic (Steinberger & Jezek 2004, Ozsoy et al. 2010).

Murray et al. (2005) aim to avoid this problem by selecting zero or more sentences per topic. For this they make use of the Σ matrix. For each topic they extract the corresponding singular value (the topic weight) from the Σ matrix and compute its percentage value compared to the sum of all singular values in that matrix. That percentage value is used as an indicator about how many sentences to select for each topic. After extracting this percentage value they follow a similar idea as in Gong & Liu (2001) and select from the V^T matrix the best sentences for the important topic first, then for the second important word, etc. until they reach the summary length. The authors compare the LSA approach to *tf*idf* and MMR for meeting records summary and report best results using LSA.

Another sentence selection method is proposed by Steinberger & Jezek (2004), who select sentences that cover many topics. The authors limit the number of topics by a predefined l. They report better summary evaluation results than the sentence selection method used in Gong & Liu (2001).

Ozsoy et al. (2010) propose a further sentence selection method[1] that works similarly to the one described in Steinberger & Jezek (2004). The authors argue that not every sentence for a given topic in the V^T matrix plays a core role for that topic and thus eliminate the less significant sentences from the row. They achieve this by adding the values of each row, computing the average value for cells in the row and setting all cell values which are less than the average to zero. After this step they follow the sentence selection idea of Steinberger & Jezek (2004) to generate the summary. Using this method the authors report better results than all previous three methods.

Lexical Chains

In lexical chains the text is represented using several undirected graphs also called chains with nodes containing the words in the text and connections between the nodes representing the relationship between the words. Lexical chains are motivated by lexical *text cohesion*. The notion of cohesion was introduced by Halliday & Hasan (1976) to account for how a text is bound together. Semantic relations such as word repetition, synonyms (e.g. “sick” and “ill” are synonyms) and hyponyms (“red” is hyponym of “colour”), and also collocation of words (words which tend to co-occur in the same lexical context) can be used as indicators for computing the

[1] Although the authors propose two methods we only describe the one that outperforms its peers.

lexical cohesion of a given text. Lexical chains are sequences of related words where the relation between the words is one of the relations mentioned above (Morris & Hirst 1991).

Barzilay & Elhadad (1997) apply lexical chains to text summarization. Using lexical chains the authors aim to understand the discourse structure of the input source and as a result produce more cohesive and coherent summaries. They compute lexical chains using WordNet (Miller 1993), an online thesaurus containing words with linguistic information and the relations between them (encoded in synsets which group words synonymous with each other). For a given text they use the nouns (words which appear as nouns in WordNet) to construct the lexical chains. For each noun the authors extract from WordNet its synsets and connect it with the words within these synsets. Each connection indicates whether the relation is identity, synonymy, antonymy ("long" is antonym of "short"), hyperonymy ("color" is hyperonym of "red") or holonymy ("tree" is holonym of "park"). Groups of related words are created. For each group different constellations are constructed, where each constellation contains different connections between the words. To each group the authors assign a score based on the number of connections that exist between the words and the relation type of the connections. From each text a set of lexical chains is derived. However, the authors use only the ones which score more highly than an experimentally set threshold (e.g. consider only the top five scoring lexical chains and discard the remaining chains). Using these high scoring lexical chains, Barzilay & Elhadad (1997) extract sentences from the document. To do this three different approaches are investigated: (1) extract sentences which contain the first appearance of a chain member, (2) extract sentences which contain the first appearance of a representative chain member and (3) extract sentences which match many chain members. Representative chain members are words which represent a target topic more than other words in the chain. Such members are computed based on word occurrence frequency in the chain. Words that occur more frequently than the average word occurrence frequency in the chain are considered as representative members. Note, in case the first word of the chain is a representative chain member then the second method is equal to the first one. The authors report that the best extracts were obtained using the second approach. It should also be noted that for each lexical chain only one sentence from the document is extracted.

Graph Based Methods

In graph based text summarization first textual units are mapped into a graph with nodes representing those units, and then edges are added representing the similarities between the different units. Early work on graph based text summarization is reported in Salton et al. (1997) and Mitra et al. (1997). In both studies the authors perform single document summarization where an input document is mapped into a graph. The nodes represent the paragraphs within the document. If two nodes are similar, an undirected edge is drawn between them, where the similarity between two nodes is measured by the cosine metric. The authors draw edges between nodes only if their similarity value is above a threshold.

After representing the input document as a bidirectional graph, the authors perform graph traversal to extract a summary. They use different methods to traverse the graph. The first one is the *bushy path* approach which aims to extract a summary that has a comprehensive coverage of the topic(s) presented in the document. In this approach *n* nodes from the graph with the largest number of edges or links between them are selected as a summary. The paragraphs in the summary follow the order of their appearance in the original text.

The authors argue that this approach might lead to summaries which are not coherent. The reason for this is that the extracted paragraphs may be at positions in the original text such that between them many other paragraphs exist which are not extracted. Without these paragraphs the context of the extracted paragraphs may be lost and thus may lead to less coherent summaries. To avoid this problem the authors propose a second method, the *depth-first path* method. This method starts with a node that is a bushy node. Following this node another node is selected that is most similar to the first one. Then the second node is followed, and the node with highest similarity to the second node is selected, etc. until *n* nodes from the graph are selected to form a summary.

The problem with bushy nodes is that some documents are divided into different segments, each with a different topic, so that using bushy nodes only to extract summaries may lead to exclusion of some topics from the summary because the nodes from a segment may not be well connected with others. To address this problem the authors use a *segmented bushy path* approach which selects a bushy node from each text segment and includes it into the summary.

In their final approach (*augmented segmented bushy path*) the authors make sure that they also include paragraphs from the introduction. The authors compare their automated summaries with manual extracts and report that the best automatic summaries were obtained using the *bushy path* and *augmented segmented bushy path* approaches, suggesting that introductory paragraphs are more important for the summary than other paragraphs, whose topics may be omitted without quality loss.

Mani & Bloedorn (1997) also apply graph based methods to multi-document summarization. In their approach the nodes of the graph represents words. The authors define the following links between the words: *SAME*, *ADJ*, *PHRASE*, *NAME*, *COREF* and *ALPHA* links.

SAME links are created between the same words occurring in different positions. To establish whether words are same or not, string similarity between words is computed using word stems instead of actual word tokens. *ADJ* links are drawn between adjacent words. Word sequences which build a phrase are linked through *PHRASE* links which connect the beginning word of a sequence with the ending word of that sequence. *NAME* link connects adjacent words which form a proper name. As in the *PHRASE* link, the beginning word of the proper name is connected with the ending word of that sequence. The *COREF* link is drawn between two sub-graphs, relating positions of name occurrences which are coreferential. Finally, *ALPHA* links are drawn between words that are semantically identical, as determined using WordNet and NetOWl.[2] These words connected with *ALPHA* links do not have to be adjacent.

Each word is scored using *tf*idf*. Starting from query nodes (topic words used as activation nodes) the authors traverse the graph using the spreading activation method (Chen & Ng 1995) to identify related document nodes. They jump from a query node to a document if that document node is same as the query node. Then using the links the graph is traversed from node to node, and the scores of the words (nodes) are updated. The new scores of adjacent nodes are updated based on a decaying exponential function of query node scores and the distance it took to come from the query node to the current document node. The authors assign different distance scales to the paths, so that traveling across sentence boundaries is more expensive than traveling within a sentence, but cheaper than traveling across paragraph boundaries.

For queries with multiple words each word can lead to a different graph path as the query words are taken as starting point for traversing the graph. By comparing different graph paths to each other the authors identify the common and different nodes. Using these common and different nodes a sentence selection procedure is performed where sentences from the input documents are scored based on the number of common and different nodes they contain (accumulation of the *tf*idf* scores averaged by the number of common and different nodes they contain). First, the highest scoring sentence is extracted, then the next highly scoring one until the number of sentences to extract is reached. The authors generate summaries containing only sentences that entail the common nodes from different graph paths and also summaries containing sentences that include information from the different nodes. In both cases the generation of summaries is performed with and without the spreading activation method, and the outputs are compared to each other. The authors report significantly better results when the spreading activation method is used compared to when it is not used.

[2] www.netowl.com

Another graph based multi-document summarization system is described in Erkan & Radev (2004). They use a random walk method to identify central sentences in the input documents. In their approach they use sentences as the textual units to map into graph nodes. The sentences are transformed into a vector space representation. The vector space contains words obtained from a larger corpus. Words (vector positions) that are also included in the sentence under consideration get the actual *tf*idf* values, and the remaining vector positions are zeros. Next, the authors compute cosine similarity between two node vectors. If the similarity is above a threshold, an undirected link is drawn between them. Then for each node a centrality score is computed which is equivalent to the number of links a node has (including the link that shows to itself). In summary generation the highly scoring sentences (sentences with high centrality scores) are extracted to form the summary.

However, the authors argue against the centrality method as it could lead to the problem that it favours sentences that are not the actual central sentences (sentences that cover the main topic of the input documents). This can happen, for instance, when a sentence is connected to many topic unrelated sentences within a document. To avoid this problem when computing the centrality score for a node the authors take the centrality scores of the neighbours of the node into consideration. The centrality score for a node u is then computed according to the following formula:

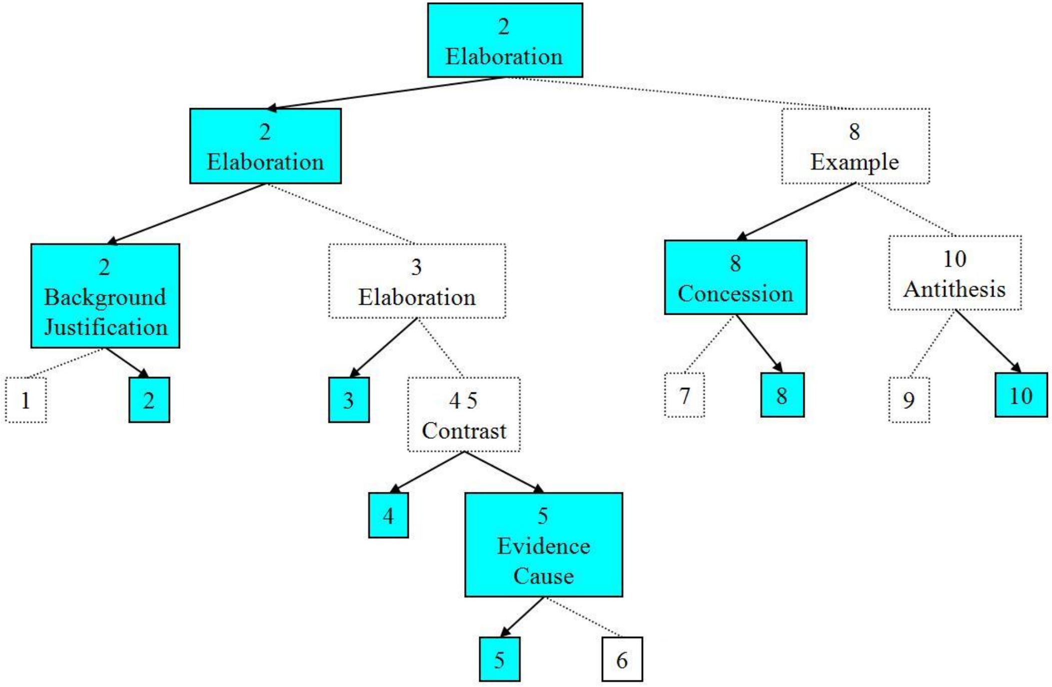
$$p(u) = \frac{d}{N} + (1-d) \sum_{v \in adj[u]} \frac{p(v)}{deg(v)} \tag{2.10}$$

where $adj[u]$ is the set of nodes adjacent to u, $deg(v)$ is the degree (number of links) of the node v, *N* is the total number of nodes in the graph, *d* is the damping factor which is typically chosen within the interval $[0.1, 0.2]$. The equation shown in Formula 2.10 is known as *LexRank* or *PageRank* which is used by Google to score documents in a search engine setting. Like the pure centrality scoring approach, the high *LexRank*ed sentences are selected to form the summary. The authors experiment with Document Understanding Conference (DUC) data (see Section 2.3) and report highly competitive results compared to the other summarization systems that were also run on the same data sets.

Rhetorical Structure Theory (RST)

The Rhetorical Structure Theory (RST) method introduced by Mann & Thompson (1988) represents input text in a tree. In RST a document is divided into different non-overlapping text spans

Figure 2.1: RST tree example. The filled leaf boxes are regarded as nucleus and the others as satellites Marcu (1997*a*).



where the text spans stand in rhetorical relation to each other. Each text span is characterized as being a nucleus or satellite. Nucleus text spans are regarded as more essential to the writer's purpose than satellite ones. Furthermore, the nucleus text spans of a rhetorical relation can occur independently from satellite text spans, but not vice versa. The following adjacent text spans (*5* and *6*), taken from Marcu (1997*a*), give an example of a nucleus and satellite:

5*: but any liquid water formed in this way would evaporate almost instantly* ***6*** *: because of the low atmospheric pressure*

A rhetorical relation "Evidence" holds between the adjacent text spans *5* and *6* (cf. Figure 2.1). The text span *5* is marked as being nucleus and is connected to the "Evidence" relation with a solid arrow. The text span *6* is marked as satellite and is connected to the relation with a dotted line.

Several authors have developed RST based text summarization methods. The most popular RST based text summarization is the one described in Marcu (1997*a*). Their work is concerned with single document summarization. For a given text, a discourse tree, such as the one shown in Figure 2.1, is constructed using rhetorical parsers (Marcu 1997*b*). After such a tree is obtained, text spans containing only nucleus are selected to form a summary. The authors use partial ordering obtained from the discourse tree to perform summary generation. Uzêda et al. (2010) provide a detailed descriptions of different selection methods used in RST based summarization.

Teufel & Moens (2002) and Teufel (2010) also use rhetorical discourse elements to perform sentence categorization in scientific papers. However, unlike the hierarchical approach of Marcu (1997*b*) the authors use flat rhetorical elements or categories to represent sentences from the input text. Using supervised machine learning methods, the connection between summarization features and manually labeled data is learnt to associate sentences with each of their categories. The most closely associated sentences are then selected from each category and included in the output summary.

2.3 Automated Summary Evaluation

Automatically generated summaries have to be evaluated in order to assess the quality of the systems used for their generation. Sparck Jones & Galliers (1996) distinguish between *intrinsic* and *extrinsic* summary evaluation methods. Intrinsic evaluation assesses the coherence and the informativeness of a summary, whereas extrinsic evaluation assesses the utility of summaries in a given application context and, for example, consists of tasks such as relevance assessment, reading comprehension etc. The next sections give an overview of both intrinsic and extrinsic approaches used in the literature.

2.3.1 Intrinsic Evaluation

In intrinsic evaluation of automatic summarization it is common to distinguish *reference* or *model* summaries from automated or *peer* summaries. Reference or model summaries are those generated manually by humans from the test documents used for the evaluation. Peer summaries are those generated automatically by the single or multi-document summarization system(s) being evaluated or by other humans, or representing other conditions (e.g. baselines). In the evaluation the peer summaries are compared to the reference summaries to assess their quality.

While there has been a wide range of work on summary evaluation in the past two decades, the most influential work has been carried out within the challenges organized by Document Understanding Conferences (DUC, http://www-nlpir.nist.gov/projects/duc/index.html) and Text Analysis Conferences (TAC, http://www.nist.gov/tac/) organized by the National Institute of

Standards and Technology (NIST) in the US. Both conferences have contributed to the dissemination of recent results, definition of tasks and evaluation setups which have focused research investigations towards new directions in text summarization. These challenges have always contained an intrinsic summarization evaluation. While the details of the tasks and the evaluation procedures have changed over time, there have always been two components to the intrinsic evaluation. One of these addresses the linguistic quality or *readability* of the peer summary; the other addresses the *informativeness* or information content of the summary, perhaps in relation to an information need or topic that may have been expressed in advance.

To assess readability the peer summary is evaluated, for example, on how coherent it is, i.e. the summary is checked to see if it contains dangling anaphora or gaps in its rhetorical structure (Mani 2001). Assessing the readability of a summary is done manually. Humans are asked to assess various aspect of the readability of a peer summary by answering questions in terms of a five point scale. The scores for the peer summaries are compared to those for the reference summaries, which are assessed in the same way as the peer summaries (Mani 2001, Dang 2005, 2006). Such manual assessment of readability is labour intensive and thus expensive to conduct and difficult to repeat. However, there is no accepted automated way of assessing the coherence of summaries that would avoid the disadvantages of manual assessment.

To assess informativeness a variety of approaches have been adopted within the document summarization community. All of them revolve around comparison of the peer summary with one or more reference summaries (using more than one reference summary helps to overcome the subjectiveness inherent in using a single reference produced by a single human summarizer). In essence these approaches are variants of two broad types. In one, the reference summaries are analyzed into semantic chunks, roughly equivalent to simple propositions and variously called “elementary discourse units”, “model units”, “summary content units” or “factoids” (Teufel & van Halteren 2004). Human judgements are made about the overlap between peer summaries and reference summaries in terms of the proportion of reference units found in the peer.

In the other type of approach, various forms of n-gram overlap between peer and reference are automatically computed and the peer given a score that reflects its recall of reference n-grams. The most popular n-gram overlap approach is ROUGE (Recall-Oriented Understudy for Gisting Evaluation) (Lin 2004). ROUGE compares automatically generated summaries against several human-created reference summaries. In this way it estimates the coverage of appropriate concepts in an automatically generated summary. ROUGE-1 to ROUGE-4 give recall scores for uni-gram to four-gram overlap between the automatically generated summaries and the reference summaries. Sequences of overlapping words that do not immediately follow each other are captured by ROUGE-L. In ROUGE-L gaps in word sequences are ignored so that, for instance, *A B C D G* and *A E B F C K D* are counted as having the common sequence *A B C D*. ROUGE-W allows the longest common sub-sequences to be controlled/weighted. ROUGE-SU4 allows bi-grams to consist of non-contiguous words, with a maximum of four words between two words in the bi-grams.

ROUGE was used by the Document Understanding Conferences (DUC) starting in 2004 to assess the quality of single and multi-document summarization systems and is now used by the Text Analysis Conference (TAC), also run by NIST. However, although ROUGE is the *de facto* evaluation system for automatically generated summaries, it has been criticized because it only performs string match between the summaries and does not take the meaning expressed in single words or sequences of words into consideration. But it has been shown that ROUGE evaluation results correlate with the results obtained from human judges (see below for further discussion).

The other broad class of approaches to assessing summary informativeness rely, as already indicated, on identifying "summary content units" in the reference summaries and determining the extent to which these are present in the peer. This type of evaluation was carried out in DUC for the first few years against a single reference summary. As the inadequacies of comparing against a single reference summary became apparent, the method was elaborated by a number of groups, the most popular being the pyramid approach of Nenkova & Passonneau (2004), Nenkova et al. (2007). From a set of model summaries the authors manually identify similar sentences. From these similar sentences summary content units (SCUs) are generated and ranked in a pyramid model. The pyramid model has n levels, where n is the number of model summaries. The levels are labeled in ascending order from 1 to n. SCUs are ranked in the pyramid according to their occurrence in the model summaries. For instance, if a SCU occurs in 3 of the 4 model summaries then this SCU will be placed in the 3rd level of the pyramid. The peer summaries are then manually evaluated against the SCUs. A summary is regarded as good if it contains a large number of the higher-level SCUs. Summaries containing more SCUs from the lower levels than from the higher level are considered poor summaries as they are less informative.

Measuring informativeness based on the overlap in content between peer and multiple reference summaries using human judgements about meaning similarity between content units in peer and reference summaries is a convincing way to evaluate peer summaries. However, such evaluations require a lot of manual work and annotation and thus are expensive to conduct in terms of time and money. They can feasibly only be carried out by a well-funded agency, such as NIST, and certainly cannot be done iteratively by developers seeking to refine a system in a tight development loop. Fortunately, various researchers have shown that there is significant correlation between ROUGE scores and approaches based on human comparison of semantic content units (indeed this was necessary for ROUGE to win acceptance). Louis & Nenkova (2008) report that ROUGE correlates highly (above 90%) with pyramid scores indicating that ROUGE is a low cost choice for obtaining similar results as manual evaluations. In addition, it has been reported that ROUGE correlates highly with human judgments (Lin 2004).

2.3.2 Extrinsic Evaluation

In extrinsic evaluation the idea is to assess a summary based on a task and measure how much help the summary provides for a human performing this particular task. To date evaluation of automatically generated summaries has been performed using information retrieval (Hand 1997, Tombros & Sanderson 1998, Saggion et al. 2002), report generation or synthesis tasks (Amigo et al. 2004, McKeown et al. 2005) and categorization and question answering tasks (Mani et al. 1999).

As mentioned in Chapter 1 we use as an application scenario the generation of captions for images pertaining to geo-located entities. Thus one of the tasks our summarization system can be used for is to index images of geo-located entities to ensure their efficient retrieval. An extrinsic evaluation could therefore involve an image retrieval task. In this evaluation task, currently used image indexing techniques (e.g. key words, short descriptions) could be compared against full summaries about location entities shown in an image. The evaluation of the relevance of the retrieved images to a given query (e.g. a place name) could be conducted manually, where a pooling approach commonly used in TREC[3] and ImageCLEF[4] could be adopted. In the pooling approach images retrieved by a query are shown to the assessor together. In this way the assessor does not know what indexing type – key words, short description or summary – caused the positive retrieval. He judges every image as relevant to the query or not regardless of the indexing type. Using this approach the summaries as one indexing type can be compared to other indexing types and conclusions about the usefulness of summaries for image indexing can be drawn.

2.4 Summary

In this chapter we described automatic summarization paradigms and provided an overview over techniques applied in previous work on single and multi-document summarization. The techniques we described belong to two groups: non-transformational techniques that do not require

[3] http://trec.nist.gov/

[4] http://www.imageclef.org/

changes in the input document structure and transformational ones in which document structure is transformed prior to summarization. When reviewing non-transformational techniques we focused on addressing the two main challenges in text summarization: identifying summary-worthy sentences or sentence scoring and generating an informative and readable summary or summary composition. For sentence scoring we discussed various features and feature combination techniques. For summary composition, we introduced several previous approaches to informativeness maximization, redundancy reduction and achieving summary coherence. Furthermore techniques that use document structure transformation for performing the summary composition have been described. Finally, different approaches to evaluation of automatically created summaries have been discussed.

CHAPTER 3

Overview of Methodology and Resources

As outlined in Chapter 1 the main goal of this work is to integrate the idea of entity type modeling into a multi-document summarization system and to apply this system to automatic generation of captions for geo-located entities. To achieve this goal we proceed in four steps: 1) collecting information related to geo-located entities, i.e. attributes associated with geo-located entity types; 2) analysing existing text resources to establish whether, and if so how, these attributes are described in existing texts; 3) exploring methods of geo-located entity type model representation and their exploitation in summarization; and 4) evaluation of our summarization techniques.

In this chapter we outline the methodology we use to implement each of these steps. In addition, we describe existing resources we use for generating and evaluating summaries for geo-located entities.

3.1 Methodology Steps

3.1.1 Step 1: How Humans describe Geo-located Entity Types

The first step in our procedure is to analyse whether there exist stable sets of attributes that different humans use to describe the same entity type, i.e. that different people tend to characterize instances of a particular entity type using the same set of attributes. This step addresses our first research hypothesis (cf. Section 1.3), which is that entity types (e.g. a church, a bridge, a mountain) are characterized by sets of attributes, some of which are entity type specific and others of which are shared between types. We formulated this hypothesis based on general observations, and the findings of an early experiment we conducted (Aker & Gaizauskas 2008), which suggested that there are stable sets of attributes which humans use to characterize specific geo-located entity types.

The main aim of this experiment was to assess whether generic or query-based summaries are more suitable as captions for location images. We used 24 images pertaining to geo-located entities around the world and performed generic and query-based summarization using an existing summarization system, SUMMA (Saggion 2008). We showed that query-based summaries tend to be better than generic summaries. However, the resulting summaries were still far from ideal. Therefore, we examined the information humans provide when asked to describe a particular geo-located entity of a specific type. We observed high levels of agreement between humans about which information to include, leading us to hypothesize that humans have an idea of what is salient regarding a certain entity type and that they use this in providing a description of that type.

In this work we report a further experiment which aims (1) to test the hypothesis that there are sets of attributes which define geo-located entity types and (2) to establish which sets of attributes are specific to particular entity types and which are shared between different types. The experiment was conducted on Amazon's Mechanical Turk.[1] We showed the participants different images pertaining to entity types around the world and asked them to formulate questions for which they would like to know the answers when seeing the image. In this way we collected 7644 questions from 184 participants. Confirming our hypothesis, the results indicate that humans do use common sets of attributes to describe entities of the same type and we are also able to identify and describe the entity-specific and shared attributes. We describe this experiment in Chapter 4.

3.1.2 Step 2: Building Corpora for Geo-located Entity Types

Having identified which sets of attributes people find relevant when describing geo-located entity types, the question arises as to whether these attributes are addressed in existing online text collections (cf. our second research hypothesis in Section 1.3) . If so, then these collections can potentially be used to extract entity type models of geo-located entities, i.e. models which capture the information about the relevant geo-located entity type attributes.

We assume that Wikipedia may be a suitable web resource for deriving geo-located entity type models. Wikipedia is a multi-lingual encyclopedia project which is free and accessible through

[1] https://www.mturk.com/mturk/welcome

the web[2]. It contains 12 million articles in 265 different languages. The richest article set is available in English with more than 2.7 Million articles (numbers are based on English Wikipedia dump from 24/07/2008).

Apart from being used as an encyclopedia, Wikipedia has also drawn the attention of researchers as a rich multi-lingual text resource. Wikipedia articles are used, for instance, for deriving a large scale taxonomy (Ponzetto & Strube 2007), named entity and term recognition and translation in question answering (Bouma et al. 2007), named entity disambiguation, translation and transliteration (Wentland et al. 2008), domain specific query translation in multilingual information access (Jones et al. n.d.), to name just a few applications. Amongst other information, high quality descriptions of geo-located entities can be found in Wikipedia, which makes it a potentially rich corpus resource for deriving entity type models.

Wikipedia would be suitable for deriving entity type models if the information contained in its geo-located entity related articles matches the information identified as relevant by human participants in our experiment from Step 1. To establish whether this is the case we automatically compare sets of attributes identified in the experiment to those manually extracted from Wikipedia articles about the same entity type. We automatically categorized each English Wikipedia article by entity type, e.g. *church*, *bridge*, etc. In total we found 107 different types covering both urban and rural geo-located entity types. The number of documents for each entity type varies between 50 and 30,000 articles. We then manually analyzed a subset of this set of entity types (the articles within the entity types) and automatically compared them to the types of information humans indicated as relevant in our online user survey. The results of this analysis indicate a high match (cf. Chapter 5), confirming that Wikipedia is suitable for deriving geo-located entity type models. This also confirms our hypothesis that relevant attributes which define geo-located entity types can be derived from existing online text resources.

3.1.3 Step 3: Building Geo-located Entity Type Models for Summarization

Our investigations from previous steps indicate that humans appear to have an entity type model of what is salient regarding entities of the same type. The question arises as to whether we can represent or approximate such an entity type model in a way that allows us to improve sentence selection for our image description generation task. While there are many ways this could be done (e.g. generating rules and/or templates as in Paice & Jones (1993), McKeown & Radev (1995), Radev & McKeown (1998)), based on considerations in Step 2, one can view corpora of

[2] http://en.wikipedia.org/wiki/Wikipedia

descriptions of entities of a given type (entity type corpora) as containing an implicit model of that type and derive different models (e.g. signature words, language models and dependency patterns) to bias the sentence selection of a summarizer. We describe each modeling approach and its impact on the quality of the entity-related descriptions in Chapter 7.

Furthermore, we hypothesize that not only sentence selection, but also sentence ordering in summary composition can benefit from entity type models. In Chapter 8 we therefore experiment with incorporating entity type modeling into the summary composition component of the summarizer.

3.1.4 Step 4: Evaluation

Automated descriptions or summaries can be evaluated using intrinsic and extrinsic approaches (cf. Section 2.3). In the intrinsic approach the evaluation examines the automated summaries directly and assesses their quality independently from any task setting where the summaries might be used. The extrinsic approach, on the other hand, assesses the quality of a summary in a task setting.

In this thesis we only report an intrinsic evaluation of automated summaries. However, in Aker, Fan, Sanderson & Gaizauskas (2012) we have reported an extrinsic evaluation by assessing the usefulness of summaries in image retrieval task. We do not report this work in this thesis as it was part of the broader TRIPOD project where different components developed by different people were used in the experimental setting. In that evaluation we collected 6,385 images from Flickr and generated several different summary types for each image with which to index it. The summaries were generated from a set of 10 top-ranked documents retrieved from the Web using the name of the geo-located entity shown in the image as a query.[3] We used textual features for generating the summaries (cf. Section 2.2.1), but also entity type models (n-gram language models) derived from existing geo-located entity descriptions. We also used the titles and tags provided with the images for indexing purposes. The results show that entity type model biased summaries in combination with existing image textual information (title and tags) lead to a significant improvement in the retrieval effectiveness task when compared to all other settings.

For intrinsic evaluation we evaluate the output of our summarizer using automatic ROUGE (Lin 2004) comparison as well as human readability assessment. Automatic evaluation using ROUGE requires reference or model summaries, which currently do not exist for the task of

[3] Based on the geo-graphical information we derived the entity name using the techniques described by Fan et al. (2010)

generating geo-located entity-related descriptions. Thus, to successfully evaluate our automated summaries we manually collected model summaries about geo-located entities. In total we have model summaries for 307 different entities/images worldwide. Each entity has up to four model summaries. The summaries were collected by 11 human subjects. We assessed the quality of the model summaries and show that they have comparable quality scores to the ones provided by DUC and TAC. We describe the model summaries in Chapter 6.

3.2 Resources

3.2.1 Summarizer

The image captions are generated using *the-MDS* (the-multi-document summarizer), an extractive, multi-document, query-based summarization system implemented in Java. The input to the summarizer is the query, i.e. the name of the geo-located entity, an entity type and *n* documents obtained from the web (see Section 3.2.2).

The summarizer uses a three step approach to create image descriptions. It first applies shallow text analysis to the given input documents. Then it uses a set of features to identify salient sentences. Finally, it performs sentence selection on the salient sentences to create the final summary. The latter two tasks are language independent and can be performed for any UTF-8 encoded language. This means that *the-MDS* needs only a shallow text analyzer for any specific language and is then able to do the summarization. In following subsections the three steps are described in more detail.

Shallow Text Analysis

The-MDS first applies shallow text analysis to the given documents. This includes sentence detection, tokenization, lemmatization, named-entity recognition and POS-tagging. For these tasks OpenNLP[4] tools are used. For performing each of the preprocessing steps OpenNLP uses maximum entropy models which are pre-trained using texts from the Wall Street Journal and the Brown Corpus.

Feature Extraction

After text analysis, *the-MDS* extracts for each sentence the following features:

[4] http://opennlp.sourceforge.net/

- ***qSim***: Sentence similarity to the query, computed as the cosine similarity over the vector representation of the sentence and the query. Each vector position contains *tf*idf* scores for the words. The *idf* table is generated on the fly from the *n* related documents input to the summarizer.
- ***cenSim***: Sentence similarity to the centroid, computed as cosine similarity over the vector representation of the sentence and the centroid. The centroid is composed as described in Section 2.2.1. However, as in Radev et al. (2004) we keep in each vector only the 100 words in the document containing the highest *tf*idf* score.
- ***senPos***: Position of the sentence within its document. The first sentence in the document gets the score 1 and the last one gets $\frac{1}{k}$ where *k* is the number of sentences in the document.
- ***isStarter***: A sentence gets a binary score if it starts with the query (geo-located entity name) term (e.g. *Westminster Abbey*, *The Westminster Abbey*, *The Westminster* or *The Abbey*) or with the entity type, e.g. *The church.* We also allow gaps (up to four words) between *the* and the query to capture cases such as *The most magnificent abbey*, etc.
- ***entityTypeModel***: A sentence is scored according to a entity type model derived from Wikipedia entity type corpora. Details about the different representations for entity type models and their application to scoring sentences are given in Chapter 7.

The query similarity (*qSim*), the centroid similarity (*cenSim*) and the sentence position (*senPos*) features are adapted from previous work (cf. Chapter 2) and the features *isStarter* and the *entityTypeModel* are novel. Further features from related work could be adopted and investigated in the context of improving the image captioning task. However, since the main research goal of this work is to evaluate the impact of entity type modeling on multi-document summarization, we experiment with the features presented above which are considered standard for automatic summarization, so that we are able to compare the performance of entity type model-based summarizer to a fairly standard system.

Sentence Scoring and Selection

The features described in the previous section are combined in a weighted linear combination to rank the sentences based on formula 3.1.

$$S_{score} = \sum_{i=1}^{n} feature_i * weight_i \tag{3.1}$$

The values for the weights are trained using machine learning techniques. We train the feature weights in two different ways. The first approach uses training instances where their outcome is known, i.e. good and bad sentences. Using these instances we train the feature weights to distinguish between good and bad sentences. However, if good sentences are combined together into a summary, there is no guarantee that the resulting summary will be optimal too. Therefore, it is necessary to predict the best actual outcome of the summarizer, i.e. the output summary, instead of its components (single sentences). To address this problem, we use a second approach where the feature weights are trained based on the actual outputs of the summarizer. In this way the feature weights are trained to distinguish between good and bad summaries.

After the scoring process, *the-MDS* composes the output summary by selecting sentences to include in it. We use two different approaches for sentence selection. The first method is the frequently used selection method (cf. Section 2.2.1) where feature weights trained on sentences are used to score sentences. After the scoring process the sentences are selected starting from the highest scoring sentence until the compression rate is reached. However, this greedy approach does not necessarily lead to the best summary. The second approach formulates the summary generation as a search problem and aims to find the best summary. It does this by selecting a subset of sentences from the input documents which when concatenated form the final summary that is also the best among all other possible summaries. Experimental results with the greedy sentence selection method are reported in Chapter 7 and those using the second approach in Chapter 8.

3.2.2 Web Documents

The entity descriptions are generated from a set of documents which have been retrieved from the web using the entity name as a query.[5] For retrieving the documents we use the Yahoo! search engine. In the retrieval process we ensure that the search results are healthy hyperlinks, i.e. that the content of the hyperlink is accessible. From the results list we filter out any site related to *VirtualTourist*, as *VirtualTourist* web-documents are used to generate our model summaries. Each remaining search result was crawled to obtain its content using a web-crawler.

The web-crawler downloads only the content of the document residing under the hyperlink, which was previously found as a search result, and does not follow any other hyperlinks within

[5] To avoid ambiguity in entity names we also used the city and the country name when querying the web.

the document. The content obtained by the web-crawler encapsulates an HTML structured document. We further process this using an HTML parser[6] to select the main content of the page, i.e. parts which do not contain advertisements, navigation hyperlinks, copyright and privacy notices (Yi et al. 2003).

Our HTML parser constructs a parsing tree from the document following the Document Object Model[7] using a technique similar to that described in Sahuguet & Azavant (2001). Within this parse tree the HTML parser checks only the BODY tag of the document. It ignores the SCRIPT, TABLE and the FORM tags within the BODY tag as these are unlikely to contain relevant text to use as entity descriptions. In addition, it ignores parts of the BODY that contain enumeration of information, such as menu items, copyright information, privacy notices and navigation hyperlinks. Furthermore, short texts are ignored as well, as they are likely to contain advertisements. A text is considered short if it has less than 5 words. This number was selected after a couple of experiments. The text identified by the HTML parser as pure is then prepared for the summarization process. The resulting data is then passed on to a summarizer to generate the image descriptions. Following this procedure we obtain 10 different web-documents for each entity which are then given as input to the summarizer.

3.2.3 Baseline Summaries

Within the intrinsic evaluation setup (cf. Section 3.1.4) we use two different baseline system summaries to compare with our summarizer:

First Document Summaries

Firstly, we use the geo-located entity names to automatically query related documents from the web using the Yahoo! Search engine. For each toponym we take the top-ranked non-Wikipedia document retrieved in the Yahoo! search results and generate a baseline summary by selecting sentences from the beginning until the summary reaches a length of 200 words.

Wikipedia Summaries

As a second baseline we use the Wikipedia article for a given geo-located entity from which we again select sentences from the beginning until the summary length of 200 words limit is

[6] http://htmlparser.sourceforge.net/

[7] http://www.w3.org/DOM/

reached. For each geo-located entity the corresponding Wikipedia article was manually identified from the list of documents retrieved by the Yahoo! Search engine. By doing this we ensured that we took the correct Wikipedia article.

By using both the first document and Wikipedia baselines, we simulate the scenario in which image descriptions are generated by a simple web search, without needing the summarizer. In other words, our system needs to significantly outperform these baselines, to justify using multi-document summarization for image captioning.

We consider the top-ranked non-Wikipedia document to be a weaker baseline than a Wikipedia article, which we take to be a strong baseline against which to compare the automated summaries. Wikipedia articles focus only on the topic they were written about, whereas an arbitrary non-Wikipedia web-document may contain other unrelated information.

3.3 Summary

In this chapter we outlined the methodology we employ to address our research questions. We divided our research procedure into four steps, in each of which we test one or more of the hypotheses underlying this work: 1) analysing attributes humans associated with geo-located entity types; 2) analysing existing text resources for deriving geo-located entity type models; 3) exploring methods of geo-located entity type model representation and their exploitation in summarization; and 4) evaluating our summarization techniques. In addition, the chapter describes three resources: the summarizer, the web documents to be summarized and the baseline summaries. Experiments and findings related to each of these steps are the subject of the following chapters.

CHAPTER 4

How Humans Describe Geo-located Entity Types[1]

In every day life we see and categorize things in our built and natural environment. For instance, if we look at different churches, we see that each of them has a different style, different look, different size, and that some of them are newer than others, etc., but still we are able to categorize them all as churches. For categorization purposes, visual attributes such as the style and the size of a church might be enough; however, to report about each church separately we need more attributes whose values make each church distinguishable from the others. We might ask: is there a general set of attributes that people use to describe any geo-located entity? Or, given that humans categorize entities into types (e.g. church, museum, lake, etc.), are the sets of attributes specific to single entity types, or perhaps shared between entity types of similar function (e.g. church and temple)?

In this chapter we aim to address these questions. Specifically, we ask: For what set of attributes related to a geo-located entity would people like to know the values when seeing this geo-located entity? And: Is this set of attributes specific to a particular entity type (e.g. church) or shared between different types? In this way we test our first research hypothesis (cf. Section 1.3), which states that entity types are characterized by sets of attributes, some of which are type specific and others of which are shared between different entity types.

We aim to answer these questions using an online survey conducted on Amazon's Mechanical Turk. In this chapter we first address related work. Then we describe Mechanical Turk, present the experimental setting of our survey and outline the preprocessing of the data (cf. Section 4.2). Finally, we present our analysis and report and discuss the results in Section 4.3.

[1] Some of the results presented in this chapter are published in (Aker & Gaizauskas 2011, Aker et al. 2013).

4.1 Related work

To the best of our knowledge, similar studies of what information types human associate with geo-located entities have not been reported previously.

Earlier work has studied image related captions or descriptions in order to understand how people describe images by looking at query logs or existing image descriptions (Armitage & Enser 1997, Balasubramanian et al. 2004, Joergensen 1996, Jörgensen 1998, Greisdorf & O Connor 2002, Choi & Rasmussen 2002, Hollink et al. 2004). These studies are aimed at understanding how people describe images in general and are not specific to locations, so they offer only limited information of the type we need for our research. For geo-located entities we are concerned with entity types such as *mountain, lake, etc.* as investigated by Smith & Mark (2001). These authors analyse how non-experts conceptualize geo-located entities by asking them to suggest entity types after seeing a phrase related to a geo-located entity, such as *natural Earth formation*. With this study the authors demonstrate that entity types of the natural landscape form a part of human cognitive inventory about geo-located terms. However, they do not investigate what characteristics or attributes humans associate with different entity types. We aim to investigate how humans characterize geo-located entity types in terms of sets of attributes, which may be specific to single entity types, or shared between entity types of similar function (e.g. church and temple).

We think that such information types human associate with geo-located entities could be used in the field of pedestrian and car navigation systems (Janarthanam et al. 2012, Dräger & Koller 2012). The systems could use the geo-located information types to describe entities shown in a route.

4.2 Experimental Setting

4.2.1 Mechanical Turk

Mechanical Turk (MTurk)[2] is a crowdsourcing service run by *Amazon*. It allows users (also called requesters) to upload tasks and obtain results within a very short time. The tasks are performed by MTurk workers. Each worker has the ability to view existing tasks (also called HITs for “human intelligence task”) and complete them for a fee offered by the requester.

[2] https://www.mturk.com/mturk/welcome

The general advantage of MTurk, the ability to collect results in a time and cost efficient manner, makes it a suitable platform for conducting this experiment. MTurk has been widely used for language processing and information retrieval tasks (Snow et al. 2008, Kittur et al. 2008, Dakka & Ipeirotis 2008, Kaisser & Lowe 2008, Yang et al. 2009, El-Haj et al. 2010). Several studies have shown that the quality of results produced by MTurk workers is comparable to that of traditionally employed experimental participants (Su et al. 2007, Snow et al. 2008, Alonso & Mizzaro 2009). However, there are also some limitations of MTurk, which are relevant for our study. Our experimental design (see next section) allows us to address some of these issues.

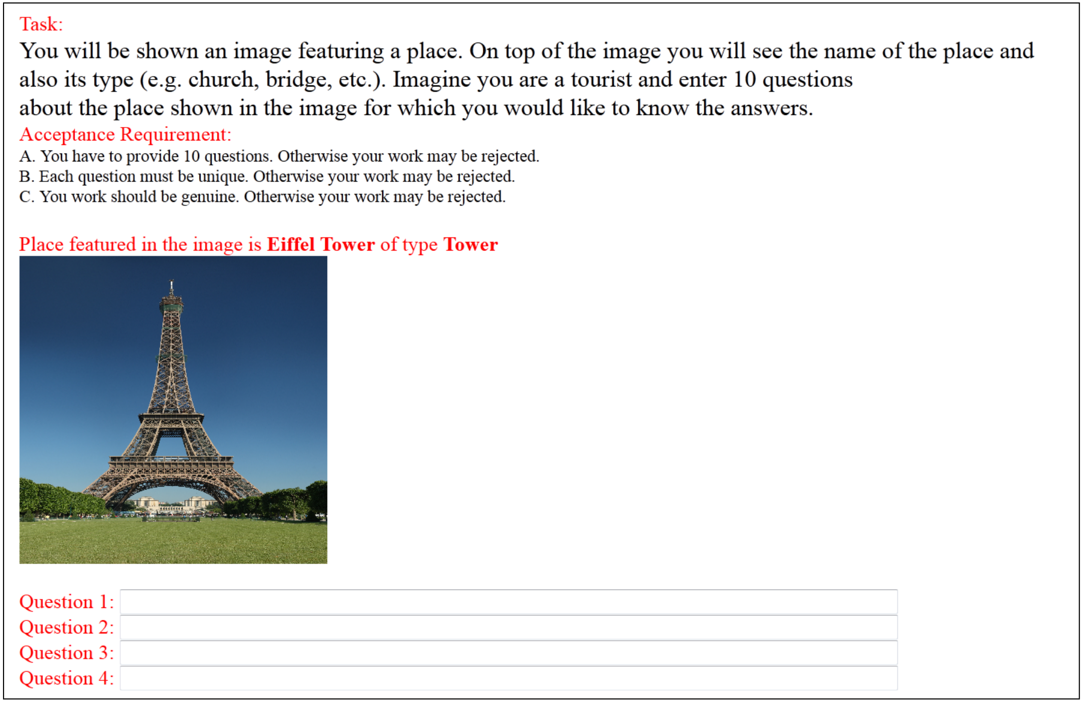
4.2.2 Experimental Design

Our experimental design is similar to the one described in Filatova et al. (2006). These authors are interested in knowing what information people expect to know when they read an article about an event. The authors use four different event types (*terrorist attack, earthquake, presidential election* and *airplane crash*). They asked ten humans to provide questions for which they would like to get the answers when they read an article about each of those events. We set up our experiment in a similar way and asked humans to provide questions for which they would like to know the answers when they see a geo-located entity. We obtain the questions via MTurk workers.

In the experiment we showed the workers an image pertaining to a particular geo-located entity (e.g. the *Eiffel Tower* in Paris as shown in Figure 4.1). We also presented the worker with the name of the entity (*Eiffel Tower*) and its type (*tower*). The workers were asked to take the role of a tourist and provide ten questions for which they would like to know the answers when they see the entity shown in the image. Note that this is different from the experimental setting reported in Filatova et al. (2006) where the human candidates are general newspaper readers and are not asked to take an explicit role when providing the questions.

The experimental design shown in Figure 4.1 allows us to address some of the quality issues associated with conducting experiments on MTurk. Related work has reported problems with spammers and unethical workers, who produce incomplete or absurd output (Feng et al. 2009, Mason & Watts 2009, Kazai, G. 2011). In our experiment, for example, text fields were provided for writing down each question based on investigations in Aker, El-Haj, Albakour & Kruschwitz (2012) who report that text fields are a good design selection in order to detect unwilling or less careful workers. In our setting text fields allow us to control the quality of the questions and reject all absurd input, like strings of arbitrary characters, etc. The workers were also required

Figure 4.1: Design of the online MTurk experiment. Because of space limitations the figure shows only text fields for four questions. However, in the experimental design there were 10 such text fields. The image is taken from Wikipedia.



Task:
You will be shown an image featuring a place. On top of the image you will see the name of the place and also its type (e.g. church, bridge, etc.). Imagine you are a tourist and enter 10 questions about the place shown in the image for which you would like to know the answers.

Acceptance Requirement:
A. You have to provide 10 questions. Otherwise your work may be rejected.
B. Each question must be unique. Otherwise your work may be rejected.
C. You work should be genuine. Otherwise your work may be rejected.

Place featured in the image is **Eiffel Tower** of type **Tower**

Question 1:
Question 2:
Question 3:
Question 4:

to provide a complete set of 10 questions. This requirement was not always fulfilled. However, we also accepted lists containing less than 10 questions, provided the questions were of good quality.

We showed images picturing geo-located entities of 40 different types randomly chosen from our entire set of 107 entity types. From these 40 entity types, 25 were urban and 15 rural (see Table 4.1). For each entity type five different entities were shown. For example, for the entity type *tower* images of *Eiffel Tower, Flag Tower of Hanoi, BT Tower, Munttoren and Bettisons Folly* were shown. These entities (towers) were manually selected from Wikipedia. Each image was shown to five different workers.

We ran the experiment for four weeks. In total we collected 7644 questions for 187 different geo-located entities. The questions came from 184 different workers. The expected number of questions was 10.000 (40 types × 5 objects × 5 workers × 10 questions). However, there are some entities for which we only have questions from two or three workers. Most entities have 40 questions. The number of questions for the urban types is 4815 and for the rural ones 2829.

Table 4.1: Geo-located entity types used in the experiment.

Urban types	Rural types
abbey, aquarium, avenue, basilica, boulevard, building, cathedral, cemetery, church, gallery, house, monument, museum, opera house, palace, parliament, prison, railway, railway station, residence, square, stadium, temple, tower, university, zoo	beach, canal, cave, garden, glacier, island, lake, mountain, park, peak, river, ski resort, village, volcano

Table 4.2: Top ten attributes with related questions.

visiting	where i can buy the ticket?, is this tower available to be visited the whole year?, when is the best time to visit?, how to get there?
location	where is garwood glacier?, where exactly is edmonton?, where it's located?
foundationyear	when was it build?, which year was this zoo opened?, when it was established?
surrounding	what are the landmarks found nearby seima palace?, what are the nearby places to visit?, what are some nearby attractioons?
features	are there any waterfalls in the park?, what does the zoo house?
history	what is the history of it?, any history related to george mason university?, what role did the palace play in the history of france?
size	how big is the complex?, how big is the temple?, how big is it, meaning what are the dimensions?
design	what is the architectural structure style?, what style of architecture is this house built in?, is it constructed by ancient technology?
mostattraction	what is the speciality of this abbey?, what is the best in this island?, what are the extraordinary facts about this place?
naming	was it named after a person?, how did the abbey get its name?, how the name for this boulevard came?

4.2.3 *Question Preprocessing*

We manually analyzed all questions in order to assess quality. Approximately 2% of the questions were empty because not all the workers wrote 10 questions for each geo-located entity. Moreover, some questions were only related to the image itself rather than to the entity shown in the image (e.g. *"when the picture is taken?"*, *"how many flowers you found in the image?"*, *"is there a bus in the picture?"*). In addition to these, some questions present non-resolved references so it is impossible to know what entity they refer to (e.g. *"what language do **they** speak?"*). Finally, there are questions which bear no relation at all to the entity in the image

Table 4.3: Attributes used to categorize the questions: attribute name (questions count in each attribute, percentage of the total number of questions)

Top Ten	Below Top Ten
visiting(1068, 17.31), location(861, 13.96), foundationyear(465, 7.54), surrounding (426, 6.91), features(357, 5.79), history(239, 3.87), size(214, 3.47), design(211, 3.42), mostattraction(165, 2.67), naming(153, 2.48)	purpose(136, 2.20), istouristattraction(132, 2.14), height(132, 2.14), visitors(126, 2.04), founder(113, 1.83), owner(109, 1.77), events(104, 1.69), type(102, 1.65), status(86, 1.39), habitants(74, 1.20), designer(69, 1.12), constructioninfo(63, 1.02), temperature(58, 0.94), length(56, 0.91), eruptioninfo(42, 0.68), comparison(42, 0.68), capacity(38, 0.62), depth(36, 0.58), maintainance(35, 0.57), arts(27, 0.44), studentsinfo(26, 0.42), subjectsofferred(25, 0.41), gravesinfo(23, 0.37), preacher(22, 0.36), firstdiscovery(20, 0.32), parkinginfo(20, 0.32), religioninfo(19, 0.31), destination(16, 0.26), origin(15, 0.24), workers(12, 0.19), width(12, 0.19), travellers(11, 0.18), travelcost(9, 0.15), waterinfo(9, 0.15), travelling(9, 0.15), firstclimber(9, 0.15), tributaries(8, 0.13), prayertime(6, 0.10), personinmonument(6, 0.10), snowinformation(6, 0.10), firstfounder(5, 0.08)

(e.g. *"how is the manager?"*). These questions do not address the task, which is to ask questions about the *entity* shown in the image, not about the image itself or related information. Therefore, we categorized all these questions as noise. They make up 19% (1479 out of 7644) of the entire question set.

We categorized the remaining 81% of the questions (6169 out of 7644) by the *attribute* the worker was seeking the value for with his/her question. An attribute is an abstract grouping of similar questions. We regard two or more questions as similar if their answers refer to the same information type. For instance, we regard the questions *"where is garwood glacier?* and *"where exactly is edmonton?"* as similar because both aim for answers related to the information type *location*. We name the attribute according to the information type it refers to (e.g. *location*). Table 4.2 shows question examples for the *top ten* attributes (the 10 attributes which have the most questions).

There are in total 146 attributes. However, 95 of them contain less than five questions, so we ignore these attributes in further analysis. We analysed the remaining set of 51 attributes (see Table 4.3), each of which has at least five related questions.[3]

[3] The total number of questions discarded with these 95 categories is 150 which makes around 2% of the questions used for categorization.

4.3 Analysis

4.3.1 Is there a set of attributes that people generally associate with geo-located entity types?

In Table 4.3 the attributes along with the number of questions for each attribute from all entity types are given. As the table shows some attributes in the left column (*top ten*) are very frequently addressed by the participants. These attributes cover the majority of the questions. More than 65% of the questions can be categorized by these ten attributes. This means that people do share ideas as to what types of information are required about a geo-located entity, and the set of *top ten* attributes captures these information types.

Table 4.4 below shows for each entity type the number of questions that can be categorized by the top ten attributes/below top ten attributes (second column) and the ten most frequent attributes (third column). The numbers within the brackets attached to each attribute indicate how many questions were categorized by that attribute. From this table we can see that the *top ten* attributes from Table 4.3 are present for most of the entity types indicating that the same type of information is relevant for several entity types. We can also see that for the majority of the entity types these ten attributes cover more than half of the questions asked about these entity types. However, although the *top ten* attributes from Table 4.3 occur for many entity types and cover most of the questions, their *popularity* is not the same in all the entity types with which they are associated.

type	questions in top ten categories/below top ten categories	top ten attributes for the entity types in first column
abbey	95/39	foundationyear(21), location(20), visiting(16), history(14), design(10), founder(7), height(5), mostattraction(4), designer(4), size(4)
aquarium	139/25	visiting(63), features(30), location(19), foundationyear(10), size(7), visitors(6), events(5), surrounding(5), parkinginfo(3), mostattraction(3)
avenue	84/34	location(28), surrounding(16), visiting(11), foundationyear(8), features(8), type(6), parkinginfo(5), visitors(5), mostattraction(5), naming(3)
basilica	103/56	visiting(25), location(23), foundationyear(21), surrounding(16), founder(7), visitors(7), history(7), purpose(6), constructioninfo(5), events(4)
beach	129/39	visiting(44), location(24), surrounding(23), features(18), visitors(12), mostattraction(7), naming(4), depth(4), length(4),foundationyear(4)

boulevard	100/37	surrounding(23), location(17), visiting(17), features(13), naming(13), foundationyear(7), length(6), visitors(5), mostattraction(4), purpose(4)
building	104/60	location(22), design(22), visiting(18), surrounding(10), foundationyear(10), height(9), constructioninfo(9), owner(9), type(8), features(8)
canal	95/81	visiting(23), location(21), surrounding(18), purpose(16), length(12), foundationyear(12), depth(9), istouristattraction(7), naming(5), history(5)
cathedral	113/60	location(30), visiting(23), foundationyear(16), history(12), founder(9), size(8), design(8), preacher(6), height(6), events(6)
cave	138/48	visiting(46), location(25), surrounding(15), features(15), firstdiscovery(13), history(12), naming(7), mostattraction(6), depth(6), design(5)
cemetery	110/57	location(30), visiting(29), gravesinfo(23), foundationyear(18), istouristattraction(11), surrounding(8), size(7), naming(5), mostattraction(4), history(4)
church	110/74	foundationyear(26), location(22), visiting(18), preacher(12), events(9), history(9), design(9), features(7), status(7), type(6)
gallery	116/61	visiting(43), location(23), arts(21), foundationyear(16), surrounding(10), design(8), events(5), type(5), mostattraction(5), purpose(5)
garden	108/37	visiting(34), features(23), location(20), foundationyear(11), size(9), constructioninfo(6), surrounding(6), owner(5), founder(5), purpose(3)
glacier	102/48	location(24), visiting(24), status(13), foundationyear(11), size(10), history(9), surrounding(8), naming(8), height(7), mostattraction(4)
house	104/49	design(27), location(22), foundationyear(16), owner(13), visiting(11), history(10), istouristattraction(9), founder(9), habitants(8), size(7)
island	88/46	visiting(21), location(19), habitants(19), features(15), size(13), surrounding(11), temperature(6), mostattraction(5), istouristattraction(5), history(3)
lake	152/37	visiting(61), surrounding(32), location(29), size(12), depth(10), features(7), mostattraction(6), purpose(6), visitors(3), status(3)
monument	110/62	foundationyear(23), location(20), visiting(14), surrounding(13), history(12), purpose(11), designer(11), height(9), istouristattraction(8), design(8)
mountain	116/57	visiting(35), location(23), surrounding(18), height(18), features(14), naming(8), temperature(6), size(5), history(5), mostattraction(4)

museum	135/30	visiting(52), location(22), foundationyear(22), mostattraction(10), features(9), history(8), arts(5), owner(5), events(4), naming(4)
opera house	115/65	visiting(33), foundationyear(20), location(17), events(17), mostattraction(10), surrounding(10), designer(9), owner(8), design(8), capacity(6)
palace	137/51	location(26), history(23), foundationyear(21), visiting(17), surrounding(12), design(12), size(12), owner(8), founder(7), constructioninfo(6)
park	144/36	visiting(41), features(39), location(26), size(12), surrounding(10), foundationyear(9), mostattraction(5), type(4), events(4), maintenance(3)
parliament	48/16	visiting(13), location(7), foundationyear(7), surrounding(6), design(5), history(4), mostattraction(3), type(3), designer(2), size(2)
peak	128/54	visiting(58), location(27), height(20), surrounding(18), temperature(10), features(6), history(6), istouristattraction(5), naming(5), type(4)
prison	24/13	foundationyear(7), design(5), history(4), capacity(4), location(3), prisonersinfo(3), type(2), size(2), visiting(2), prisoners(2)
railway	14/35	location(7), travelcost(7), travellers(6), travelling(4), technique(3), foundationyear(3), status(2), owner(2), length(2), capacity(2)
railway station	74/48	foundationyear(22), location(19), visiting(7), naming(5), travellers(5), history(5), design(5), type(4), surrounding(4), travelling(4)
residence	102/53	design(28), location(21), foundationyear(15), visiting(13), habitants(10), purpose(8), history(8), size(7), type(6), owner(6)
river	87/102	visiting(25), location(19), length(18), surrounding(16), purpose(15), origin(12), status(9), features(8), waterinfo(7), events(6)
ski resort	111/44	visiting(40), features(26), location(18), surrounding(8), height(7), events(6), size(6), temperature(5), history(5), foundationyear(5)
square	141/37	visiting(28), location(26), surrounding(22), history(17), foundationyear(14), naming(13), mostattraction(8), istouristattraction(8), visitors(7), features(7)
stadium	91/66	location(30), events(25), visiting(23), capacity(15), foundationyear(11), surrounding(11). owner(6), size(6), type(6), constructioninfo(4)

temple	117/51	location(28), visiting(22), foundationyear(17), design(17), visitors(12), surrounding(8), religioninfo(8), size(7), history(6), founder(6)
tower	100/56	visiting(24), location(21), foundationyear(18), design(16), purpose(15), height(14), features(6), designer(5), founder(5), naming(4)
university	81/99	location(32), studentsinfo(26), subjectsofferred(25), visiting(11), foundationyear(10), size(9), maintenance(8), history(6), type(4), features(4)
village	71/33	habitants(18), surrounding(17), visiting(16), location(11), features(10), mostattraction(5), size(5), naming(4), type(4), visitors(2)
volcano	88/90	eruptioninfo(42), visiting(26), location(21), surrounding(20), height(15), status(14), foundationyear(7), naming(6), history(4), istouristattraction(4)
zoo	135/24	visiting(41), features(34), location(19), foundationyear(14), size(10), mostattraction(8), surrounding(5), owner(5), events(4), naming(3)

Table 4.4: Statistics about attributes in each entity type.

We define the popularity of an attribute for an entity type as the number of questions categorized under this attribute for that particular entity type. Attribute popularity indicates how important the type of information represented by the attribute is for the particular entity type. For instance, the attribute *visiting* is the most popular attribute (has the maximum number of 52 questions) in the entity type *museum* (see Table 4.4). This indicates that it is most important for people to know how much the entry to the museum costs or when the museum opens. This information is more relevant than, e.g. knowing when the museum is built. However, if we look at the entity type *house*, we can see that the same attribute *visiting* is not the most popular one. It occurs at position five with only 11 questions. For this entity type people seem to be interested most in knowing the *design* of the house and less in the information related to visiting.

The remaining 35% of the questions are spread over the remaining 41 attributes as shown in the right column (*below top ten*) of Table 4.3. From this list we can observe that there are some specific attributes which are only present for a particular entity type or for a few entity types. For instance, as shown in Table 4.4, the attribute *eruptioninfo* is associated only with the entity type *volcano*. This attribute contains questions related to the eruption of different volcanos *"How often this volcano erupts in fuji?", "Has it erupted before?", "When was the last time it erupted?"*. From Table 4.4 we can see that this attribute is also the most popular for the *volcano* entity type, while information related to *visiting*, *location*, etc., which is generally most frequently asked for, comes after *eruptioninfo*.

From this we can conclude that even though entity types share the *top ten* attributes from Table 4.3, the differences in attribute popularity indicate that these are not equally important for all entity types. Therefore, the question arises as to whether each entity type has an associated specific set of attributes, or whether there are similar entity types that can be grouped according to which information is required to describe them. This is our second research question.

4.3.2 Is each set of attributes specific to a particular entity type or shared between several entity types?

To address this question we compare different entity types using different sets of attributes from Table 4.3 and investigate the degree of similarity between them based on these different sets. Note that our aim is not to compare entity types using their attributes and identify similar and dissimilar entity types. Such an analysis could be performed by computing, e.g., the dice coefficient or the cosine metric over the vector space representations of the attributes. However, our aim is to compare how (dis)similar different entity types are on the different set of attributes and to draw conclusions about specific and shared attributes.

For doing this analysis, we use Kendall's Tau rank correlation coefficient as a metric which indicates similarity between entity types, while considering the attribute popularity ranking in their attribute sets as well. Kendall's Tau correlation coefficient is equal or close to 1, if two events are highly correlated in rankings and close to 0 if there exists little or no correlation between the ranks of the two events.

The attributes are ranked according to their popularity, i.e. the number of questions contained under each attribute. If shared attributes of two different entity types have similar rankings, this is an indication that these attributes are of similar importance for both entity types. Kendall's Tau for these entity types will return a high correlation for the pair, and we will refer to such entity types as similar. A difference in attribute rankings between different entity types will lead to low Kendall's Tau correlation indicating that the entity types are dissimilar.

In our analysis we aim to identify the entity types whose correlation coefficient is higher than the mean correlation and those that are correlated with a coefficient lower than the mean. We also want to understand whether there are entity types that are only highly correlated for a set of attributes and whose correlation potentially drops for another set of attributes.

In our analysis we report comparisons based on three different sets of attributes: *all*, *top ten* and *below top ten*. *All* is the entire set of attributes shown in Table 4.3 (attributes from columns *top ten* and *below top ten*). The *top ten* attributes include the top ten attributes that are shared between most of the entity types and thus more likely to render similarity. The *below top ten* set contains all remaining attributes.

If *all* attributes are taken, there is in general a correlation between entity types in both urban and rural categories. On average the urban entity types correlate with *0.55* (median *0.57*) and the rural types with *0.53* (median *0.53*). The mean and median correlation coefficient lie close to each other, which indicates that the distribution of highly correlating entity types is similar to that of entity types with low correlation.

Tables 4.5 and 4.6 show three groups of entity type pairs in rural and urban areas respectively. In the first column the entity type pairs whose correlation coefficient is higher than the mean for *all* attributes are shown. The second column shows the entity type pairs whose correlation is higher than the mean in the *top ten* attributes, but drops below the mean for the *below top ten* attributes. Finally, in the third column the pairs which are correlated with the coefficient lower than the mean for *all* attributes are presented.

From Tables 4.5 and 4.6 we can see that high correlation in attributes is always present when the entity types have, e.g., the same look, design or the same purpose. For the urban areas we have entity types such as *church, basilica, abbey, cathedral* and *temple* which correlate with a coefficient higher than the mean of *0.55* when *all* attributes are considered. A similar picture can be drawn for other urban entity types such as *house-residence*, *building-residence* or *museum-operahouse*. In rural areas, entity types related to mountainous areas (e.g. *glacier, mountain, peak, volcano, ski resort* etc.) are correlated above the mean of *0.53*. The same is valid for water bodies like *canal, lake, river*, etc.

The results confirm our intuition that high similarity in attributes is found between entity types which share features like look, design, purpose, etc. The correlation coefficient is always low for entity types which do not share these aspects, such as the ones shown in the third column of Tables 4.5 and 4.6. This indicates that aspects used for categorizing entities into entity types also play a role in deciding which entity types are similar for purposes of describing them.

However, the second column of the tables also highlights the importance of shared ideas of what information is relevant for describing geo-located entities for entity type similarity (Section 4.3.1). The entity types shown in the second column of the tables are correlated above the mean

Table 4.5: Kendall's Tau correlation results for the urban entity types.

Always High Correlation	High Correlation in Top Ten and Low Correlation in Below Top Ten Attributes	Always low correlation
church;temple, abbey;temple, building;tower, building;residence, abbey;basilica, house;residence, abbey;church, basilica;temple, basilica;cathedral, museum;operahouse, house;palace, gallery;museum	house;prison, museum;railway, basilica;railway, monument;prison, university;cathedral, prison;palace, prison;cathedral, cemetery;aquarium, railwaystation;cathedral, cemetery;parliament	prison;aquarium, monument;railway, railway;cathedral, university;railway, museum;prison, temple;railway, railway;operahouse, parliament;railway, abbey;railway, tower;railway

Table 4.6: Kendall's Tau correlation results for the rural entity types.

Always High Correlation	High Correlation in Top Ten and Low Correlation in Below Top Ten Attributes	Always low correlation
mountain;volcano, mountain;skiresort, mountain;peak, peak;volcano, peak;glacier, mountain;glacier, glacier;skiresort, park;garden, lake;canal, canal;river	village;river, lake;island, canal;volcano, island;river, mountain;river, peak;lake, island;garden, canal;glacier, skiresort;garden, mountain;canal	lake;volcano, canal;island, lake;glacier, glacier;river, canal;village, lake;garden, village;garden, glacier;village, island;glacier, park;cave

for *top ten* attributes and their correlation drops below the mean when *below top ten* attributes are used.

4.3.3 Discussion

Our analyses have shown that people taking the role of a tourist do share ideas as to what information is relevant for describing geo-located entities. This result is expected given the findings of cognitive psychology research on conceptualization as discussed in Chapter 1. More importantly, however, we identified a set of attributes or information types that reflects these ideas. These findings prompted the question, whether each entity type has a specific set of information associated with it, or whether there are attributes shared by several entity types.

We found that some entity types are similar and not only share the information types associated with them, but also the importance ranking of these types. Such similar entities were mostly entities of similar purpose, look or design (e.g. churches and temples, rivers and lakes, etc.). However our results also show that some entity types that are not similar in purpose, look and design still have attributes in common and thus are similar when considering these attributes. In these cases people will refer to the shared set of frequently requested attributes we identified.

These findings lead us to retain our first research hypothesis, which states that entity types are characterized by sets of attributes, some of which are type specific and others of which are shared between different types. In this thesis we use these results to compare the information types indicated by humans to the information types extracted from collection of existing text resources (see Chapter 5). With this we test our second research hypothesis and establish whether there is a similarity between the information types obtained through the online survey and the types included in existing text descriptions. We aim to use such existing text descriptions to derive geo-located entity type models. The comparative analysis reveals that one could use the similarity scores between the entity types to create hierarchical groups of types, e.g. using machine learning techniques. The hierarchical groups could be used in cases where there are not enough textual resources available for a particular entity type. In this case the entity type model for this particular type could be derived from the textual descriptions of other types belonging to the same group. We explore this idea in Chapter 7.

4.4 Summary

In this chapter we investigated which information types (attributes) humans associate with geo-located entities from urban and rural landscape. We identified a set of attributes that are relevant for any entity type, but also found that an appreciable proportion of attributes is entity type specific. Even in the set of shared attributes, not all attributes were equally important for each entity type. Based on the importance ranking of information types for each entity type, we were able to identify similar entity types. These results will guide the acquisition of descriptions of geo-located entities to be used for entity type modeling.

CHAPTER 5

Building Corpora for Geo-located Entity Types

In Chapter 4 we showed that humans taking the role of a tourist have a set of attributes for which they would like to know the values when they see a geo-located entity of a specific type. We identified a set of attributes which were relevant for any entity type, but also found that an appreciable proportion of attributes is entity type specific. Even in the set of shared information types, not all information types were equally important for each entity type.

If humans use sets of attributes to describe geo-located entity types, then, we hypothesize that the existing written descriptions of geo-located entities will also address these attributes (cf. Chapter 1, Section 1.3). In this chapter we investigate Wikipedia as a possible resource containing written descriptions of geo-located entities. Since our investigation confirms that attributes humans identify as relevant are also found in Wikipedia articles, we propose a method for deriving *entity type corpora* from Wikipedia. We will use these entity type corpora as a basis for building entity type models – as described in subsequent chapters. In this chapter we introduce an automatic method for categorization of Wikipedia articles according to the entity type they describe. We then report an evaluation of this categorization and compare it to existing categorization methods.

5.1 Wikipedia as a resource for building entity type text corpora

An entity type text corpus is a collection of texts or articles about entities of specific type, such as *church, bridge, river, etc.* Entity type text corpora will be used as a basis for deriving entity type models, which we will integrate in our summarizer for further investigation (see Chapter 7).

We hypothesize that Wikipedia may be a suitable web resource for collecting entity type text corpora, since it contains high quality descriptions of geo-located entities. Wikipedia would be a suitable resource for building entity type corpora if the information contained in its articles related to geo-located entities matches the information identified as relevant by human participants in our experiment described in Chapter 4. To establish whether this is the case we compared sets of attributes identified in the experiment from Chapter 4 to those extracted from Wikipedia articles about the same entity type.

We do this by analyzing English Wikipedia articles about different geo-located entities from the same entity type for recurring descriptions of the entity attributes. For example, the attribute *location*, which refers to where an entity is situated, is a frequently addressed attribute in Wikipedia articles. The Wikipedia article for *Parc Guell*, for instance, says that the park is *situated on the hill of el Carmel in the Gracia district of Barcelona, Catalonia, Spain.* Similarly, for *Green Park*, the attribute *location* is supplied by the statement that *it lies between London's Hyde Park and St. James's Park.* In this case we would say that both articles from the corpus for the entity type *park* contain descriptions addressing or supplying attribute *location*.

The analysis was conducted with Wikipedia articles automatically categorized by entity type. We used *Is-A* patterns to categorize Wikipedia articles by entity type such as *church, bridge, river, etc.* (see Section 5.2.1).

To carry out the analysis, we first randomly selected 15 entity types from both the urban and rural categories. Specifically, 7 entity types were selected from the urban category (cathedral, gallery, museum, opera, palace, stadium, and university), whereas 8 were chosen from the rural category (beach, cave, island, lake, mountain, park, river, and volcano). For each entity type, we selected 10 Wikipedia articles about entities of this type, resulting in a manually analyzed set of 150 articles. To select the articles we randomly picked from each entity type corpus 10 articles which where correctly categorized by our *Is-A* pattern approach (see Section 5.2.1) and which contained at least 10 sentences in the first paragraph. [1]

Once the articles were retrieved, three annotators analyzed the sentences of the first paragraph of each article. The objective of the analysis was to determine if the attributes identified as relevant by the MTurk workers in the image labeling task reported in Chapter 4 can also be found in Wikipedia articles. The most relevant attributes identified by MTurk workers were shown in

[1] We choose to analyze the first paragraph only instead of the entire Wikipedia article to reduce the labour needed for the experiment. Our random analyses of Wikipedia articles showed that the most relevant information is contained in the first paragraph, provided it is of sufficient length. Therefore, we only select Wikipedia articles if their first paragraph is at least 10 sentences or longer.

Table 4.4 (Chapter 4, Section 4.3.3), where the ten most frequently addressed attributes for each entity type were listed. For each sentence from the first paragraph of Wikipedia articles, annotators were asked to determine whether that sentence describes one or more attributes from this list and, if so, which.

Table 5.1 shows the result of this analysis for each entity type analyzed. In the analysis we count how many times the ten most frequent attributes found through the MTurk workers (see Table 4.4 in Chapter 4) occurred in the ten Wikipedia articles for each entity type. The numbers within the brackets attached to each attribute indicate how many times that attribute occurred in the ten Wikipedia articles. We use these numbers to sort the attributes in descending order. Attributes with "0" in the brackets indicate that the attribute did not occur in the Wikipedia first paragraph. We also list the ten most frequent attributes from the MTurk survey to have direct comparison.

As can be seen, most of the attributes can be identified in the sentences of the first paragraph of Wikipedia articles. There are only 17 cases out of 150 where an attribute identified relevant by the MTurk workers does not occur in the Wikipedia paragraphs. For instance, *temperature* is such a case. This attribute is a detail difficult to find in the first paragraph of the Wikipedia articles related to islands or mountains, though it is interesting according to the MTurk workers.

Moreover, it is worth noting that if ordered by frequency the order of attributes found in the Wikipedia articles varies with respect to the MTurk analysis. While the MTurk workers gave preference to the visiting and location attributes for these selected entities, these two being the most and the second most frequent, in Wikipedia articles we found that these attributes, despite being relevant, are not always among the two most frequent ones. Here, we observe that the type of information more frequent for each entity type varies depending on the entity type itself. For instance, for the entity type *gallery* the most frequent attributes are *arts* and *events*. This means that articles about galleries focus more on the features of the exhibitions they show, rather than on any other type of information, e.g., concerning visiting information such as the entry fees, opening hours, etc., as is mainly the case in the MTurk analysis. One reason for this might be that the MTurk workers were asked to take the role of a tourist when proving the questions (see Section 4.2.2).

We also analyzed whether the popularity of the attributes obtained from Wikipedia articles correlates with the popularity of the attributes obtained through the MTurk workers. To do this we compute the Kendall's Tau correlation coefficient as in Chapter 4. Table 5.2 shows the resulting Kendall's Tau correlation between the Wikipedia and the MTurk results for each of the entity types analyzed.

Entity Type	Attributes
beach	features(34), visiting(32), surrounding(23), location(14), visitors(6), mostattraction(6), naming(4), depth(2), length(2), foundationyear(0)
beach (MTurk)	visiting(44), location(24), surrounding(23), features(18), visitors(12), mostattraction(7), naming(4), depth(4), length(4), foundationyear(4)
cathedral	history (30), location(16), design(16), visiting(7), foundationyear(5), preacher(3), founder(2), size(1), height(1), events(0)
cathedral (MTurk)	location(30), visiting(23), foundationyear(16), history(12), founder(9), design(8), size(8), preacher(6), height(6), events(6)
cave	location(11), features(11), naming(11), firstdiscovery(8), visiting(6), history(6), mostattraction(5), depth(5), surrounding(0), design(0)
cave (MTurk)	visiting(46), location(25), surrounding(15), features(15), firstdiscovery(13), history(12), naming(7), mostattraction(6), depth(6), design(5)
gallery	arts(30), events(24), location(19), foundationyear(8), type(7), purpose(6), visiting(5), surrounding(2), design(1), mostattraction(0)
gallery (MTurk)	visiting(43), location(23), arts(21), foundationyear(16), surrounding(10), design(8), events(5), type(5), purpose(5), mostattraction(5)
island	location(13), features(9), size(7), visiting (6), habitants(5), surrounding(3), history(3), mostattraction(1), temperature(0),istouristattraction(0)
island (MTurk)	visiting(21), location(19), habitants(19), features(15), size(13), surrounding(11), temperature(6), mostattraction(5), istouristattraction(5), history(3)
lake	features(36), visiting(18), surrounding(14), location(11), size(11), mostattraction(5), depth(4), purpose(3),visitors(2), status(1)
lake (MTurk)	visiting(61), surrounding(32), location(29), size(12), depth(10), features(7), mostattraction(6), purpose(6), visitors(3), status(3)
mountain	naming(13), visiting(12), location(11), height(10), surrounding(9), size(8), history(5), mostattraction(5), features(1), temperature(0)
mountain (MTurk)	visiting(35), location(23), surrounding(18), height(18), features(14), naming(8), temperature(6), size(5), history(5), mostattraction(5)
museum	features(23), history(22), location(13), arts(10), foundationyear(7), visiting (6), owner(3), mostattraction(2), events(2), naming(0)
museum (MTurk)	visiting(52), location(22), foundationyear(22), mostattraction(10), features(9), history(8), arts(5), owner(5), events(4), naming(4)
opera house	location(16), foundationyear(11), events(9), designer(7), owner(7), design(4), surrounding(3), visiting(2), capacity(2), mostattraction(0)
opera house (MTurk)	visiting(33), foundationyear(20), location(17), events(17), mostattraction(10), surrounding(10), designer(9), owner(8), design(8), capacity(6)
palace	history(62), visiting(17), location(14), design(8), foundationyear(5), surrounding(5), owner(5), size(4),founder(4), constructioninfo(4)
palace (MTurk)	location(26), history(23), foundationyear(21), visiting(17), surrounding(12), design(12), size(12), owner(8), founder(7), constructioninfo(6)

park	features(21), visiting(15), location(12), size(9), foundationyear(8), mostattraction(5), type(5), surrounding(2), events(1), maintenance(0)
park (MTurk)	visiting(41), features(39), location(26), size(12), surrounding(10), foundationyear(9), mostattraction(5), type(4), events(4), maintenance(3)
river	location(17), surrounding(11), origin(9), status(9), features(9), length(8), visiting (0), purpose(0), waterinfo(0), events(0)
river (MTurk)	visiting(25), location(19), length(18), surrounding(16), purpose(15), origin(12), status(9), features(8), waterinfo(7), events(6)
stadium	capacity(13), location(11), type(10), events(9), foundationyear(8), surrounding(5), owner(4), size(3), constructioninfo(3), visiting(0)
stadium (MTurk)	location(30), events(25), visiting(23), capacity(15), foundationyear(11), surrounding(11), owner(6), size(6), type(6), constructioninfo(4)
university	history(22), location(20), studentsinfo(11), size(7), foundationyear(5), type(5), features(3), subjectsofferred(2), visiting (1), maintenance(0)
university (MTurk)	location(32), studentsinfo(26), subjectsofferred(25), visiting(11), foundationyear(10), size(9), maintenance(8), history(6), type(4), features(4)
volcano	surrounding(15), eruptioninfo(12), location(12), status(9), height(6), foundationyear(6), visiting(5), history(4), naming(3), istouristattraction(1)
volcano (MTurk)	eruptioninfo(42), visiting(26), location(21), surrounding(20), height(15), status(14), foundationyear(7), naming(6), history(4), istouristattraction(4)

Table 5.1: Results of analysis of Wikipedia articles.

The results obtained show high correlation coefficients for most of the entity types. The highest correlation is obtained for the entity types *beach* and *park* and the lowest for *university*. On average the correlation coefficient is *0.52* showing in general a high or significant correlation between the entity types analyzed. With 10 instances as is the case in our comparison Kendall's Tau correlation is regarded as significant when it is equal or above *0.46* and one-tailed statistical test is performed. For two-tailed analysis the significance value is *0.51*[2].

These observations show that the geo-located entity information that people request when they see an entity of a particular type corresponds to that which people use when they write descriptions about entities of these types.[3] Therefore, we can retain our second research hypothesis, which states that the attributes characterizing entity types are addressed in existing online collections of text resources, so that entity type models of geo-located entities can be derived from these text resources. Our analysis shows that Wikipedia is a suitable resource for building entity type text corpora from which entity type models can be derived.

[2] http://www.answers.com/topic/critical-values-for-kendall-s

[3] Wikipedia entries may refer to properties not included in the human-compiled list but these properties such as pronunciation are common in Wikipedia and are included under the *name* attribute.

Table 5.2: Kendall's Tau correlation results of attributes from Wikipedia articles and MTurk survey.

Annotator	Entity types	Kendall's Tau	Sentence count
A1	beach - beach (MTurk)	0.82	10
A1	gallery - gallery (MTurk)	0.33	13
A1	island - island (MTurk)	0.6	10
A1	palace - palace (MTurk)	0.73	13
A1	university - university (MTurk)	0.2	13
A2	cathedral - cathedral (MTurk)	0.64	12
A2	cave - cave (MTurk)	0.46	10
A2	museum - museum (MTurk)	0.37	14
A2	lake - lake (MTurk)	0.73	10
A2	opera house - opera house (MTurk)	0.28	12
A3	mountain - mountain (MTurk)	0.46	11
A3	park - park (MTurk)	0.82	10
A3	river - river (MTurk)	0.42	11
A3	stadium - stadium (MTurk)	0.37	11
A3	volcano - volcano (MTurk)	0.6	12
average		**0.52**	**11.4**

5.2 Building entity type text corpora from Wikipedia

5.2.1 Entity type text corpora collection procedure

Having shown that Wikipedia is a suitable resource from which to build entity type corpora, we now describe the process of collecting entity type corpora from Wikipedia articles. In this process we aim to identify Wikipedia articles about geo-located entities and categorize them by entity types. For example, an article about the *Westminster Abbey* should be categorized under the entity type *church*.

We use a Wikipedia dump (English Wikipedia dump from 24/07/2008) and automatically categorize each article in the dump by entity type. We follow a two-step approach to perform the categorization. In the first step we use an automaton to find a sentence segment likely to contain the category. In the second step another automaton is run on the sentence segment from the first step to identify the category.

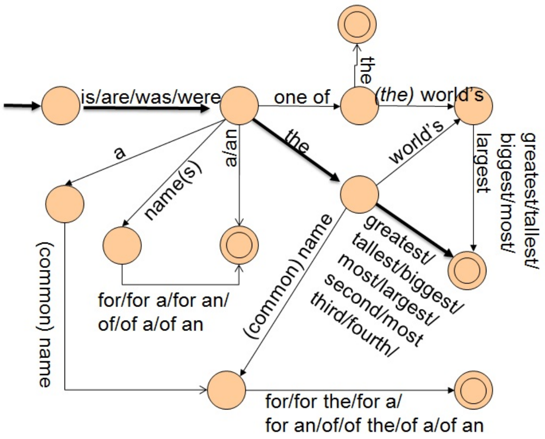


Figure 5.1: Step 1 automaton.

Step 1

In the first step we take each Wikipedia article, split it into sentences and keep only the first ten sentences. Then each of these ten sentences is checked for the occurrence of an *Is-A* pattern using the automaton shown in Figure 5.1. Our patterns are in the fashion of Hearst (1992) and Mann (2002) who used manually written patterns to extract hyponyms from large text corpora. If we find the entity type, e.g. in the second sentence, we stop and do not parse the subsequent sentences.

The automaton starts with the arrow that does not contain a label. The terminal states are denoted by double circles. The arrows in the figure indicate the transitions between two states. The concatenation of the terms on the transition arrows from the start state to a terminal state form our *IS-A* patterns. For instance, a possible pattern is *is the greatest* which can be constructed when the bold arrows are followed. The patterns are strings. Terms on the transition arrows separated by / indicate that there is more than one possible way to make a valid transition. The transition arrow marked with the label *is/are/was/were* indicates that the transition is valid if the sentence contains *is*, *are*, *was* or *were*. Transition labels containing *(...)* indicate optional terms. For instance, the transition label *(the) world's* could take the form *the world's* or only *world's*, taking out the optional term *the*.

If the automaton is given a sentence, it looks for the occurrences of the terms labeled in the transitions. If it finds any, it performs a transition from the current state to the next one. If the new state is not a terminal one, then it continues to the next state by again checking whether or

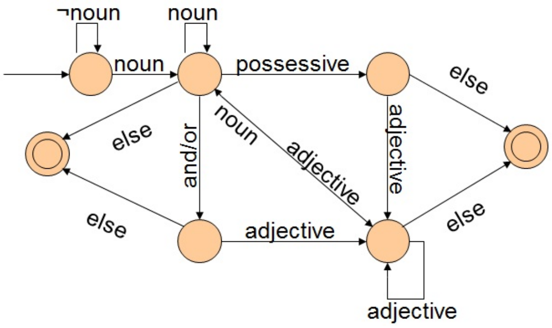


Figure 5.2: Step 2 automaton.

not any term in the current transition label occurs in the sentence. If it finds a terminal state, it stops and returns the pattern that was found in the sentence.

For example, for the article about *Westminster Abbey* the first sentence is:

The Collegiate Church of St Peter at Westminster, which is almost always referred to by its original name of Westminster Abbey, is a large, mainly Gothic church, in Westminster, London, just to the west of the Palace of Westminster.

For this sentence the automaton finds the pattern *is a* and terminates. After this step we keep the sentence part that occurs after the *Is-A* pattern and discard the rest. The remaining part after deletion is:

large, mainly Gothic church, in Westminster, London, just to the west of the Palace of Westminster.

It should be noted that the automaton can find more than one pattern for a segment string. However, it only returns the longest one.

Step 2

In order to find the category of the Wikipedia article we apply noun phrase search on the sentence segment from step 1. The texts are preprocessed by tokenization and POS tagging. Noun phrases are identified based on rules composed of different sequences of POS tags as shown in the automaton in Figure 5.2.

The *noun* input in the automaton is the starting POS tag for any sequence of rules. Therefore the automaton starts by searching for the first occurrence of a noun in the shortened version of the sentence. Following the first occurrence of a noun, the automaton checks for further POS tags which can be combined with a noun, e.g. *noun noun*, *noun possessive*, *noun possessive adjective noun* etc. The automaton terminates if *else* is found. After the automaton reaches the term *else* it takes from the sequence of words the noun that occurs at the end of the sequence. For instance, if the automaton finds the sequence *noun adjective noun else* it takes the second noun as the category. It can also happen that there is more than one consecutive noun in the sequence such as *noun adjective noun noun else*. In this case the automaton returns the two nouns at the end of the sequence as the category name.

For the above first sentence for *Westminster Abbey*, for instance, the POS tagged version of the shortened sentence looks like[4]:

large/ADJ mainly/ADV Gothic/ADJ church/N ,/, in/P Westminster/PN ,/, London/PN ,/, just/ADV to/P the/DET west/N of/P the/DET Palace/PN of/P Westminster/NP

In this shortened sentence the automaton finds as the first noun *church/N else*. After *church* it does not find an allowable POS and follows the *else* transition leading to the sequence *noun else*. Therefore it terminates and returns *church* as the category, or entity type, for the article about *Westminster Abbey*.

In this way about 2.1 out of 2.7 millions articles were automatically categorized.[5] Altogether 40648 categories have been identified by our procedure. However, not all of these categories are related to geo-located entities, e.g., there are categories such as *politician*, *leader*, *machine*, etc., which are not useful for our purposes. We manually filtered all identified categories in order to retain the ones describing geo-located entities. This resulted in a set of 734 categories. From that set we retained the categories that are associated with at least 50 Wikipedia articles. That reduced the set to 175 categories. Finally, we manually assigned specific categories such as *suspension bridge* to a more general category *bridge*. In this way we collected 107 categories containing articles about places around the world (see Figure 5.3).

[4] POS tags are obtained using the original longer version of the sentence. After the truncating the original sentence we also truncate the corresponding parts from the POS tagged sequence for the sentence.

[5] We have checked several of these 0.6 millions uncategorized articles and found that the first paragraph contained only a single sentence. From this sentence our automaton in step 1 failed to find any segment likely to contain the category name.

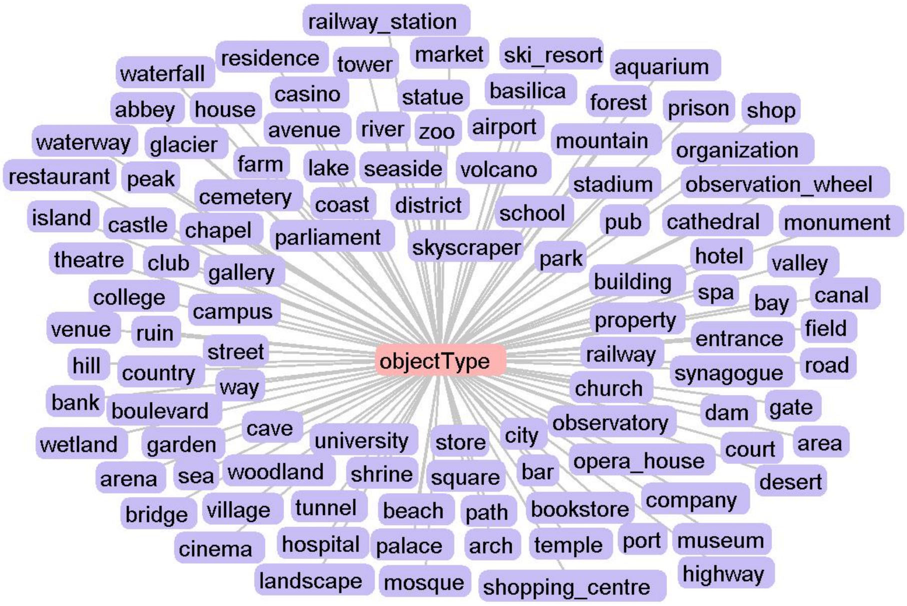


Figure 5.3: Entity types identified from Wikipedia

5.2.2 *Evaluation*

We manually evaluated our categorization of Wikipedia articles by different entity types. First, we randomly selected 35 entity type corpora from the set of 107 described in Section 5.2.1. Next, from each of these 35 entity type corpora we randomly selected 50 articles. Then we checked for each of these articles whether it is correctly or wrongly assigned to that particular entity type. Finally, we calculated an accuracy value for each entity type which is the the proportion of correctly assigned articles in the set of 50 articles. The results of this evaluation are shown in Table 5.3.

We observed the maximum (100%) accuracy in the *railway station, ski resort, shopping center, mountain, highway, railway station, island, village, arena, aquarium, bridge, castle* and *airport* entity type corpora and the minimum accuracy (62%) in the *landscape* entity type corpus. Overall accuracy of our *Is-A* pattern is 91%.

The main problem that causes the *Is-A* patterns to fail is ambiguity. Some articles have patterns that are matched by our application, but do not entail the entity type of the entity the article is about. Rather, the matched pattern contains a piece of information about something else related to the entity.

Table 5.3: Entity types and the accuracy of the categorization.

Entity Type	Accuracy	Entity Type	Accuracy
shopping center	1.0	**ski resort**	1.0
mountain	1.0	**highway**	1.0
railway station	1.0	mosque	0.66
waterfall	0.98	street	0.94
landscape	0.62	restaurant	0.88
island	1.0	**airport**	1.0
area	0.84	volcano	0.96
village	1.0	zoo	0.96
arena	1.0	wetland	0.95
bank	0.94	monument	0.86
university	0.98	building	0.8
park	0.98	gallery	0.88
museum	0.98	canal	0.98
temple	0.94	tower	0.86
prison	0.9	residence	0.76
aquarium	1.0	**castle**	1.0
bridge	1.0	waterway	0.98
river	0.97	**average accuracy**	**0.91**

Table 5.4 shows the resulting entity types and the number of articles categorized under each entity type. The first column of Table 5.4 lists the entity types which we manually marked as urban types. There are in total 80 entity types which fall into the urban category. The second column shows the entity types (27 in total) manually categorized as rural types. We use these corpora to derive entity type models.

5.2.3 Related Work

A similar categorization of Wikipedia articles but into a different set of categories has already been investigated. The result of this categorization is stored in an ontology called DBpedia[6] (Auer & Lehmann 2007, Auer et al. 2007). Auer & Lehmann (2007) and Auer et al. (2007) use Wikipedia *infoboxes* to categorize Wikipedia articles by subject. Infoboxes in Wikipedia are templates and are used to present certain summary or overview information about the subject of the article. An infobox for the Wikipedia article about the *Yosemite National Park* is shown in Figure 5.4.

[6] http://www4.wiwiss.fu-berlin.de/dbpedia/dev/ontology.htm

Table 5.4: Entity types (80 urban and 27 rural types) identified by our Is-A patterns along with the number of articles in each corpus.

urban types	rural types
school 15794, city 14233, organization 9393, university 7101, area 6934, district 6565, airport 6493, railway station 5905, company 5734, park 3754, college 3749, stadium 3665, road 3421, country 3186, church 3005, way 2508, museum 2320, railway 2093, house 2018, arena 1829, club 1708, shopping centre 1509, highway 1464, bridge 1383, street 1352, theatre 1330, bank 1310, property 1261, castle 1022, court 949, hospital 937, skyscraper 843, hotel 741, garden 739, building 722, market 712, monument 679, port 651, temple 625, square 605, store 547, campus 525, palace 516, tower 496, cemetery 457, cathedral 402, residence 371, gallery 349, prison 348, canal 332, restaurant 329, observatory 303, zoo 302, statue 283, venue 269, parliament 258, shrine 256, synagogue 236, bar 229, arch 223, avenue 202, casino 179, waterway 167, tunnel 167, ruin 166, chapel 165, observation wheel 158, basilica 157, cinema 144, gate 142, aquarium 136, entrance 136, opera house 134, spa 125, shop 124, abbey 108, boulevard 108, pub 92, bookstore 76, mosque 56	village 39970, island 6400, river 5851, mountain 5290, lake 3649, field 1731, hill 1072, forest 995, peak 906, bay 899, valley 763, sea 645, beach 614, volcano 426, glacier 392, dam 363, waterfall 355, cave 341, path 312, coast 298, desert 248, ski resort 227, landscape 220, farm 179, seaside 173, woodland 154, wetland 151

Figure 5.4: Wikipedia infobox for Yosemite National Park.

```
{{Infobox Protected area
| name            = Yosemite National Park
| iucn_category   = Ib
| map             = US_Locator_Blank.svg
| map_caption     = Map of the USA
| locator_x       = 20
| locator_y       = 84
| location        = [[California]], [[United States]]
| nearest_city    = [[Mariposa, California|Mariposa]]
| lat_d           = 37.8499232
| long_d          = -119.5676663
| region          = US-CA
| scale           = 300000
| area            = {{convert|761266|acres|km2}}
| established     = October 1, 1890
| visitation_num  = 3,280,911
| visitation_year = 2004
| governing_body  = [[National Park Service]]
| world_heritage_site = 1984
| website         = [http://www.nps.gov/yose/ www.nps.gov/yose]
}}
```

Table 5.5: DBpedia Categories related to locations.

river, historic place, lake, mountain, building, airport, station, skyscraper, bridge, stadium, shopping mall, lighthouse, hospital, historic building, protected area, lunar crater, world heritage site, park, island, wine region, ski area, cave, populated place, country, municipality, city, road, company, radio station, school, university, college, soccer club, educational institution

The authors use the first line of the infobox to categorize all Wikipedia articles by subject. For the *Yosemite National Park*, for instance, the subject is *protected area*. However, the categorization of places by subjects in the current version is not precise enough to be used as entity types. The subjects used in the infoboxes contain more abstract descriptions or generalizations and thus, do not cover all subjects identified by our procedure. The DBpedia ontology contains only 34 entity types which are related to locations (see Table 5.5).

Another approach which uses infoboxes for categorizing Wikipedia articles is described in Wu & Weld (2007). Wu and Weld argue that not all Wikipedia articles have an infobox and are therefore not covered by the DBpedia ontology. To categorize articles without infoboxes the authors use the Wikipedia category lists[7] of these articles and compare them to subjects collected from infoboxes. More precisely, they first scan the Wikipedia articles with infoboxes and extract from them the infobox subjects. Next, they parse the Wikipedia category names of the articles that do not have infoboxex and extract from them head nouns. Finally, they compare the infobox subjects with the head nouns and, if there is a match, assign each article the subject from the infobox as the category. However, this approach only enhances existing DBpedia categories with new articles. The number of categories is the same and thus, again not suitable for our purposes.

Another similar ontology, the *YAGO* ontology, is described in Suchanek et al. (2007, 2008). The authors categorize each Wikipedia article into one of the categories provided by Wikipedia itself. To do this Suchanek et al. follow a similar idea as in Wu & Weld (2007) and extract from the Wikipedia category lists the head nouns as candidate categories for the article. However, unlike Wu and Weld Suchanek et al. keep only head nouns that are plural. The extracted head nouns are referred as “conceptual categories” for the article.

Furthermore, Suchanek et al. use WordNet to enhance the list of conceptual categories. Each conceptual category is checked WordNet for synsets. A synset is a set of synonyms for a given word (Miller 1995). If a synset is successfully found, then the terms in the synset are taken to

[7] Each Wikipedia article is assigned one to many category names covering the categories the article falls into. These category names are found at the end of each article.

enhance the conceptual category list for each article. The following list shows the conceptual categories for *Eiffel Tower* that the authors identify:

```
Former_world's_tallest_buildings
Michelin_Guide_starred_restaurants_and_chefs
Skyscrapers_in_Paris
artifact
building
restaurant
structure
```

The first three entries are obtained analysing the Wikipedia categories and the following four are obtained from WordNet. The list contains different entity type candidates for *Eiffel Tower*, which can be used as abstract categories for the tower such as *artifact, structure, building, etc*. However, the list also contains wrong entries such as *restaurant* which do not express the entity type of *Eiffel Tower* at all. To identify the correct and specific category or entity type from the *YAGO* ontology the generalized and the incorrect entity types must be filtered out. Given these problems the *YAGO* ontology is currently not suitable for our purposes.

5.3 Summary

In this chapter we first investigated our hypothesis that Wikipedia articles about geo-located entities will mention the attributes we have obtained through our MTurk survey. We have selected 15 entity types and 10 Wikipedia articles for each type and checked whether the top 10 attributes obtained through the MTurk survey also occur in those articles. Our investigation shows that this is indeed the case. Therefore we use Wikipedia articles to construct entity type corpora, collections of Wikipedia articles about geo-located entities of specific types. We reported the entity type corpora collection procedure, which involves categorization of Wikipedia articles by entity type using *Is-A* patterns and manual filtering of articles not related to geo-located entities, resulting in a collection of entity type corpora with 107 different entity types. In an evaluation of this categorization procedure, we demonstrated an overall categorization accuracy of 91%. Note that we have used *Is-A* patterns for extracting the type information because it was easy to implement. However, we plan to also investigate more sophisticated approaches using, e.g., some semi-automated machine learning techniques and aim to compare the results. Finally, we discussed related work and described reasons why their categories are not suitable for our purpose.

CHAPTER 6

Model Summaries[1]

In this chapter we describe how we gathered a set of human generated model summaries. As described in Section 2.3, such a set of model summaries is needed for the computation of the ROUGE evaluation metric, which expresses the n-gram overlaps between automatic and human generated summaries (model summaries). In DUC and TAC (see Chapter 2) model summaries for various domains, such as news, events, etc., are provided for evaluation so that any study involving these domains can reuse the provided data for evaluation purposes. However, such evaluation data does not exist for our domain of geo-located entities. Thus, in order to be able to evaluate our automated summaries we collected our own set of model summaries.

This data set consists of descriptions of images showing geo-located entities along with their corresponding images. The model summaries were generated by eleven humans by extraction of relevant information about geo-located entities from location descriptions found on the social web site *Virtualtourist*.[2] The corpus contains 937 model summaries for 307 different locations. The model summaries are evaluated based on a manual readability evaluation similar to DUC and TAC. The results of this evaluation are comparable with those reported by DUC for the readability assessment of model summaries.

In this chapter we first describe the set of geo-located entities for which we generate model summaries in Section 6.1. Section 6.2 describes previously used methods to collect model summaries and explains how we approach this task. VirtualTourist is described in Section 6.3. We then describe the collection of model summaries in Section 6.4 and report the results of manual readability evaluation of model summaries in Section 6.5.

[1] Some of the results presented in this chapter are published in Aker & Gaizauskas (2010*b*).

[2] www.virtualtourist.com

6.1 The Image Set

The first step in our model summary generation process is to select a set of images showing geo-located entities for which we want to have model summaries. We manually selected a set of 307[3] different images from VirtualTourist showing locations around the world such as *Parc Guell, the London Eye, Edinburgh Castle*, etc. Each image pertains to a different entity.

Next, we manually categorized these images by geo-located entity type. To categorize an image by its entity type, we use its name to retrieve the Wikipedia article, which contains a description of the entity shown in the image. We then apply the Wikipedia article categorization procedure, using *Is-A* patterns as described in Section 5.2.1, to categorize the article about the image in question. The resulting entity type is then assigned to the image. For example, for the image showing *Westminster Abbey* we used the toponym *Westminster Abbey* to retrieve the Wikipedia article about the abbey. The sentence containing *Is-A* pattern ("is a large, mainly Gothic **church**") is then found in the first paragraph of the article, and the entity type *church* is selected from this sentence. Finally, the image showing the Westminster Abbey is categorized under the entity type category *church*.

Not all of our 107 entity types extracted from the entire Wikipedia dump (cf. Section 5.2.1) are covered by the entity types of the image set for which model summaries are generated. Table 6.1 shows the image entity types which constitute a subset containing 60 of the 107 entity types from Wikipedia, and Table 6.2 shows the remaining 47 entity types not covered by the entity types of the image set. From Table 6.1 it can be observed that the entity types covered by our image set mainly describe types of entities which can be regarded as tourist attractions or places to visit. This reflects the nature of VirtualTourist from which our image collection was selected. The users of VirtualTourist are tourists and post their travel images to the site. Thus, it is unlikely that images of entities of types such as schools, hospitals, companies, banks, etc. are captured by VirtualTourist users and posted to the site.

Another reason why only a subset of 107 entity types is covered is that some types are not popular enough. In VirtualTourist there are images of restaurants, for instance, however, they either do not contain descriptions, or if they do, the descriptions consist of only one or two sentences. However, we are interested in entities (images) with descriptions from which we can also derive model summaries (see Section 6.2), so ideally only images with a substantial amount of textual information should be included in our image set. However, this was not

[3] This was the set for which we could get enough textual descriptions to derive model summaries.

always possible, so some images with small descriptions were included. This led to the problem that for some entity types we have only one entity with a single model summary (e.g. *ski resort*). These are the reasons why entity types shown in Table 6.2 are not covered by our image set.

Table 6.1: Image entity types, number of different entities and the number of model summaries for each entity type. Entity types in bold are rural types.

Entity Type	Entity	Summary	Entity Type	Entity	Summary
	Count	Count		Count	Count
mountain	7	18	cemetery	1	4
street	6	13	college	3	5
beach	7	18	house	5	13
cave	1	1	**village**	5	8
zoo	4	10	abbey	1	4
hill	5	16	church	11	32
lake	3	6	museum	17	55
pub	2	2	basilica	2	8
gate	1	4	**glacier**	1	1
temple	8	29	parliament	3	12
statue	2	8	market	2	8
railway	2	3	**ski resort**	1	1
avenue	2	7	stadium	2	5
theatre	2	8	aquarium	2	5
cathedral	11	35	bridge	9	31
opera house	4	16	palace	14	52
railway station	1	4	mosque	4	13
waterfall	3	4	road	1	1
valley	1	1	**island**	7	14
area	5	15	**volcano**	2	4
skyscraper	2	5	monument	10	31
district	3	11	boulevard	1	2
university	6	14	building	9	23
park	14	45	gallery	2	7
venue	1	1	canal	1	6
observation wheel	1	4	tower	8	31
prison	2	7	residence	2	6
castle	14	51	square	18	63
hotel	4	7	garden	4	14
river	8	26	chapel	1	4

Table 6.2: Wikipedia entity types not covered by the image entity type set. Entity types in bold are rural types.

highway, shopping centre, **landscape**, **wetland**, bank, restaurant, waterway, shop, cinema, arena, **desert**, **field**, port, arch, hospital, casino, **peak**, court, **sea**, company, bookstore, **path**, tunnel, **forest**, **bay**, **dam**, entrance, **farm**, synagogue, bar, shrine, **woodland**, country, ruin, club, spa, property, observatory, city, store, way, campus, airport, **coast**, school, **seaside**, organization

6.2 Model Summaries

One way to gather model summaries, similar to that followed in DUC and TAC, is to give the documents (e.g. web-documents retrieved by a search engine using the entity name as a query) to be summarized by a summarization system to human subjects and ask them to generate a summary from these documents about a specific topic. The resulting summary must also not exceed a certain word count. The summaries can be extractive, i.e. created by extraction of entire sentences from the documents, or abstractive, i.e. taking the most relevant sentences from the documents and re-wording or writing a new summary based on the information content they contain (Mani 2001).

Another way of generating model summaries is to collect “information nuggets” from the documents to be summarized. Information nuggets are facts which help humans to assess automatic summaries by checking whether the automatically generated summary contains the fact or not (Voorhees 2003). Aker & Gaizauskas (2008) used nuggets to collect model summaries about geo-located entities. They showed documents to be summarized to human subjects and asked them to collect up to five information nuggets for each image featuring a specific entity (the entity name is taken as topic). Then they compiled the nuggets from different users about the same entity and automatically selected the sentences from the documents where the nuggets occurred. The sentences selected in this way form the model summaries which are finally compared to automatically generated summaries using ROUGE (Lin 2004).

In both cases the generation of model summaries requires that humans read all the documents to be summarized and select the content to go into a summary. However, going through all the documents and reading them is a labour-intensive task (in time and money). In addition, this task requires a certain level of competence while generating summaries. The humans need first to understand the documents, then to identify and discard the information not relevant to the topic and finally to combine the remaining relevant information into a coherent summary. However, to get highly skilled summary generators or abstractors can be very expensive.

To reduce the burden of model summary generation we have used *VirtualTourist* as a resource (see Section 6.3). Collecting model summaries from VirtualTourist image descriptions (VT descriptions) has the following advantages:

VT descriptions are "natural" model summaries. The descriptions are usually written in such a way that they can be directly taken as model summaries in that they are written to concisely convey essential information about the geo-located entity in the image. Furthermore, since they are descriptions spontaneously written by humans and associated with images there is an argument for preferring them as model descriptions to summaries artificially created from documents mentioning the geo-located entity.

VT descriptions are shorter. The descriptions are shorter (average length is 87 words) than documents which need to be summarized if following a DUC-like approach. This reduces the time required for reading to the time which needs to be spent reading only one short image description, or a few short descriptions if the model summary is composed from more than one pre-existing image description.

VT descriptions are more focused on the geo-located entity. The descriptions are focused on the entity shown in the image. If the description, for instance, is about a church, then it usually contains when the church was built, the name of the architect or designer, where the church is located, how to reach the church (public transport), etc. This contrasts with documents retrieved using the location name as a query, which may have a different focus and either not contain the relevant information or contain it in a non-obvious place in the document. In other words, the content selection has already been done in the image descriptions, again reducing the time and effort needed to create model summaries.

Given these advantages it was decided to use VirtualTourist image descriptions as a resource for extracting model summaries. Note that this choice distinguishes the summary evaluation from other summary evaluations in that the reference or model summaries are not derived from the documents from which the automated summaries are themselves generated. A likely consequence of this is that the automated summary scores will be lower than those to be expected when the reference and peer summaries are generated from the same source; however, given the redundancy of information on the web, this effect should not be too high for well-known geo-located entities.

In the following sections we describe VirtualTourist, the model summary collection procedure and the evaluation of the collected model summaries.

Table 6.3: Example description about *Korean War Veterans Memorial* from VirtualTourist descriptions.

The Korean War Veterans Memorial was created to honour members of the United States Armed Forces who served in the Korean War, particularly those that were killed, are still missing, or were held as prisoners of war. The 19 sculptures, designed by Frank Gaylor, are approx 7”3 tall and consist of 14 Army, 3 Marines, 1 Navy and 1 Air Force. The bushes amongst the statues are to symbolise the rough terrain encountered in Korea, the granite strips are to symbolise the obstacles overcome in the war.

6.3 VirtualTourist

VirtualTourist[4] is one of the largest online travel communities in the world where over six million travelers around the world share information in the form of geo-located entity descriptions with each other. It contains 3 million photos of more than 58.000 destinations worldwide. The descriptions are written in English and contain a minimum of 11 and maximum of 2752 words. The average number of words in descriptions is 87.7. An example average description is shown in Table 6.3.

6.4 Collection Procedure

VirtualTourist uses a tree structured schema for organizing the descriptions. The tree has *world* at the root and the *continents* as the direct children of world. The *continents* contain the *countries* which have the *cities* as direct children. The leaves in the tree are the geo-located entities visited by travelers.

We selected from this structure a list of popular cities such as *London, Edinburgh, New York, Venice, Florence,* etc., assigned different sets of cities to different human subjects and asked them to collect up to four model summaries for each entity from their descriptions with length ranging from 190 to 210 words.

During the collection we ensured that the summaries did not have personal information and that they did genuinely describe a geo-located entity, e.g. *Westminster Abbey*. If the descriptions contained personal information, this was removed. In case a description did not have enough words, i.e. the number of words was less than 190, more than one description was used to build a model summary. This process is shown in Figure 6.1. The first summary (summary 1) about

[4] www.virtualtourist.com, site visited 01/02/2008

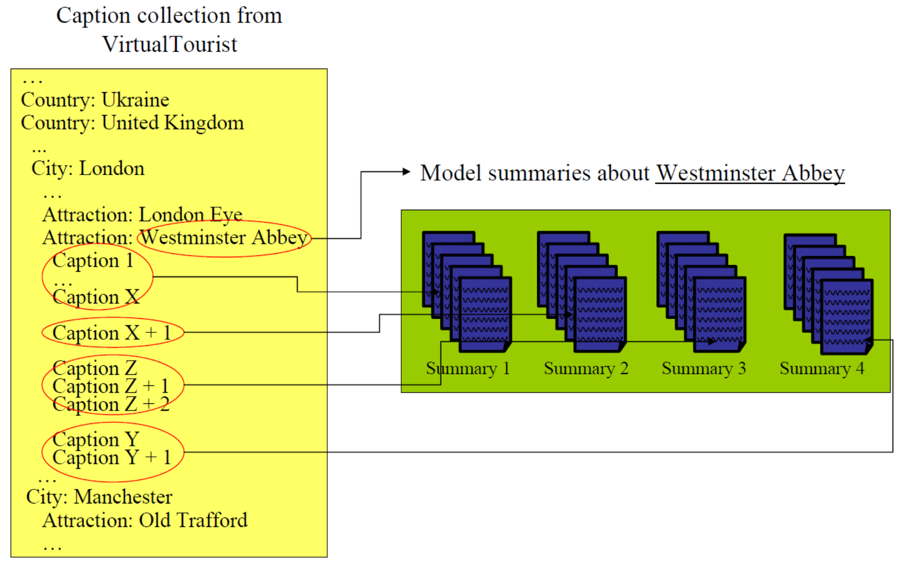


Figure 6.1: Model Summary Collection

Westminster Abbey is generated using the captions *1* to *X*, the second one using the caption *X+1*, the third one using captions *Z* to *Z+2* and the fourth summary using the captions *Y* and *Y+1*. While doing this, we also ensured that the resulting summary did not contain redundant information. In addition, a manually written sentence based on directions and address information, which is given by VirtualTourist users in form of single terms after each description, was optionally added to the model summary. However, this was only done if the description contained less than 190 words. If the description contained more than 210 words, we deleted the less important information. What information is considered less important is subjective and depends on the person collecting the descriptions. Some VirtualTourist descriptions contain sentences recommending what one can do when visiting the place. These sentences usually have the form "you can do XXX". We allowed our model summary collectors to retain such sentences as they contain relevant information about the place. Finally, some descriptions contain sentences which refer to their corresponding images (e.g. "you can see my son and my husband next to..."). We asked our summary gatherers to delete any such sentences referring to images.

We collected model summaries for 307 different geo-located entities from various cities around the world. The number of images/entities with at least four model summaries is 170[5]. 41 have

[5] There are three entities which have five and another entity which has six model summaries. These entities are included in the number of entities with four model summaries.

Table 6.4: Example model summary about Edinburgh Castle with 199 words.

Edinburgh Castle stands on an extinct volcano. The Castle pre-dates Roman Times and bears witness to Scotland's troubled past. The castle was conquered, destroyed, and rebuilt many times over the centuries. The only two remain original structures are David's Tower and St. Margaret's Chapel. Edinburgh Castle - now owned and managed by Historic Scotland - stands 2nd. only to the Tower of London as the most visited attraction in the United Kingdom. Take note of the two heros who guard the castle entrance - William Wallace and Robert the Bruce their bronze statues were placed at the gatehouse in 1929 a fitting tribute to two truely Great Scots. Inside the Castle, there is much to see. It was the seat (and regular refuge) of Scottish Kings, and the historical apartments include the Great Hall, which houses an interesting collection of weapons and armour. The Royal apartments include a tiny room in which Mary, Queen of Scots gave birth to the boy who was to become King James VI of Scotland and James I of England upon the death of Queen Elizabeth in 1603. The ancient Honours of Scotland - the Crown, the Sceptre and the Sword of State - are on view in the Crown Room.

three model summaries, 33 have two and 63 have only one model summary. An example model summary about *Edinburgh Castle* is shown in Table 6.4.

Our data set is much larger than that used by DUC and TAC for summarizer evaluation. In DUC and TAC a maximum of 50 topics, each with four model summaries, is provided for testing purposes. As noted by Owczarzak & Dang (2009), for obtaining a reliable significance test between different summarizers 36 topics would be sufficient. Thus we believe that our data set is big enough to obtain reliable significance tests for our automatically generated summaries.

6.5 Evaluation

The automatic assessment of summary quality (e.g. using ROUGE) requires high quality model summaries, so that final conclusions about the performance of different systems are well grounded. Ensuring the quality of model summaries usually requires manual evaluation of the model summary set. Manual assessment involves presenting the summary to human subjects and asking them to score the presented summary based on criteria expressing the linguistic and content side of the summary. In DUC, for instance, a manual assessment scheme is used to measure the quality of the automatically generated summaries, as well as the model summaries. Human subjects are presented with the summaries and asked to assess each summary based on the following criteria (each criterion has a five point scale with high scores indicating a better result in relation to that criterion) (Dang 2005, 2006):

Table 6.5: Readability five point scale evaluation results. In total the columns sum to 489.

Feature	strongly agree (5)	agree (4)	neither agree nor disagree (3)	disagree (2)	strongly disagree (1)
grammaticality	390	74	14	9	2
redundancy	464	15	9	1	0
clarity	459	21	4	2	3
focus	443	34	10	2	0
coherence	444	26	14	4	1

- **Grammaticality**: The description does not have formatting or capitalization errors or obviously ungrammatical sentences (e.g., fragments, missing components) that make the text difficult to read.
- **Redundancy**: There is no unnecessary repetition in the description. Unnecessary repetition might take the form of whole sentences that are repeated, or repeated facts, or the repeated use of a noun or noun phrase (e.g., “Westminster Abbey”) when a pronoun (“it”) would suffice.
- **Clarity**: It is easy to identify who or what the pronouns and noun phrases in the description are referring to. If a person or other entity is mentioned, it should be clear what their role in the story is. So, a reference would be unclear if an entity is referenced but its identity or relation to the story remains unclear.
- **Focus**: The description has a focus; sentences should only contain information that is related to the rest of the description.
- **Coherence**: The description is well-structured and well-organized. The description should not just be a heap of related information, but should build from sentence to sentence to a coherent body of information about a topic.

We followed the DUC approach to assess our model summaries. To assess all model summaries is a labour intensive work. Thus, we randomly selected around half of the model summaries (in total 489) and distributed them among three different human assessors who differed from the summary collectors. The humans were assigned different summary sets and were asked to assess the summaries according to the criteria described above. The results are shown in Table 6.5. In the table columns 2 to 6 show the total number of model summaries which obtained the label of the specific column.

Most of the model summaries (94% or more) obtained scores at level 5 and 4. These results show that our summaries are high quality model summaries and are applicable for automatic evaluation of systems producing image descriptions. The results are also comparable with the results reported by DUC about the readability assessment of model summaries. Dang (2005, 2006) reported that the model summaries perform 95% or above at level 5 and 4 (when taken together) in each of the criteria.

6.6 Summary

In this chapter we described the process of model summary collection for summarizer evaluation. We described an image set containing 307 images for which we collected model summaries. Each image pictures a geo-located entity and was collected from VirtualTourist, an online travel site. We showed that our image set covers 60 of the 107 entity types extracted from Wikipedia. For each image up to four model summaries were collected by human participants by extraction of relevant sentences from VritualTourist descriptions. We showed that their quality scores are comparable to those reported in DUC and TAC for their model summaries. Our model summaries are constructed from existing descriptions taken from VirtualTourist which is a different approach to that followed in DUC and TAC. In DUC and TAC model summaries are constructed from documents that are also used as input to the summarizer. We use the collected model summaries for intrinsic evaluation of our summarizer in different experimental settings described in the following chapters.

CHAPTER 7

Entity Type Models for Multi-Document Summarization[1]

So far we have shown that there exist a set of information types (attributes) humans associate with geo-located entities from urban and rural landscape. We have also shown that such attributes do occur in Wikipedia articles. Based on this we compiled corpora of Wikipedia articles from which we can extract models of how these entity type attributes are described in English sentences (entity type models). In this chapter we aim to use these entity type models to bias the sentence selection module of the summarizer to score sentences that conform to these models more highly than those which do not. This is done by including an additional feature, the entity type model feature (entityTypeModel) in the sentence scoring set of features of the summarizer. In addition, we also outline an idea of how entity type models can be used for addressing the challenges of summary composition: reduction of redundancy and sentence ordering within the summary.

We investigate three different methods for entity type modeling or deriving the entity type model feature from entity type corpora: *signature words*, *language modeling* and *dependency patterns*. These methods differ in how they represent the collected entity type corpora as a model. We evaluate each method based on its impact on automatic image description generation performance and report the results. We report the results of the automatic evaluation using the ROUGE metric and those of human readability assessment as outlined in Section 3.1.4. The models are derived from descriptions belonging to a single entity type corpus such as *church* but also from descriptions coming from groups of entity types such as *museum, opera house, gallery*.

[1] Some of the results presented in this chapter are published in Aker & Gaizauskas (2009, 2010*a*).

In this chapter we first describe the three entity type modeling strategies: signature words (Section 7.1), language models (Section 7.2) and dependency patterns (Section 7.3) and explain how they are applied to entity type corpora. Sections 7.4 and 7.5 describe how we use the entity type models in the summarization process. Section 7.6 outlines the training and testing procedures. The results of both automatic and manual evaluations are discussed in Section 7.7. Finally, in Section 7.8 the results for entity type models built from groups of entity types are presented and discussed.

7.1 Signature Words

Lin & Hovy (2000) introduce the notion of *signature words* for summarizing articles about news events. They define them as a family of related terms. These signature words are similar to the significant words defined by Luhn (1958) (see Chapter 2). However, instead of identifying them using manually set thresholds Lin & Hovy (2000) find them automatically. Like Luhn, the authors use signature words to represent the topic in the input documents. The topic words are selected from the input documents by comparing them to pre-classified texts on the same topic using the likelihood ratio λ (Dunning 1993), a statistical test to compute the likelihood of a word being a member of the set of relevant documents rather than the non-relevant ones. For each word in the input documents the authors compute the likelihood of the occurrence of that word in the pre-classified topic text collection. Another likelihood value is computed using the same word and another text collection that is out-of-topic. If the word has higher likelihood for the topic text collection than for the out-of-topic one, then the word is taken as a signature for the topic. Otherwise, the word is omitted from inclusion. They experimented with single signature words (uni-grams), two consecutive words (bi-grams) and three consecutive signature words (tri-grams) and report best summary results using bi-grams. In each case they used lemmas of the words. As topics the authors use *overcrowded prisons*, *cigarette consumption*, *computer security* and *solar power* and the corresponding articles from the TIPSTER-SUMMAC collection (Mani et al. 1999).

In the summarization process each sentence from the input documents of a specific topic is checked for whether or not it contains any word from the set of signature words of that topic. The score of the sentence is the sum of the weights of signature words it contains. Lin and Hovy integrated the signatures into the SUMMARIST (Hovy & Lin 1998) summarization system and compared the performance of signature words with two other features: sentence position[2] and

[2] In each document the first sentence gets the highest score and the last one the lowest.

tf*idf. The authors reported that signature words outperformed the other two features, the worst performing feature being tf*idf.

We use signature words as one method for entity type modeling. We derive signature words from the Wikipedia articles describing geo-located entities of the same entity type (see Chapter 5).

Application to Entity Type Corpora To derive a signature for each entity type corpus we use the following formula and generate signature words containing uni-gram and bi-gram signatures:

$$ngram = (entitytype, [(ngram_1, freq_1), .., (ngram_n, freq_n)]) \tag{7.1}$$

where *ngram* is either a single word (uni-gram) or two consecutive words (bi-gram). We do not use tri-grams as, according to Lin and Hovy, they were not a good choice for topic representation. As in Lin & Hovy (2000) lemmas of the words are used for both uni-gram and bi-gram models. We obtain the lemmas using the OpenNLP tools.[3]

We use the frequency information (*freq*) for each n-gram from the entity type corpus to score the sentences in the input documents (i.e. the web documents which are input to the summarizer for automatic summarization). We regard sentences from the input documents which contain frequent n-grams from the entity type corpus as more highly relevant for inclusion in the summary than sentences which contain less frequent n-grams. Therefore, when building the signature word models we take as the n-gram score the count of its n-gram lemma over the entire entity type corpus. We only consider open class words (nouns, verbs, adjectives and adverbs) as potential signature words. The identification of open class words is performed using the OpenNLP tools.

7.2 Language Models

Language models are used in different fields with different purposes. In information retrieval (IR), for instance, language models are used to retrieve documents relevant to a query. Song & Croft (1999), for example, use n-gram language models in a generative paradigm and first derive a distinct n-gram language model for each document. Based on this language model the probability of generating each term in the query is computed. The probability of generating the query is the product of probabilities of generating each of the terms occurring in the query.

[3] http://opennlp.sourceforge.net/

Finally, the documents are ranked in descending order based on the probability assigned to the query. Therefore, if terms of a document lead to higher generation probabilities, than this document is more relevant to the query.

Nenkova et al. (2006) investigate the impact of generative language models on multi-document summarization and compare such models to a non-generative approach. In their experiments the authors use DUC data for development and testing: they use the DUC 2003 input documents for generating their language model and test the impact of the model on the DUC 2004 data. The language model (M) contains single words with probabilities obtained through corpus statistics, $p(w_j) = \frac{C_{wj}}{N}$, where C_{wj} is the number of times the word w_j occurs in the corpus and *N* is the total count of words in the corpus. Nenkova et al. (2006) use the language model *M* to score each sentence *S* in the summarizer input documents based on two different approaches: accumulative and generative.

$$SumScore(S, M) = \sum_{w_j \in S} p_M(w_j) \tag{7.2}$$

$$AverageScore(S, M) = \frac{\sum\limits_{w_j \in S} p_M(w_j)}{|\{w_j | w_j \in S\}|} \tag{7.3}$$

$$MultiScore(S, M) = \prod_{w_j \in S} p_M(w_j) \tag{7.4}$$

In the accumulative scoring, the authors use the sum of word probabilities obtained from the model M to score each sentence of the input documents. This is done both with normalization over the total number of words in a sentence (Formula 7.3) and without such normalization (Formula 7.2). Instead of using probability values, the actual frequencies of the words could be used to compute these accumulative scores. The accumulative score computation (i.e. summation) is not affected by whether a frequency or a probability or another representation is used. However, this is not the case in a generative scenario, where the likelihood of a sentence being generated by a model M is computed, as given in Formula 7.4. According to Formula 7.4 short sentences are given higher likelihood than long ones regardless of their summary relevance. This is because the probability values are always between 0 and 1, so their product will be greater in case of shorter sentences than in case of longer ones because of the nature of multiplication with numbers from this interval: the more factors in the multiplication, the less is the product. The authors evaluate the quality of their summaries using ROUGE (Lin 2004). Compared to other summarization systems whose performances are also reported on the same DUC 2004 data, the summaries generated by Nenkova et al. (2006) through the different formula 7.2, 7.3 and 7.4 are ranked 4, 6 and 16 respectively. In total there are 20 different systems (including the ones of Nenkova et al. (2006)).

We use n-gram language models as a second method for representing entity type models. We use n-gram language models in generative way. However, we address the problem of the unfair bias of short sentences over the long ones and use the geometric mean of the computed probability score over the entire sentence (cf. Section 7.4.2).

Application to Entity Type Corpora As an alternative to signature words we also generated language models from the entity type corpora. As mentioned above our language models are entirely used in a generative way, i.e. we calculate the probability that a sentence is generated based on an n-gram language model. As for the signature word models we generate a uni-gram and a bi-gram model from each entity type corpus.

$$ngram = (entitytype, [(ngram_1, prob_1), .., (ngram_n, prob_n)]) \tag{7.5}$$

where again *ngram* is either the lemma of an uni-gram or bi-gram. $prob_i$ is the probability of an n-gram calculated using Good-Turing estimation (Jurafsky & Martin 2008):

$$prob(ngram) = \frac{(r+1)\frac{E(N_{r+1})}{E(N_r)}}{N} \tag{7.6}$$

where *r* is the number of times an n-gram is seen, N_r is the number of different n-grams seen exactly *r* times in the entire corpus, *E*(N_r) is the expected value of N_r and *N* is the number of words in the entire corpus. However, in case *r=0* (an n-gram is not seen) the probability is calculated as *E*(N_1)/*E*(N_0N). N_0 is the number of n-grams which have not been seen. It is calculated by taking the square of the number of all seen n-gram types minus their sum.

7.3 Dependency Patterns

Dependency patterns are concatenated terms extracted from dependency parse trees. Like signature words and language models, dependency patterns have been exploited in various language processing applications. In information extraction, for instance, dependency patterns have been used to fill manually constructed domain templates with information extracted from text resources (Yangarber et al. 2000, Sudo et al. 2001, Culotta & Sorensen 2004, Stevenson & Greenwood 2005, Bunescu & Mooney 2005, Stevenson & Greenwood 2009) but also to create these domain templates automatically (Sudo et al. 2003, Sekine 2006, Filatova et al. 2006, Etzioni et al. 2008, Banko & Etzioni 2008, Li et al. 2010).

However, dependency patterns have not been used extensively in summarization tasks. We are only aware of the work described in Nobata et al. (2002), who used dependency patterns in combination with other features to generate extracts in a single document summarization task. The authors use the DUC 2001 training set to derive their patterns. The set contains 30 topics each with 10 documents. For each topic their patterns are derived by first parsing the sentences in the topic documents for dependency analysis [4] and later extracting the most frequent dependency subtrees from them. In testing they parse each sentence in the same way they do for the training sentences, derive patterns from it and check whether these patterns occur in the set of pattern obtained from the training data. For each match they take the accumulated frequency information of the training patterns to score the sentence.

The authors do not report the performance of each feature separately on the quality of the summaries. However, they mention that when learning weights in a simple feature weighting scheme, the weight assigned to dependency patterns was lower than that assigned to other features. The small contribution of the dependency patterns may have been due to the small number of documents they used to derive their dependency patterns – as mentioned above they gathered dependency patterns from only ten domain specific documents which are unlikely to be sufficient to capture repeated features in a domain.

Application to Entity Type Corpora We use our entity type corpora to derive dependency patterns. Our patterns are derived from dependency trees which are obtained using the Stanford parser[5]. Each article in each entity type corpus was pre-processed by sentence splitting and named entity tagging[6]. Then each sentence was parsed by the Stanford dependency parser to obtain relational patterns. As with the chain model introduced by Sudo et al. (2001), our relational patterns are concentrated on the verbs in the sentences and contain *n+1* words (the verb and *n* words in direct or indirect relation with the verb). The number *n* is experimentally set to two words.

[4] Before parsing named entity tagging is performed.

[5] http://nlp.stanford.edu/software/lex-parser.shtml

[6] For performing shallow text analysis including named entity tagging the OpenNLP tools were used.

Table 7.1: Example sentence for dependency pattern.

Original sentence: The bridge was built in 1876 by W. W.
After NE tagging: The bridge was built in DATE by PERSON
Input to the parser: The entityType was built in DATE by PERSON.
Output of the parser: *det(entityType-2, The-1), nsubjpass(built-4, entityType-2), auxpass(built-4, was-3), prep-in(built-4, DATE-6), agent(built-4, PERSON-8)*
Patterns: The entityType built, entityType was built, entityType built DATE, entityType built PERSON, was built DATE, was built PERSON

Table 7.2: Five frequent patterns from the entity type corpora *river* and *volcano*.

river	volcano
location is entityType, is a tributary, length is km, is entityType flows, location is located	location is entityType, is entityType located, is active entityType, is complex entityType, is highest entityType

For illustration consider the sentence shown in Table 7.1 that is taken from an article in the *bridge* corpus. The first two rows of the table show the original sentence and its form after named entity tagging. The next step in processing is to replace any occurrence of a string denoting the entity type by the term “entityType” as shown in the third row of Table 7.1. The final two rows of the table show the output of the Stanford dependency parser and the relational patterns identified for this example.

To obtain the relational patterns from the parser output we first identified the verbs in the output. For each such verb we extracted two further words being in direct or indirect relation to the current verb. Two words are directly related if they occur in the same relational term. The verb *built-4*, for instance, is directly related to *DATE-6* because they both are in the same relational term *prep-in(built-4, DATE-6)*. Two words are indirectly related if they occur in two different terms but are linked by a word that occurs in those two terms. The verb *was-3* is, for instance, indirectly related to *entityType-2* because they are both in different terms but linked with *built-4* that occurs in both terms. It should be noted that we consider all direct and indirect relations while generating the patterns. The patterns generated for the example sentence are shown in the bottom of Table 7.1.

Following these steps we extracted relational patterns for each entity type corpus along with the frequency of occurrence of the pattern in the entire corpus. Table 7.2 shows five frequent patterns from the entity type corpora *river* and *volcano*.

7.4 Entity Type Model Features

In previous sections we described three different methods for creating entity type models from the entity type corpora. We will use these different models as an *entityTypeModel* feature, as mentioned in Section 3.2.1, to compute sentence scores in our summarizer. Depending on which entity type modeling method is used this feature will be named differently and its application in computing sentence scores will be different. In the following we describe how sentence scores are computed with each of the entity type model features.

7.4.1 Signature Words

We use the signature words to score each sentence in the input documents according to formula 7.7. In the formula the score of a sentence *S* is the sum of frequencies (*freq*) of n-grams from the signature word model *SigM* found also in the sentence *S*. We refer to this feature as *SigMSim*.[7]

$$SigMSim(S, SigM) = \sum_{ngram \in SigM \cap S} freq_{SigM}(ngram) \tag{7.7}$$

7.4.2 Language Models

The sentence score with language models is calculated according to formula 7.8.

$$LMSim(S, LM) = \sqrt[n]{\prod_{ngram \in S} prob_{LM}(ngram)} \tag{7.8}$$

In this case the score of a sentence *S* is the product of probabilities (*prob*) of its n-grams where the *prob* values are obtained from the language model *LM*. We refer to this feature as *LMSim*.[8] We take the geometric mean of the generative model shown in formula 7.4 (*n* is the number of n-grams constructed from the sentence *S*). This is in order to avoid the problem of favoring short sentences over the long ones by the generative model as discussed above (cf. Section 7.2).

[7] We use *SigMSim-1* to refer to uni-gram signature models and *SigMSim-2* to bi-gram ones.

[8] We use *LMSim-1* to refer to uni-gram language models and *LMSim-2* to bi-gram ones.

7.4.3 Dependency Patterns

The score with the dependency patterns is computed in a similar fashion to the *SigMSim* feature. We assign each sentence a dependency similarity score. To compute this score, we first parse the sentence on the fly with the Stanford parser and obtain the dependency patterns for the sentence. We then associate each dependency pattern of the sentence with the occurrence frequency of that pattern in the dependency pattern model (*DpM*). The dependency pattern feature (*DpMSim*) is then computed as given in formula 7.9. It is the sum of all occurrence frequencies of the dependency patterns in the *DpM* detected also in a sentence *S*.

$$DpMSim(S, DpM) = \sum_{p \in S} freq_{DpM}(p) \tag{7.9}$$

7.5 Dependency Patterns for Redundancy Reduction and Sentence Ordering

Apart from sentence scoring the dependency patterns can also be used to address two further challenges of multi-document summarization: the reduction of redundancy and sentence ordering. In this section, we outline and evaluate a possible way dependency patterns could be used for these tasks. As described below the approach we propose here requires manual preprocessing and categorization of dependency patterns, which makes it difficult to transfer the same idea to other domains. Therefore, in Chapter 8 we propose a fully automated approach to dealing with these challenges.

We can use the dependency pattern approach to address the problem of redundancy in the output summary in a novel way. Often, important information which must be included in the summary is repeated several times across the document set, but must be included in the summary only once. The common approach to avoiding redundancy is to use a text similarity measure to block the addition of a further sentence to the summary if it is too similar to one already included. Instead, since specific dependency patterns express specific types of information, we can group the patterns into groups expressing the same type of information and then, during sentence selection, ensure that sentences matching patterns from different groups are selected in order to guarantee broad, non-redundant coverage of information relevant for inclusion in the summary. This means that we may want to ensure that the summary contains a sentence describing the type of the entity, its location and some background information. For example, for the entity *Eiffel Tower* we may aim to say that it is a tower, located in Paris, designed by Gustave Eiffel, has a height of 324 meters, etc. To be able to do so, we categorize dependency patterns according to the type of information they express. For doing this we used the attributes discussed in Chapters 4 and 5:

- **entityType**: sentences containing the "entity type" information of the entity such as *Eiffel Tower* is a *tower*
- **location**: sentences containing information about where the entity is located
- **foundationyear**: sentences containing information about when the entity was built
- **specific**: sentences containing some specific information about the entity
- **surrounding**: sentences containing information about what other entities are close to the main entity
- **visiting**: sentences containing information about e.g. visiting times, etc.

The attributes *foundationyear, location, surrounding* and *visiting* are the same as the ones described in Chapter 4 and found in common for most of the entity types. The *entityType* attribute was found in all Wikipedia articles and is described in Chapter 5. Please note that this attribute also entails the commonly used *name* attribute in the Wikipedia articles, as it contains the mention of the name of the entity. The attribute *specific* represents a super group for all the remaining attributes we discussed in Chapters 4 and 5. We are interested in including, e.g., information about the height of the *Eiffel Tower* (attribute *height*) or its designer (attribute *designer*). These are all attributes "specific" to the entity type *tower* and might not appear for other entity types such as *volcano*. Thus instead of using all the specific attributes about an entity type we used the attribute *specific* to refer to these attributes while categorizing dependency patterns.

We manually assigned each dependency pattern in each corpus-derived model to one of the above attributes, provided it occurred five or more times in the entity type corpora. The patterns extracted for our example sentence shown in Table 7.1, for instance, are all categorized by *foundationyear* attribute because all of them contain information about the foundation date of an entity.

We make use of these attributes and apply the dependency patterns to categorize the sentences from the input documents to reduce the redundancy and order sentences within the summary. We refer to these summaries as *DepCat* summaries. Note that *DepCat* uses dependency patterns to categorize the sentences rather than rank them. It can be used independently from other features to categorize each sentence by one of the attributes described above. To do this, we obtain the relational patterns for the current sentence, check whether for each such pattern whether it is included in the *DpM*, and, if so, we add to the sentence the attribute the pattern was manually associated with.

For *DepCat* we proceed as follow. We first categorize the sentences into the six information types specified above. We sort the sentences in each category according to their sentence scores. The best scoring sentence goes to the top. Then we select from the categories (starting from top of the ranked list) sentences in the summary until the summary limit of 200 words is reached. We select the sentences from the categories in the order of: “entityType”, “location”, “foundationyear”, “specific”, “surrounding” and “visiting”. From each of the first three categories (“entityType” “location” and “foundationyear”) we take a single sentence to avoid redundancy. The same is applied to the final two categories (“surrounding” and “visiting”). Then, if length limit is not violated, we fill the summary with sentences from the “specific” category until the word limit of 200 words is reached.

7.6 Training and Testing procedure

For training and testing we use our model summary set described in Chapter 6. The set contains 307 geo-located entities. For each geo-located entity there are up to four model summaries that were created manually. Each summary contains a minimum of 190 and a maximum of 210 words. We divide this set of geo-located entities into training and testing sets. Both sets are described in the following subsections.

7.6.1 Training the Prediction Model

To obtain the feature weights (prediction model) for sentence scoring we use linear regression. Linear regression is a least square error method. It finds the values for the feature weights by predicting the actual sentence scores using the values of the sentence features. Because of this it requires some training data consisting of assessed sentences where each sentence has a final score and values for the features.

We use sentences of 202 geo-located entities from the 307 set for composing the training data. For each training entity we gather all descriptions associated with it from *VirtualTourist*. We compute for each sentence in each description a ROUGE score by comparing the sentence to those included in the model summaries for that particular entity and retain the highest score. We also run the feature extraction portion of the summarizer on each sentence in order to compute feature scores for the sentences. Finally, we include each sentence, its feature representation and its ROUGE score in a training file.

As objective metrics to maximize we use ROUGE 1 (R1), ROUGE 2 (R2) and ROUGE SU4 (RSU4), leading to three training input files differing only in the ROUGE metric scores. R1 and R2 compute the number of uni-gram and bi-gram overlaps, respectively, between the automatic and model summaries. RSU4 allows bi-grams to be composed of non-contiguous words, with a maximum of four words between the bi-grams. We used ROUGE as a metric to maximize because we also use it to automatically evaluate our output summaries.

7.6.2 Testing

For testing purposes we use 105 geo-located entities from our set of 307 entities. For each geo-located entity we use a set of web-documents as input (see Section 3.2.2) and generate a summary from these documents using our summarizer.

We generate the summaries using the summarization features described in Section 3.2.1 as well as the entity type model features described in Section 7.4. The features are used to score sentences in the input documents.

After the sentence scoring process, the summarizer selects sentences for summary generation. The summary is constructed by first selecting the sentence that has the highest score, followed by the next sentence with the second highest score until the compression rate is reached. As in Saggion & Gaizauskas (2004), Saggion (2005) (see also Section 2.2.1 for more details), before a sentence is selected a similarity metric for redundancy detection is applied to each sentence to decide whether a sentence is distinct enough from already selected sentences to be included in the summary or not. The summarizer first eliminates closed class words (prepositions, articles) from the sentences and then measures lemma overlap with the lemmas of the remaining open class words (nouns, verbs, adjectives and adverbs), which are, according to Ye et al. (2005), a strong basis for measuring similarities between sentences. We refer to this method as *greedySelection*. Note that we do not use *greedySelection* when the *DepCat* feature is used.

7.7 Evaluation

To evaluate our approach we used two different assessment methods: ROUGE (Lin 2004) and a manual readability assessment.

Table 7.3: Model, Wikipedia baseline and isStarter+LMSim-2+DepCat summary for Eiffel Tower.

Model Summary	Wikipedia baseline summary	isStarter+LMSim-2+DepCat summary
The Eiffel Tower is the most famous place in Paris. It is made of 15,000 pieces fitted together by 2,500,000 rivets. It's of 324 m (1070 ft) high structure and weighs about 7,000 tones. This world famous landmark was built in 1889 and was named after its designer, engineer Gustave Alexandre Eiffel. It is now one of the world's biggest tourist places which is visited by around 6,5 million people yearly. There are three levels to visit: Stages 1 and 2 which can be reached by either taking the steps (680 stairs) or the lift, which also has a restaurant "Altitude 95" and a Souvenir shop on the first floor. The second floor also has a restaurant "Jules Verne". Stage 3, which is at the top of the tower can only be reached by using the lift. But there were times in the history when Tour Eiffel was not at all popular, when the Parisians thought it looked ugly and wanted to pull it down. The Eiffel Tower can be reached by using the Mtro through Trocadro, Ecole Militaire, or Bir-Hakeim stops. The address is: Champ de Mars-Tour Eiffel.	The Eiffel Tower (French: Tour Eiffel, [tur efel]) is a 19th century iron lattice tower located on the Champ de Mars in Paris that has become both a global icon of France and one of the most recognizable structures in the world. The Eiffel Tower, which is the tallest building in Paris, is the single most visited paid monument in the world; millions of people ascend it every year. Named after its designer, engineer Gustave Eiffel, the tower was built as the entrance arch for the 1889 World's Fair. The tower stands at 324 m (1,063 ft) tall, about the same height as an 81-story building. It was the tallest structure in the world from its completion until 1930, when it was eclipsed by the Chrysler Building in New York City. Not including broadcast antennas, it is the second-tallest structure in France, behind the Millau Viaduct, completed in 2004. The tower has three levels for visitors. Tickets can be purchased to ascend either on stairs or lifts to the first and second levels.	The Eiffel Tower, which is the tallest building in Paris, is the single most visited paid monument in the world; millions of people ascend it every year. (**entityType**) The tower is located on the Left Bank of the Seine River, at the northwestern extreme of the Parc du Champ de Mars, a park in front of the Ecole Militaire that used to be a military parade ground. (**location**) The tower was met with much criticism from the public when it was built, with many calling it an eyesore. (**foundationyear**) Counting from the ground, there are 347 steps to the first level, 674 steps to the second level, and 1,710 steps to the small platform on the top of the tower. Although it was the world's tallest structure when completed in 1889, the Eiffel Tower has since lost its standing both as the tallest lattice tower and as the tallest structure in France. The tower has two restaurants: Altitude 95, on the first floor 311ft (95m) above sea level; and the Jules Verne, an expensive gastronomical restaurant on the second floor, with a private lift. (**specific**) There is an entrance fee of between euro;4.10 and euro;10.70 for adults and between euro;2.30 and euro;5.90 for children, depending on which floor you wish to visit by elevator. (**visiting**)

Table 7.4: ROUGE scores for each single feature and Wikipedia baseline. The numbers 1 and 2 after the model features *SigMSim* and *LMSim* indicate the use of uni-gram (1) or bi-gram (2) version of those models.

Recall	cenSim	senPoS	qSim	isStarter	SigMSim-1	SigMSim-2	LMSim-1	LMSim-2	DpMSim	Wiki
R2	.0734	.066	.0774	.0869	.08	.079	.079	.0895	**.093**	.097
RSU4	.12	.11	.12	.137	.133	.133	.135	.142	**.145**	.14

7.7.1 ROUGE Assessment

In the first assessment we compared the automatically generated summaries against model summaries written by humans using ROUGE (Lin 2004). Following the Document Understanding Conference (DUC) evaluation standards we used ROUGE 2 (*R2*) and ROUGE SU4 (*RSU4*) as evaluation metrics (Dang 2006).

As baselines for evaluation we used summaries extracted from the *top document* retrieved from the web and *Wikipedia* (see Section 3.2.3).

First, we compared the baseline summaries against the VirtualTourist model summaries. Wikipedia baseline ROUGE scores (R2 .097***, RSU4 .14***) are significantly higher than the first or top document ones (R2 .042, RSU4 .079).[9]

Secondly, we separately ran the summarizer over the input web-documents for each single feature and compared the automated summaries against the model ones. The results of this comparison are shown in Table 7.4.

From the table we see that automated summaries using each of the features achieve lower ROUGE scores than the Wikipedia baseline, thus indicating that initial sentences from Wikipedia articles are indeed of high quality. The opposite is true for the summaries obtained from the first top web document: the automated summaries using any of our summarization features score higher than the first document baseline ones (R2 .042, RSU4 .079, not shown in the table). For this reason, we will focus on the Wikipedia baseline summaries to draw conclusions about the quality of our automatic summaries. Table 7.3 shows the Wikipedia baseline summary for the *Eiffel Tower*.

Turning to the ROUGE results for single summarization features in Table 7.4, we can see that the dependency model feature (*DpMSim*) contributes most to the summary quality according to the two ROUGE metrics. It achieves significantly higher ROUGE scores than all other features (***), except the *LMSim-2* feature, where it leads to a small improvement. Compared to the Wikipedia baseline (*Wiki*) the *DpMSim* summaries achieve insignificantly different ROUGE scores.

The lowest ROUGE scores are obtained if only sentence position (*senPos*) is used. These scores are significantly lower than those of the Wikipedia baseline, which is also true for all other features except *LMSim-2* and *DpMSim*.

To see how the ROUGE scores change when features are combined with each other we used different combinations of the features, ran the summarizer for each combination and compared the automated summaries against the model ones.[10] Among the different combinations we also

[9] To assess the statistical significance of ROUGE score differences between multiple summarization results we performed a pairwise Wilcoxon signed-rank test. We use the following conventions for indicating significance level: *** = $p < .0001$, ** = $p < .001$, * = $p < .05$ and no star indicates non-significance.

[10] For each feature combination a different set of weights are trained using linear regression as described in Section 7.6. Although we used R1, R2 and RSU4 as metrics when training the weights we obtained the best results when R2 was used. Thus the results we report for each feature combination are based on R2 trained feature weights.

Table 7.5: ROUGE scores of feature combinations which score moderately or significantly higher than dependency pattern model (*DpMSim*) feature and Wikipedia baseline.

Recall	isStarter + LMSim-2	isStarter + LMSim-2 + DepCat***	DpMSim	Wiki	User−To−User
R2	.095	**.102**	.093	.097	0.11
RSU4	.145	**.155**	.145	.14	0.16

included the dependency pattern categorization (*DepCat*) feature.[11] Table 7.5 shows the results of feature combinations which score moderately or significantly higher than the dependency pattern model (*DpMSim*) feature score shown in Table 7.4. In Table 7.5 we also give ROUGE scores of model summaries compared to each other (column *User-To-User*) which represent the upper bound scores one could achieve with automatic summaries.

The results show that combining *DpMSim* with other features did not lead to higher ROUGE scores than those produced by that feature alone. In contrast to this, the feature *LMSim-2*, which on its own has a performance insignificantly different from *DpMSim* (Table 7.4), combines well with other features. In combination with the *isStarter* feature (see Section 3.2.1), it achieves ROUGE scores comparable to *DpMSim*. The best results, however, are achieved if categorization using dependency patters (*DepCat*) is added to this combination (*isStarter+LMSim-2+DepCat*). Such summaries categorized by dependency patterns achieve significantly higher ROUGE scores than the Wikipedia baseline[12] and also score very close to the *User-to-User* upper bound. Table 7.3 shows a summary about the *Eiffel Tower* obtained using this *isStarter+LMSim-2+DepCat* feature.

7.7.2 Readability Assessment

We also evaluated our summaries using a readability assessment as in DUC and TAC. DUC and TAC manually assess the quality of automatically generated summaries by asking human subjects to score each summary using five criteria – *grammaticality, redundancy, clarity, focus* and *structure*. Each criterion is scored on a five point scale with high scores indicating a better result (Dang 2005).

For this evaluation we used the same 105 entities as in the ROUGE evaluation. As the ROUGE evaluation showed that the dependency pattern categorization (*DepCat*) renders the best results

[11] *DepCat* is used to re-order the sentences scored by other features. It is not included in Formula 3.1 (see Section 3.2.1) to obtain a feature combination. Also when *DepCat* is used we switch off the *greedySelection*.

[12] For both ROUGE R2 and ROUGE SU4 the significance is at level $p < .0001$.

Table 7.6: Readability evaluation results: – Wikipedia baseline (W), isStarter + LMSim-2 (SLM) and isStarter + LMSim-2 + DepCat (SLMD)

	5			4			3			2			1		
Criterion	W	SLM	SLMD	W	SLM	SLMD	W	SLM	SLMD	W	SLM	SLMD	W	SLM	SLMD
clarity	72.6	50.5	53.6	21.7	30.0	31.4	1.2	6.7	5.7	4.0	10.2	6.0	0.5	2.6	3.3
coherence	67.1	39.0	48.3	23.6	31.4	26.9	4.8	12.4	11.9	3.3	10.2	9.8	1.2	6.9	3.1
focus	72.1	49.3	51.2	20.5	26.0	25.2	3.8	10.0	10.7	3.3	10.0	10.5	0.2	4.8	2.4
grammar	48.6	55.7	62.9	32.9	29.0	30.0	5.0	3.1	1.9	11.7	12.1	5.2	1.9	0	0
redundancy	69.8	42.9	55.0	21.7	17.4	28.8	2.4	4.5	4.3	5.0	27.1	8.8	1.2	8.1	3.1

Table 7.7: Readability evaluation results showing only the percentage values of summaries which achieved scores at levels four or above.

Criterion	W	SLM	SLMD
clarity	94.3	80.5	85
coherence	90.7	70.4	74
focus	92.6	75.3	76.4
grammar	81.6	84.7	92
redundancy	91.5	60.3	83

when used in feature combination *isStarter + LMSim-2 + DepCat*, we also performed the readability assessment on summaries generated using this feature combination. For comparison we also evaluated summaries which were not structured by dependency patterns (*isStarter + LMSim-2*) and the Wikipedia baseline summaries.

We asked four people to assess the summaries. Each person was shown all 315 summaries (105 from each summary type) in a random way and was asked to assess them according to the DUC and TAC manual assessment scheme (for the scheme see Section 6.5). The results are shown in Table 7.6. In the table each cell shows the percentage of summaries scoring the ranking score heading the column for each criterion in the row, as produced by the summary method indicated by the subcolumn heading. The numbers indicate the percentage values averaged over the four assessors.

We see from Table 7.6 that using dependency patterns to categorize the sentences and produce a structured summary helps to obtain more readable summaries. Looking at the 5 and 4 scores in Table 7.7 we see that the dependency pattern categorized summaries (*SLMD*) have better clarity (85% of the summaries), are more coherent (74% of the summaries), contain less redundant information (83% of the summaries) and have better grammar (92% of the summaries) than the ones without dependency categorization (80%, 70%, 60%, 84%). The big difference in redundancy scores (83% versus 60%) shows in particular that the *DepCat* feature is a useful feature for redundancy reduction in summaries.

The scores of our automated summaries were better than the Wikipedia baseline summaries in the *grammar* feature. We included the *grammar* feature in the evaluation to be consistent with the evaluation criteria used in DUC and TAC. The low grammar score in Wikipedia summaries is due to non-standard characters used to describe how an entity is pronounced in other languages. Some of these non-standard characters were not properly displayed in the manual evaluation tool and therefore the Wikipedia summaries were assigned by the human assessors lower grammar scores. Summaries that did not have these problems obtained higher grammar scores. In all other features the Wikipedia baseline summaries obtained better scores than our automated summaries. This comparison shows that there is still a gap to fill in order to obtain more readable summaries.

7.7.3 Discussion

In our single feature analysis the results indicate that the entity type model features indeed help the summarizer to produce better summaries. Using any of our entity type model features we have obtained higher ROUGE scores than when standard summarization features *cenSim*, *senPos* and *qSim* were used to produce the summaries. However, not all methods for entity type modeling have shown equal performance, suggesting that the way entity type models are represented plays a role in how useful they are as summarization features. In our case, summaries obtained through the standard feature *isStarter* are better than those generated by signature word (*SigMSim*) and unigram language models (*LMSim-1*). As described in Section 3.2.1, the *isStarter* feature looks in each sentence only for an occurrence of the given query (entity name) and entity type. We believe that sentences starting with the query or entity type are likely to be salient for the given entity name which therefore leads to better scoring summaries. Bigram language models (*LMSim-2*) and dependency patterns (*DpMSim*) on the other hand significantly outperformed the *isStarter* feature, *DpMSim* being the single feature which lead to the highest scoring summaries, almost identical to the Wikipedia baseline.

From this we can conclude that the summaries obtained using signature word and language models are not as good as the ones obtained using dependency patterns. The main weakness of signature words and n-gram language models is that they only capture very local information about short term sequences and cannot model long distance dependencies between terms. For example one common and important feature of entity descriptions is the simple specification of the entity type, e.g. the information that the entity *London Bridge* is a *bridge* or that the *Rhine* is a *river*. If this information is expressed as in the first line of table 7.8, signature words and n-gram language models are likely to reflect it, since one would expect the tri-gram *is a bridge*

Table 7.8: Example of sentences which express the type of an entity.

London Bridge is a **bridge**...
The **Rhine** (German: Rhein; Dutch: Rijn; French: Rhin; Romansh: Rain; Italian: Reno; Latin: Rhenus West Frisian Ryn) is one of the longest and most important **rivers** in Europe...

to occur with high frequency in a corpus of bridge descriptions. However, if the type predication occurs with less commonly seen local context, as is the case for the entity *Rhine* in the second row of Table 7.8 – *one of the longest and most important rivers* – signature words and n-gram language models may well be unable to identify it.

Intuitively, what is important in both these cases is that there is a predication whose subject is the entity instance of interest, and the head of whose complement is the entity type: *London Bridge ... is ... bridge* and *Rhine ... is ... river*. Sentences matching such patterns are likely to be important ones to include in a summary. The results suggests that rather than representing entity type models via corpus-derived signature words or language models, it is better to represent them using corpus-derived dependency patterns instead.

The investigation of feature combinations has also showed that using dependency patterns for redundancy reduction and sentences ordering within a summary (feature *DepCat*) significantly improves the quality of summaries. Interestingly, when dependency patterns are used for sentence scoring (*DpMSim*) no further improvement could be observed in additionally using dependency patterns for redundancy reduction and sentence ordering (*DepCat*). However, *DepCat* significantly improved the ROUGE scores of the summaries generated by the combination of the bigram language models (*LMSim-2*) and the *isStarter* feature (*isStarter* + *LMSim-2* + *DepCat*). This combination of features produced structured summaries which led to significantly better results than Wikipedia baseline summaries and almost equal to human generated baseline summaries as assessed by ROUGE. Human readability assessment reflected these ROUGE scores for the grammaticality aspect of the summaries. However, the automated *isStarter* + *LMSim-2* + *DepCat* summaries scored lower in fluency and redundancy than Wikipedia baseline, indicating that usage of *DepCat* for these purposes still has scope for improvement.

From these results we can conclude that it is possible to generate higher quality geo-located entity descriptions using automatic summarization techniques than simply referring to the existing descriptions in Wikipedia, which justifies using automatic summarization for image description generation generally, not only in cases where no Wikipedia descriptions for a given entity exist. Since use of entity type models represented as dependency patterns was crucial for achieving this result, we conclude that dependency patterns are worth investigating for entity focused automated summarization tasks. Such investigations should in particular concentrate on how dependency patterns can be used to order sentence within the summary, as our best results were achieved when dependency patterns were used for this purpose. In particular, replacing manual categorization of dependency patterns which was necessary for this purpose with an automatic procedure needs to be addressed.

7.8 Grouping of Geo-Located Entity Types

In Chapter 4 we showed that if entity types have similar look and purpose people tend to agree on what information to associate with them. The question now arises as to whether it is possible to derive entity type models for grouped types, rather than for single types, such that these models still improve the performance of summary generation for a single geo-located entity. This would be very useful when there is a geo-located entity for which no or not enough textual resources are available. In this case text resources of similar entity types could be used to derive an entity type model for that type. For example, we showed in our experiment in Chapter 4 that entity types *church, basilica, abbey, cathedral* and *temple* correlate highly with each other (see Section 4.3.2). Some of these entity types, like *church*, have more frequently occurring instances, than others (e.g. *basilica*), i.e. there are typically more churches than basilicas, and therefore it can be expected that there are more church descriptions to build entity type corpora from than there are basilica descriptions. If a summary for a basilica needs to be generated, but little or no information exists on this entity, then texts describing churches and other religious geo-located buildings could be used to derive entity type models, and these models can be used to generate a description of the basilica in question. We therefore investigated whether deriving entity type models from grouped entity type corpora has any effect on the summary results.

In total our geo-located entity collection covers 60 entity types (see Chapter 6). As discussed in Section 4.3.3 machine learning techniques could be applied to perform hierarchical grouping between them. However, for simplicity we perform manual grouping based on the look and purpose of the entity types. The resulting set of groups of similar entity types is shown in Table 7.9.

Using these groups of entity types we derive entity type models. We investigate only the bigram language model (*LMSim-2*) and the dependency model (*DpMSim*) because they were the best performing features in the previous experiment (see Section 7.7). With this we aim to

Table 7.9: Groups of entity types.

Group name	**Entity types within the group**
religious places	church, cathedral, chapel, basilica, synagogue, abbey, shrine, mosque, temple
mountainous areas	mountain, peak, volcano, ski resort, glacier, hill
buildings	tower, skyscraper, house, building, residence, palace, castle, hotel, parliament
water bodies	canal, lake, river, waterfall
cultural attractions	museum, opera house, gallery
roads	road, avenue, boulevard
streets	street, square
transport sites	railway, railway station
sea sides	beach, coast, bay
populated areas	district, village, city
education	college, university
shopping areas	market, shopping centre, shop, store
monuments	monument, statue
places of entertainment	restaurant, casino, bar, cinema, pub, club
civil engineering	bridge, gate, arch
places for relaxation	park, garden
places for sport	stadium, arena
animal theme parks	zoo, aquarium

Table 7.10: ROUGE scores of features *LMSim−2*, *LMSim−2g*, *DpMSim* and *DpMSim−g* (g indicates features which are derived from groups of entity type corpora).

Recall	LMSim−2	LMSim−2g	DpMSim	DpMSim−g
R2	.089	.087	.093	.092
RSU4	.142	.14	.145	.144

investigate whether deriving these two models from grouped entity type corpora has any effect on the summary results. The results of the ROUGE evaluation are shown in Table 7.10.

From Table 7.10 we can see that compared to single entity type models there is a small decrease in both ROUGE 2 and ROUGE SU4 scores when group of entity types are used to derive the models. These non-significant changes on the scores show that in general grouping of similar entity types can be performed without losing too much in summary quality. Therefore, if there is an entity type for which no or not enough textual resources are available, text resources of similar entity types could be used to build an entity type model for that type. However, when text resources exist for every single entity type, as is the case in our entity type corpus, the results indicate that deriving single entity type models instead of group models and using these in generating image descriptions lead to better ROUGE results.

7.9 Summary

In this chapter we have investigated three different methods to derive entity type models from entity type corpora: signature words, language models and dependency patterns. We discussed the use of these methods within the summarizer to bias sentence selection. We showed that dependency pattern models yield summaries which score more highly than the summaries obtained using signature word or language models which use a simpler representation of an entity type model. Dependency pattern models can contribute both to better sentence scoring and readability in particular clarity and coherence scores. From this we conclude that entity type models as represented by dependency patterns do lead to improved results in entity focused automatic text summarization. The downside of the approach proposed here for redundancy and sentence ordering within the summary was that it relied on manual categorization of dependency patterns. We aim to address this in the next chapter, where a fully automatic procedure for incorporating dependency patterns into summary building is introduced. Finally, we also showed that deriving entity type models from groups of similar entity types is possible, which is useful in cases in which there exist limited text resources for single entity types. For such entity types entity type models of similar entity types can be used instead without loosing too much in summary quality.

CHAPTER 8

Addressing the Challenges of Summary Composition[1]

In the previous chapter we demonstrated that entity type modeling using dependency patterns helps to improve the performance of a multi-document summarizer on the entity focused summarization task. Dependency entity type models were used both to score the sentences and in summary composition, to address redundancy reduction and sentence ordering within the output summary. Our proposed method to integrate dependency pattern models into the summarizer for redundancy reduction and sentence ordering involved manual categorization of the patterns. This is a limiting factor, which makes the idea of entity type modeling difficult to port to other domains. If descriptions for entities from a different domain are to be generated, manual categorization of dependency patterns for this domain has to be performed which is a laborious and expensive task. To address this problem, in this chapter we present a fully automatic framework, which permits the challenges of summary composition to be addressed in a fully automatic way, without manual intervention.

Finding the subset of the sentences to include in the output summary, so that the information content is maximized, the redundancy minimized and the sentence order optimized can be regarded as a search problem. The ideal solution would be for the search algorithm to address all these three challenges at the same time and output the desired summary. However, previous work has investigated various ways to solve the search problem by dealing with these challenges separately – mostly the informativeness and redundancy problems are addressed, while the sentence ordering within the summary is not considered. In addition, related work uses an existing prediction model to guide the search, assuming that the model can distinguish between good

[1] Some of the results presented in this chapter are published in Aker et al. (2010), Aker, Cohn & Gaizauskas (2012).

and bad summaries. However, this is problematic because the model is not trained to optimize summary quality but some other peripheral objective. What is desired is a prediction model that learns the system parameters to best describe a training set consisting of pairs of documents and reference summaries.

We show in this chapter that the search problem can be solved optimally and efficiently using A* search (Russell et al. 1995) with all summarization challenges integrated. Furthermore, we run our A* search algorithm with the training of the prediction model intact. Our training algorithm learns system parameters such that the best scoring *whole summary* under the prediction model has a high score under an evaluation metric (i.e. ROUGE). To learn the training model we use discriminative training.

In this chapter we first explain the multi-document summarization task as a search problem (Section 8.1). We then outline our framework for the informativeness maximization (Section 8.2), the redundancy reduction (Section 8.3), and sentence ordering within the output summary (Section 8.4). In Section 8.5 we describe the discriminative training approach we take within this framework. The evaluation results are given in Section 8.6.

8.1 Summary Composition as Search Problem

We formulate the summary composition task as a search problem. Amongst a set of sentences, we search for a subset of sentences such that the information content is maximized, the redundancy minimized, and the best sentence order is achieved. A summarization model is used to score summaries. Summaries are ranked according to these scores, so that in search, the summary with the highest score can be selected. We use the following summarization model s to score a summary:

$$s(\mathbf{y}|\mathbf{x}) = \sum_{i \in \mathbf{y}} \phi(x_i)\lambda \tag{8.1}$$

where $\mathbf{x}$ is the document set, composed of k sentences, $\mathbf{y} \subseteq \{1 \ldots k\}$ is the set of sentence indexes selected for the summary, $\phi(\cdot)$ is a feature function that returns a set of features values for each candidate summary, and λ is the weight vector associated with the set of features. We use standard summarization features described in Section 3.2.1 but also entity type models such as n-gram language models and relational dependency patterns (see Chapter 7). The weights are learned by the discriminative training approach described below in Section 8.5.

In the search process the summarization model is used to find the summary $\hat{\mathbf{y}}$ with maximal summary score:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} s(\mathbf{y}|\mathbf{x}) \tag{8.2}$$

8.2 Searching for the Most Informative Summary

The goal of the search for the most informative summary is to find a subset of sentences from the entire set of scored sentences to form a summary which is most related to a given query. The search is constrained so that the subset of sentences does not exceed the summary length threshold. While searching, a graph is constructed whose nodes are search states and whose edges represent sentences which get added to the summary if the edge is traversed (see Figure 8.1). Each node has associated information about the summary length, summary score, and a heuristic function score. The search starts with an empty summary (start state, length 0, a summary and heuristic score of 0), and follows one of the outgoing arcs to expand it. A new state is created when a new sentence is added to the summary. The new state's length is updated with the number of words of the new sentence. The score of the state is computed under the summarization model described in the previous section. The heuristic function score is computed given an admissible heuristic, also described below. A goal state is any state or summary where it is not possible to add another sentence without exceeding the summary length threshold. The summarization problem is then equivalent to finding the best scoring path (sum over the sentence scores on this path) between the start state and a goal state.

We use the A* search algorithm Russell et al. (1995) to efficiently traverse the search graph and accurately find the best scoring path. In A* search a best-first strategy is applied to traverse the graph from a starting state to a goal state. The search procedure requires a scoring function for each state, here $s(\mathbf{y}|\mathbf{x})$ from Equation 8.1, and a heuristic function which estimates the additional score to get from a given state to a goal state. The search algorithm is guaranteed to converge to the optimal solution if the heuristic function is *admissible*, that is, if the function used to estimate the cost from the current node to the goal never overestimates the actual cost.

Our A* search implementation is shown in Algorithm 1. Given a set of sentences to summarize, a summary scoring function and a heuristic function, the algorithm aims to find the best scoring summary. The search starts with an empty search space and incrementally builds a search graph

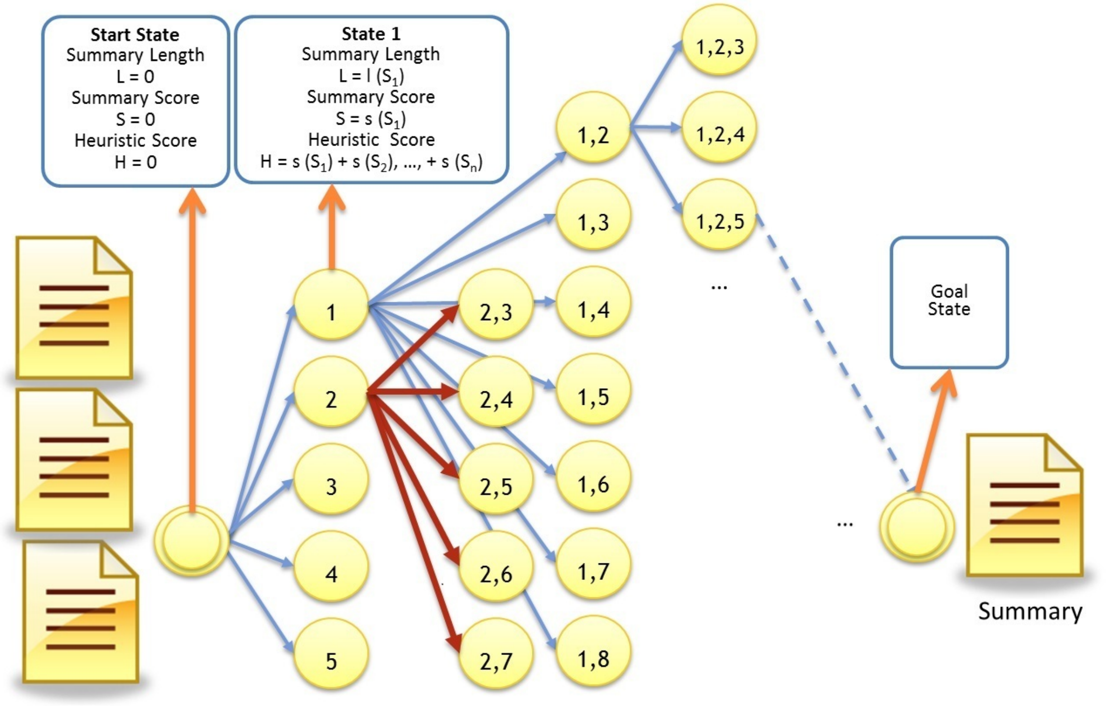


Figure 8.1: Search graph for extractive summarization.

by visiting the sentences in the order they are sorted.[2] Every new state in the search graph is stored in a priority queue which is sorted by the sum of the state's sentence scores and its heuristic score. The best state is popped off the queue to expand first (line 4). The resulting new states from every best state are stored in the queue (lines 8–15). A final state is marked using a flag (the last entry in the tuple in lines 2, 8 and 13). If a T state is popped off the queue (line 4), it is a summary that is better than all the summaries still in the queue. The reason for this is that T states are popped off the queue because of their actual summary score excluding the heuristic score. The function $\text{length}(\mathbf{y}, \mathbf{x}) = \sum_{n \in \mathbf{y}} \text{length}(x_n)$ returns the length of sentences specified.

The function $h(\mathbf{y}; \mathbf{x}, L)$ in line 12 of Algorithm 1 is the heuristic function shown in Algorithm 2.[3] It provides an upper bound on the additional score achievable in reaching a goal state from state $\mathbf{y}$ (current summary), which makes the heuristic admissible. In the algorithm, the shorthand $s_n = \phi(x_n) \cdot \lambda$ for sentence n's score, $l_n = \text{length}(x_n)$ for its length and $l_{\mathbf{y}} = \sum_{n \in \mathbf{y}} l_n$ for the total length of the current state (unfinished summary) is used. Within the heuristic function

[2] Sorting is based on $\frac{score}{length}$.

[3] We have experimented with different heuristics and use here the best performing one. We include in Appendix A the different heuristics we have investigated.

Algorithm 1 A* search for extractive summarization.

```
Require: set of sentences, $\mathbf{x} = x_1, \ldots, x_k$
Require: scoring function $s(\cdot)$
Require: heuristic function $h(\cdot)$
Require: summary length limit $L$
 1: $v = 0$, summary score
 2: schedule = $[(0, \emptyset, \mathrm{F})]$                    {priority queue of triples}
                                          {(A* score, sentence indices, done flag)}
 3: while schedule $\neq$ [] do
 4:   $v, \mathbf{y}, f \leftarrow$ pop(schedule)
 5:   if f = T then
 6:     return $\mathbf{y}$                                                {success}
 7:   else
 8:     push(schedule, $(s(\mathbf{y}|\mathbf{x}), \mathbf{y}, \mathrm{T})$)
 9:     for $y \in [(\max(\mathbf{y}) + 1) \cdot \cdot k]$ do
10:       $\mathbf{y}' \leftarrow \mathbf{y} \cup y$
11:       if length($\mathbf{y}', \mathbf{x}$) $\leq L$ then
12:         $v' \leftarrow s(\mathbf{y}''|\mathbf{x}) + h(\mathbf{y}'; \mathbf{x}, L)$
13:         push(schedule, $(v', \mathbf{y}', \mathrm{F})$)
14:       end if
15:     end for
16:   end if
17: end while
```

we use the entire score of a sentence when it fits to the summary (lines 7 to 9). In case the next sentence in the sentence list is too long to fit within the current summary, the algorithm then skips sentences until it finds the best scoring sentence that does fit (lines 11 to 17).

8.3 Searching for the Least Redundant Summary

To address redundancy within a summary we investigate two different approaches. First, we extend the set of features which are used to score a summary with an extra redundancy feature (Section 8.3.1). In the second approach, the A* search as described in Section 8.2 is modified to deal with redundancy.

Algorithm 2 Agg. + final heuristic, $h(\mathbf{y};\mathbf{x},L)$

```
Require: $\mathbf{x}$ sorted in order of score/length
$v \leftarrow 0$, summary score
$l' \leftarrow l_{\mathbf{y}}$
for $n \in [\max(\mathbf{y})+1, k]$ do
    if $s_n \leq 0$ then
        return v
    end if
    if $l' + l_n \leq L$ then
        $l' \leftarrow l' + l_n$
        $v \leftarrow v + s_n$
    else
        for $m \in [(n+1) \cdot \cdot k]$ do
            if $m \leq k \ \wedge \ s_m > 0$ then
                if $l_{\mathbf{y}} + l_m \leq L$ then
                    return $v + s_m \frac{L - l_{\mathbf{y}'}}{l_m}$
                end if
            end if
        end for
    end if
end for
return $v$
```

8.3.1 Dynamic Filter to Reduce Redundancy

The usual practice in sentence scoring is to compute sentence scores based on the combination of feature values, treating each sentence in the input documents separately from the others. However, in the case of multi-document summarization, where sentences from separate documents may well express the same information, this relatedness also needs to be captured and accounted for when calculating the summary-worthiness of the sentences. That is, if a sentence is redundant in relation to the other sentences in the input documents, its score should reflect this fact.

Therefore we introduce an additional feature to capture the redundancy phenomenon in the sentence scores. The prediction model (see Section 8.5) should learn to use this feature in a way that reduces redundancy within a summary.

We refer to this feature as SF – the value of the feature should decide which sentences to favour for the inclusion in the summary and which to exclude. The value of SF for sentence x_i is computed as a sum of the redundancy scores of all sentence pairs (x_i, x_j) with $j \in \{\mathbf{1} \ldots \mathbf{k}\} \setminus \{\mathbf{i}\}$ and k the number of sentences in the input documents:

$$SF(x_i) = \sum_{j=1, i \neq j}^{k} sim(x_i, x_j) \tag{8.3}$$

The function $sim(.,.)$ is given in Equation 8.4. The function is a Jaccard function and is used to compute the redundancy score between two sentences x_i and x_j.

$$sim(x_i, x_j) = \frac{1}{n} \sum_{l=1}^{n} \frac{|ngrams(x_i, l) \bigcap ngrams(x_j, l)|}{|ngrams(x_j, l)|} \tag{8.4}$$

where ngrams(x_i, n) is the set of n-grams in sentence x_i and ngrams(x_j, n) in sentence x_j. This method returns 0 if x_i and x_j do not share any n-grams. When all n-grams of x_j are found in the list of n-grams of x_i the method returns 1. Note that we use this function only to see how many n-grams of x_j are found in x_i. The other direction (i.e. how many n-grams of x_i are found in x_j) is less important for our purpose.

Our redundancy function is similar to the loss function described by Berg-Kirkpatrick et al. (2011), where the authors count only the bi-gram overlaps between two text units. However, using bi-grams in a redundancy function can only work well if the goal is also to maximize an objective function that purely uses bi-grams to measure the similarity between two text units, such as ROUGE-R2 (Lin 2004). However, if the aim is also to maximize another objective such as ROUGE-SU4, where uni-grams are also used for computing the similarity, then uni-grams must also be considered when doing the similarity check. Since we aim to maximize both R2 and RSU4 (see Section 8.5.2), we set $n = 2$, i.e. we use both uni-grams and bi-grams to measure the similarity between two sentences. We divide by 2 to normalize the result to 1.

The higher the value of a sentence's SF, the more redundant it is relative to the remaining sentences. We use the framework as presented in the previous section and run it with and without the SF feature. In the mode with SF included, it is left to the training module to decide which values of SF to select to produce optimal summaries.

It is possible that the best summary is the one containing sentences with high SF scores. Highly similar sentences are likely to cover a lot of information contained in the input documents, so that including them in a summary would result in a concise description of the input documents. This is somewhat similar to the idea of selecting a sentence if it is highly connected to many other sentences (Kruengkrai & Jaruskulchai 2003, Mihalcea & Tarau 2004). This sentence is used as representative for all the other connected sentences.

Note that since this SF approach computes similarity against all input sentences, it does not reflect the actual summarization scenario in which it is only the similarity relative to a subset of already selected sentences that matters. Nonetheless, we deemed it worth investigating and the results bear this out (cf. Section 8.6.2).

Alternatively, it could happen that the best summary is the one with sentences having low SF scores leading to a summary containing diverse information. Such summary would cover all topics in the input documents, not only their main focus. Diversity has been shown to be an important factor in Information Retrieval (IR) where more diversity in the retrieved documents leads to higher user satisfaction (Lin et al. 2010), and it has also been used in extractive text summarization as a way to reduce redundancy of the output summaries (Carbonell & Goldstein 1998, Zhu et al. 2007).

A third possibility is that SF values from the middle of the range lead to the best summaries resulting in summaries balanced between the two extremes (Li et al. 2009). Therefore, whether to favour a focused, middle range or diverse summary is left to be learned by the training module and is not set manually in advance.

8.3.2 A Search with Redundancy Reduction*

Our second approach to dealing with redundancy within multi-document summaries implements the idea of omitting or *jumping over* redundant sentences when selecting summary-worthy sentences from the input documents. When sentences from the input documents are merged and sorted in a list according to their summary-worthiness, the generation of a summary starts by first including a top summary-worthy sentence in the summary, then the next one until a desired summary length is reached. If a sentence from the list is found to be similar to those already included in the summary (i.e. to be redundant), then this sentence should not be included in the summary, but rather *jumped over*.

We integrate the idea of *jumping over* redundant sentences into our A* search algorithm described in Section 8.2. The difference between the implementation we present in this Section and the one described in Section 8.2 is the integration of a function $jump(\cdot)$ into the search process. We use this function to jump over a sentence when it is redundant with respect to the summary $\mathbf{y}$. Thus we do not only skip a sentence if it is too long (line 11 of Algorithm 1 and lines 7 and 13 of Algorithm 2) as in the algorithm described in Section 8.2 , but also when it is redundant compared to the summary created so far. To do this we replace the jump conditions of the A* search algorithm described in Section 8.2 with:

$$\text{lengthConstraintsOK} \wedge \text{jump}(\mathbf{y}, y) == false \tag{8.5}$$

where lengthConstraintsOK represents the situation when the next sentence does not violate the summary length in Section 8.2 and jump$(\mathbf{y}, y) == false$ the case where the next sentence is not redundant and therefore not to be jumped over.

Note that jumping over redundant sentences does not violate the admissible behavior of the heuristic. If on a path from the start state to an end state no redundant sentences are found, then there is no jump, and our modified heuristic reduces to the one described in Section 8.2. If a jump happens, the heuristic score is still an upper bound on the actual summary score that is reachable by that particular path and therefore admissible. The reason for this is that for each path where a jump happens only the number of sentences to be visited is reduced, but the order of the sentences in the sorted list and their raw scores are kept the same. Thus we can say that the heuristic with jump integrated is equal to the heuristic described in Section 8.2 only with a reduced number of sentences to be used in summary generation. Reducing the number of sentences to visit is related to the idea of restricting the number of states to be visited based on an initial state (Hölldobler et al. 2006). In our case the initial state is the current state, i.e. a summary generated so far.

We use the strategy of jumping based on a redundancy threshold (JRT) to implement the jump$(, .,)$ function. This method is as follows.

We use the similarity score of a sentence x_i with respect to the summary $\mathbf{y}$ and a redundancy threshold R to decide whether to jump over the sentence or not. In general we jump over a sentence x_i if its similarity score is above R (see Algorithm 3). The similarity scores are computed using the sim$(., .)$ function shown in Equation 8.4.[4]

[4] In the Equation the expression sentence x_i can be replaced by summary $\mathbf{y}$.

Algorithm 3 Jump when similarity score is above a threshold R, $jump(\mathbf{y}, x_i)$

```
Require: require a redundancy threshold $R$
Require: require a sentence $x_i$
Require: require a summary $\mathbf{y}$
if sim($\mathbf{y}, x_i$) $\leq R$ then
    return $false$
end if
return $true$
```

The idea of omitting redundant sentences if their redundancy score exceeds a threshold has already been introduced in previous work (Barzilay et al. 1999, Lin & Hovy 2002, Saggion 2008, Sauper & Barzilay 2009). However, in contrast to these studies, in which the redundancy threshold is set manually, we learn it automatically.

To learn the redundancy threshold R we make use of the entire framework (search and training) and proceed as shown in Figure 8.2. The learning procedure starts in the box denoted with *Start*. In the beginning (the top left of the figure) we create a random value $R \in (0, 1]$. In addition to this R we generate two further values: $R + 0.1 \leq 1$ and $R - 0.1 > 0$. These two additional numbers are used to move R towards its optimum value. All three Rs are used to generate n best summaries for each training document set using A* search. In the A* search we also require a prediction model to score the sentences. For this we start with an initial prediction model (initial feature weights W).

For each of the R values (denoted with r in the figure) we then create an n best list using A* search leading to $3 \times n$ summaries. If there are summaries from a previous step, we merge them with the new n best list, so that in training the entire history of n best lists is provided. For each summary its corresponding R value is known. Note that we copy the summaries from the previous steps to the current one. This is required by our discriminative training algorithm (see Section 8.5) as it performs better when it sees all n-best summaries of all previous steps.

Next, these n best summaries are input to the discriminative training algorithm to train new weights W', i.e. a new prediction model. The way the discriminative training algorithm creates the new weights is based on feature weight variations with the aim of maximizing the total summary quality scores obtained by an automatic metric such as ROUGE. After each variation of a feature weight, the training algorithm sorts the summaries of each document set using the summary scores generated by the summarization model. Then from each document set it

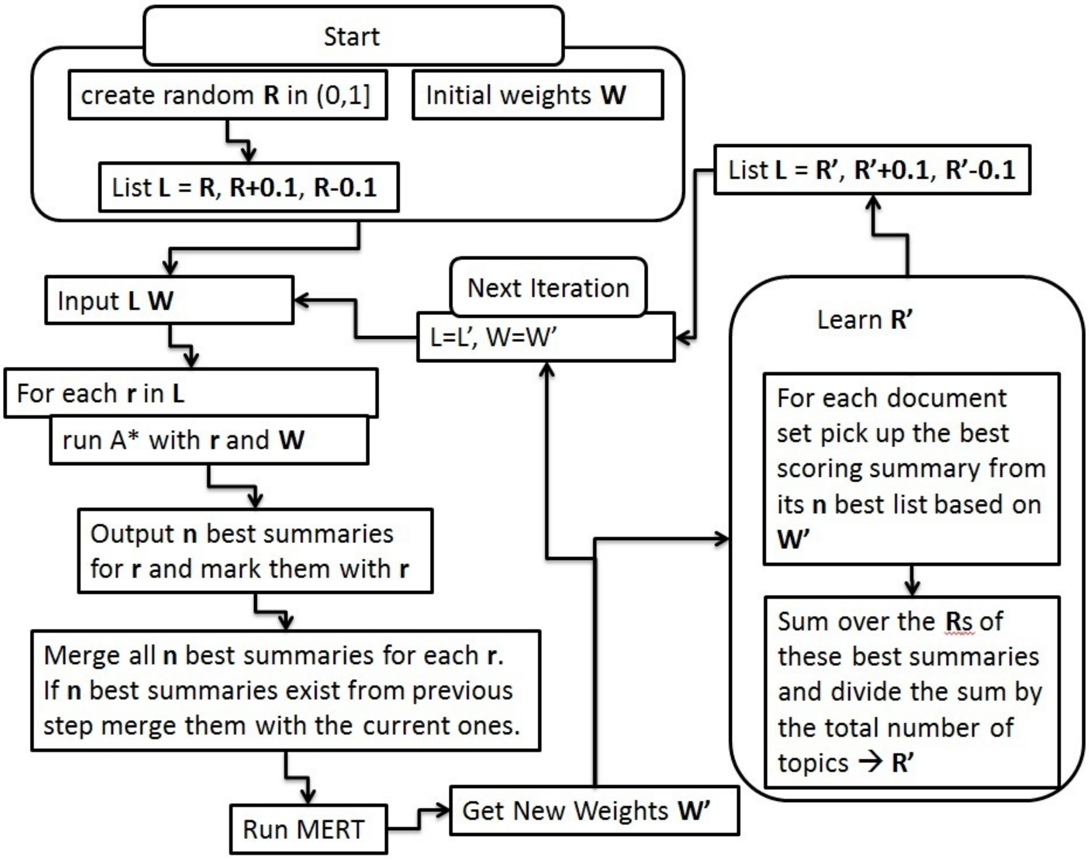

Start
create random **R** in (0,1]
Initial weights **W**
List **L** = **R**, **R+0.1**, **R-0.1**
Input **L W**
For each **r** in **L**
run A* with **r** and **W**
Output **n** best summaries for **r** and mark them with **r**
Merge all **n** best summaries for each **r**. If **n** best summaries exist from previous step merge them with the current ones.
Run MERT
Get New Weights **W'**
Next Iteration
L=L', W=W'
List **L** = **R'**, **R'+0.1**, **R'-0.1**
Learn **R'**
For each document set pick up the best scoring summary from its **n** best list based on **W'**
Sum over the **Rs** of these best summaries and divide the sum by the total number of topics → **R'**

Figure 8.2: Learning the redundancy threshold R.

picks the top summary and sums the ROUGE metric scores. If the sum of scores is better then the previous one, it continues with the variation of the other feature weights until a maximum metric score is achieved. When learning the new R' we make use of these top summaries from the document sets. Using the new W' we identify for each document set the top summary, sum the R values of those summaries (in total m for m document sets) and divide the sum by m to obtain the new R'. We replace R with R' and W with W' and repeat the entire process until no new summaries are added to the n best list, when the process stops.

8.4 Searching for a Well Ordered Summary

To create a well ordered or a coherent summary we perform automatic sentence ordering while generating the summary. In Chapter 7 we used dependency patterns along with their manually annotated categories to restrict the inclusion of sentences in the summary to those containing patterns from the manually annotated categories. We also used those patterns to order the sentences included in the summary. Our evaluation showed that doing this led to significant improvements in ROUGE scores and to better readability scores in the human evaluation. Therefore, we adopt this idea in our automatic sentence ordering.

The approach to sentence ordering we propose is based on modeling the flow of information types between sentences found in geo-located entity type corpora. The information types are represented using dependency patterns derived from dependency parse trees. To learn or to exploit such models requires that each sentence processed in training or in summary generation is associated with one or more information types. Sentence (1), e.g., contains two information types: *entity type* and *entity location* information.

(1) *Uppsala Cathedral (Swedish: Uppsala domkyrka) is a cathedral located centrally in the city of Uppsala, Sweden.*

(1) tells us to which entity type *Uppsala Cathedral* belongs (i.e., it is a cathedral), and secondly tells us where the cathedral is located. For each entity type we learn models of information flow by parsing descriptions of multiple instances of entities of that type and observing recurring sequences of dependency patterns. When composing the summary the A* search uses the information type flow model to determine the most likely order between the sentences in the summary.

8.4.1 Generating Information Type Flow Models

An information type flow model FM is created for each entity type corpus containing a collection of Wikipedia articles belonging to a specific entity type such as *bridge*. The model contains a list of dependency pattern pairs with frequency counts. The frequency count for each pair of patterns is obtained from the entire corpus. Two dependency patterns build a pair if they occur in adjacent sentences. Each FM specific to an entity type models the flow between the different information types of adjacent sentences within the articles of that entity type corpus. The following examples show a possible information type flow extracted from the *bridge*, *museum* and *church* corpora.

```
FROM BRIDGE corpus:
                        @@entity is entityType
```

```
                    @@is a entityType
                    @@is entityType crosses
entity is entityType@@was built date
                    @@it was built
                    @@entityType was built
was built date      @@is feet long
                    @@entityType is long
                    @@was added register
is feet long        @@entity has design
                    @@has a design
                    @@has wooden design
entity has design   @@deck is made
                    @@is made planks
                    @@is painted red

FROM MUSEUM corpus:
                    @@entity is entityType
                    @@is a entityType
                    @@is entityType located
entity is entityType@@it is located
                    @@it located location
                    @@entity is located
it is located       @@entity is founded
                    @@was founded date
                    @@is oldest entityType
entity is founded   @@entity collections include
                    @@entity is run
                    @@entity was designed


FROM CHURCH corpus:
                       @@entity is entityType
                       @@is a entityType
                       @@is entityType located
entity is entityType   @@entityType is located
                       @@was built date
                       @@entityType was built
entityType is located  @@added national register
                       @@entityType was commissioned
                       @@was commissioned date
added national register@@to serve community
                       @@serve catholic community
                       @@entityType was founded
```

The patterns are split by “@@”. The patterns on the right of “@@” follow the ones on the left. For each pattern on the left we include three possible patterns that follow the left one. For each left pattern the first right pattern is the most frequent pattern following the one on the left, the second one the second most frequent and the third one the third most frequent one. In the first pattern pairs, no dependency pattern is found on the left of “@@” indicating a possible start of an article by using the information type on the right of “@@”. We extracted these examples by following the most frequently occurring starting pattern. For instance, the pattern flow for the *bridge* entity type says that the article should first start with the entity type definition (e.g. *Galata Köprü is a bridge*). Next, it should contain information about its construction date. This is followed by more descriptive information such as the length, design and the inside and outside look.

We use such a flow model to compute the flow probability between two sentences using maximum likelihood estimation. Let y_k and y_i be two sentences such that y_i immediately follows y_k. Then the flow probability between y_k and y_i is given by the following equation:

$$fp(y_k, y_i) = \underset{<m \in M, n \in N>}{\arg\max} \frac{C_{FM}(pattern_m, pattern_n) + 1}{C_{FM}(pattern_m) + V} \tag{8.6}$$

where $pattern_m$ is a dependency pattern extracted from y_k and $pattern_n$ is a dependency pattern y_i (note that a sentence can have more than one dependency pattern). We represent the set of patterns extracted from sentence y_k by M and those from y_i by N. The function $C_{FM}(.,.)$ returns the count of pair of patterns from the flow model FM, and $C_{FM}(.)$ the frequency count for a single pattern. The counts are taken across a corpus. We use add-one smoothing (Jurafsky & Martin 2008) to address unseen transitions between information types; i.e. we add 1 to the numerator and the number of information type pairs in the flow model FM (V) to the denominator.

8.4.2 Using Information Type Flow Models in Summary Generation

While generating a summary each sentence in the input documents has associated with it a set of information types. We use these information types along with the flow model to create the most likely flow between the sentences in the summary. More precisely, for a given summary we extract dependency patterns from the last sentence in the summary. We also extract patterns from the current sentence that is a potential candidate to be included in the summary. Based on the extracted patterns from both sentences and the flow model FM we compute the flow probability between the current candidate and the last sentence in the summary. If this probability is too low than the candidate sentence is not included in the summary and is jumped over.

However, to determine when a probability is too low requires some kind of threshold that all probabilities can be compared to. One way to determine this threshold is to set it to an intuitively sensible, but arbitrary value from the interval $[0, 1]$. However, this is not satisfactory since there is no guarantee that another value from the interval would not lead to summaries with a better information flow. In theory such a threshold could be learned from the data. For this, one would generate summaries using different thresholds in the interval and in each case measure the information flow quality of the resulting summaries. The threshold leading to the highest quality score would be taken as the learned flow threshold. However, in practice it is not possible to do this since there is no metric to determine the flow quality that can be computed automatically and assessing the large numbers of summaries that would arise in learning an optimal threshold using manual readability assessment is simply not feasible. Therefore, instead of determining a threshold value we make the decision about inclusion or exclusion of the current candidate sentence based on the sentence following the candidate sentence in the list of all candidate sentences as shown in Algorithm 4.

Algorithm 4 Excludes the current sentence y_i when the following sentence y_j is more likely to follow the last sentence chosen for the summary, $jumpFM(\mathbf{y}, x_i)$

```
Require: sentence $y_i$
Require: sentence $y_j$
Require: last sentence in the summary $y_k$
if fp$(y_k, y_i) \geq$ fp$(y_k, y_j)$ then
    return $false$
end if
return $true$
```

We compute the flow probability between the sentence y_j and y_k. y_j denotes the sentence in the candidate sentence list[5] which follows the current candidate sentence y_i. y_k is the last sentence in the summary. We then compare this flow probability between y_j and y_k to the probability obtained between y_i and y_k. If the probability between y_j and y_k is higher than the probability between y_i and y_k, we jump over y_i and do not include it in the summary. Otherwise, it is included in the summary.

Note that a sentence that is not likely to follow the last sentence in the summary at one point in time and is therefore jumped over can again become likely later as more sentences are added to the summary, so that the last sentence in the summary changes. Such a sentence should be reconsidered when the last sentence in the summary changes. However, reconsidering such sentences breaks the admissible behavior of the heuristic used in the A* search described above. We only include sentences in the summary in decreasing order of sentence score. This ensures that there is an upper bound on the maximum score a summary can achieve, as required for the A* heuristic to be admissible. A sentence jumped over has always higher score than the following one. However, if this order of the sentences according to scores is not kept, as would be required if sentences were reconsidered once they have been discarded, the upper bound on the achievable score for the summary cannot be guaranteed. Due to this we do not reconsider sentences once they are jumped over.

Sentences in the input data containing starter patterns, i.e. patterns extracted from the first sentence in the articles of the corpus, are all potential first candidate sentences for the final summary. If the summary is constructed by measuring the flow according to these starter sentences, the sentence order in the summary is likely to be better than if a non-starter sentence is used. Thus, in the summarization process we assign to each of the input sentences an additional feature to indicate whether the sentence contains a starter pattern or not. We rank such sentences more highly than those which do not contain starter patterns. The order of such starter sentences is determined based on their summary worthiness scores. In this way we force the A* search to start with such a starting sentence and construct the summary by biasing the order according to this starter sentence. By doing this we aim to maximize the quality of the sentence ordering in the final summary.

Similarly to the redundancy approach, we replace the conditions for skipping a sentence from inclusion into a summary (see line 11 of Algorithm 1 and lines 7 and 13 of Algorithm 2) with the following function:

[5] We order sentences based on their scores and number of words ($\frac{score}{numberofwords}$) and visit them in descending order.

$$\text{lengthConstraintsOK} \land \text{jumpFM}(\mathbf{y}, y) == false \tag{8.7}$$

When integrating both the redundancy reduction and sentence ordering as constraints while generating a summary we use the following function as replacement for the skip condition in line 11 of Algorithm 1 and in lines 7 and 13 of Algorithm 2:

$$\text{lengthConstraintsOK} \land \text{jump}(\mathbf{y}, y) == false \land \text{jumpFM}(\mathbf{y}, y) == false \tag{8.8}$$

8.4.3 Related Work

In Section 2.2.1 we discussed various approaches to sentence ordering. One of the earliest approaches to sentence ordering for multi-document summarization is chronological ordering (McKeown et al. 1999, Lin & Hovy 2001, Radev et al. 2004). In the chronological approach sentences within a summary are ordered according to the publication date of the input documents, so that the sentences coming from the documents published earlier occur earlier in the summary than the sentences which come from documents published later. However, Barzilay et al. (2002) show that chronological information is not sufficient for ordering sentences within a summary. Others have investigated relationships between referring expressions occurring at the beginning of each sentence to perform sentence ordering (Pollock & Zamora 1975, Saggion et al. 2003, Farzindar et al. 2005). Since resolving the relationship between the referring expressions is difficult when multiple documents are used, the impact of this approach for sentence ordering is rather limited.

Barzilay & Lee (2004) and Fung & Ngai (2006) perform sentence ordering based on topic orders derived from input documents. This approach differs from ours in that we derive our ordering information not from the texts to be summarized but from collections of articles about entities of the same type as the entity for which a summary is to be generated. Determining the order of summary sentences based on the order of topics in input documents can be difficult if the input texts are documents which do not share sequences of topic transitions. This is the case with the web-documents about geo-located entities which we use in our application scenario.

There has also been a focus on probabilistic approaches to sentence ordering (Lapata 2003, Soricut & Marcu 2006, Barzilay & Lapata 2005, 2008, Lin et al. 2011, Bollegala et al. 2012, Louis & Nenkova 2012). However, these studies perform the sentence ordering task in isolation from a (multi-) document summarization task. The aim is to reorder a full set of unordered input sentences. However, in a multi-document summarization scenario only a subset of the input sentences is extracted from different documents at arbitrary positions. Therefore, these approaches may not be applicable for sentence ordering in a multi-document summarization setting.

Our information type inspired approach is related to sentence categorization into predefined categories according to the information the sentence conveys. Various authors have proposed and used categories to order the sentences (Liddy 1991, Teufel & Moens 2002, Teufel 2010, Bollegala et al. 2010, Liakata et al. 2010). Liakata et al. (2010), for instance, work with scientific papers and use predefined manually created categories such as *Background Hypothesis, Motivation, Goal, Object, Method, Model, Experiment, Observation, Result* and *Conclusion* to map the input sentences into. In the summarization process they propose using the categories in the given order and take for each category the highest ranking sentence to include in the summary.[6]

However, our sentence ordering method differs crucially from this and similar approaches in that we do not assume a fixed number of predefined information type categories into which to classify sentences. Instead, our set of information types is derived automatically from existing geo-located entity descriptions. In addition, in our approach the order of inclusion of information types in the final summary is decided based on the information type flow model, whereas in the aforementioned related work, not only the categories, but also their order is predefined and applied to all different collections of input documents. Because we make no assumption about the categories and their order, our approach is not limited to a domain of geo-located entities, but can be used in any domain.

8.5 Training and Testing procedure

Similar to the training and testing procedure described in Section 7.6 we use for training and testing our model summary set described in Chapter 6. As in Section 7.6 these model summaries are divided into training and testing. We use the training set to obtain weights for the summarization features. However, unlike the training approach outlined in Section 7.6 that used single sentences to obtain the feature weights in this chapter we use complete summaries to obtain those values. Obtaining the feature weights or training the prediction model is the subject of the next section.

[6] The results of this summarization step are not published yet. The described idea is based on personal communication with the author.

8.5.1 Training the Prediction Model

We frame the training problem as one of finding model parameters λ, such that the predicted output $\hat{\mathbf{y}}$ closely matches the gold standard $\mathbf{r}$.[7] The quality of the match is measured using an automatic evaluation metric. We adopt the standard machine learning terminology of loss functions, which measure the degree of error in the prediction, $\Delta(\hat{\mathbf{y}}, \mathbf{r})$. In our case the accuracy is measured by the ROUGE score, R, and the loss is simply 1 - R. The training problem is to solve

$$\lambda = \arg\min_{\lambda} \Delta(\hat{\mathbf{y}}, \mathbf{r}) \tag{8.9}$$

where $\hat{\mathbf{y}}$ and $\mathbf{r}$ are taken to range over sets of subsets of sentences taken from the input documents.

The prediction model is trained using the minimum error rate training (MERT) technique (Och 2003). MERT is a first order optimization method using Powell search to find the parameters which minimize the loss on the training data. MERT requires n-best lists which it uses to approximate the full space of possible outcomes. A* search is used to construct these n-best lists and MERT to optimize the objective metric, i.e. ROUGE, that is used to measure the summary quality.

8.5.2 Overview of Training and Testing procedure

In both training and testing we use our geo-located entity collection described in Chapter 6. As in Chapter 7 we again use the same 202 entities for training and the remaining 105 entities for testing.

In both training and testing for each entity we generate a set of summaries using our summarizer (see Chapter 3) and A* search and compare the results against model summaries of that entity using ROUGE.

In the training mode we run A* search with an initial prediction model and generate n best summaries with length threshold L for each entity.[8] On these n best summaries we run MERT to update or re-train the prediction model. We iterate this process until the prediction model does not change. In the testing mode we only use A* search once and run it with the prediction model

[7] The human generated gold standard is typically an abstractive summary, and as such it is usually impossible for an extractive summarizer to match it exactly.

[8] We set n to 10 and L to 200 words.

learned in the training mode. We generate only one summary for each entity.[9] When training the prediction model we again use ROUGE as a metric to maximize. In particular we use R2 and RSU4.

This training and testing set up integrates training the prediction model with the search for the best summary. The prediction model is trained on the summaries produced by the search method in combination with reference summaries (i.e. gold standard summaries generated by humans). In this way our work departs from related work (see Section 2.2.1) in which the prediction model is trained on the reference summaries only. In that case the model does not optimize summary quality but some other peripheral objective, so this does not guarantee that the model will successfully distinguish between good and bad summaries. By implementing the search with the training of the prediction model intact, we ensure that the prediction model parameters are learned in such a way that the best scoring *whole summary* under the prediction model has a high score under the ROUGE evaluation metric. Therefore, we hypothesize that these summaries will have higher quality than summaries generated by methods in which training and search are two independent steps, i.e. where the prediction model is first trained on reference summaries and then used in search.

8.6 Evaluation

In this section we report three sets of experiments that we performed in order to evaluate our summarization framework described in previous sections. We first evaluate the discriminative learning with integrated search approach to training the prediction model (Section 8.6.1). To do so, we compare this approach, which works with whole summaries, to a sentence-level approach commonly taken in previous work and also adopted in Chapter 7. The results of this system on a test data set provide us with a baseline to which we compare summarizers with integrated redundancy reduction (Section 8.6.2) and sentence ordering (Section 8.6.3). Both ROUGE evaluation and human readability evaluation are reported to assess the quality of the summaries with these summary composition challenges addressed and compare them to those generated without redundancy and sentence ordering features.

[9] We use same summary length threshold $L = 200$ as in the training.

8.6.1 A Comparison Between Different Approaches for Training

As described in Section 8.5 above, we use the MERT technique to learn the feature weights in such a way that the best scoring *whole summary* under the prediction model has a high score under the ROUGE evaluation metric. This differs from the training strategy using linear regression which we reported in Chapter 7. One potential downside of the regression based methods is that they work in a single instance mode, i.e. on single sentences. This means that the prediction model is trained to identify the best sentences. However, if best sentences are combined together into a summary, there is no guarantee, that the resulting summary will be optimal too. Therefore, we assume that it is better to predict the best actual outcome of the summarizer, i.e. the output summary, instead of its components (single sentences). Unlike the regression method, the discriminative training method we propose here can work on the summary level. It uses the n-best whole summaries for each geo-located entity to create the prediction model, and we hypothesize that a summarizer using such a prediction model can generate better summaries than the one based on sentence-level training.

To assess whether this hypothesis holds, we compare our summary-based training approach with the search intact to a sentence-based training baseline approach. For both approaches we report the results on both the training and testing data sets (cf. Section 8.5.2). The rationale behind using training data in this evaluation is to inspect whether MERT indeed maximizes the metric it is supposed to maximize. For example, if the metric to maximize is R2 during training, we expect that R2 results on the training data set will be higher than those for RSU4 and vice versa. It is only possible to draw this conclusion if the summarizer is tested on the training data, since test data set is unseen during the training. However, for establishing how good our summarizer with MERT training is, we assess its performance on the test data set. This evaluation serves as a baseline for further assessment of summarizers with integrated redundancy and sentence ordering.

For these evaluations on both training and test data sets we distinguish between three different summaries: *wordLimit*, *sentenceLimit* and *regression*, where *wordLimit* and *sentenceLimit* summaries are the ones generated using the model trained by MERT and *regression* summaries are our baseline.

wordLimit and *sentenceLimit* refer to the summary length constraint. We differentiate between summaries with a word limit (*wordLimit*, set to 200 words as mentioned in Section 8.5) and summaries containing *N* sentences (*sentenceLimit*) as stop condition in A* search. By doing so, we aim to assess whether in our framework, with training and search combined, the way the input to the training is constrained in length influences the quality of the output. Specifically, there may be a drop in performance if the summarizer is trained in *wordLimit* mode, i.e. if it is trained to generate summaries in terms of a pre-specified number of words, but is run in the *sentenceLimit* mode, where it needs to generate a predefined number of sentences, some of which may be longer than others, compared to the condition where training and testing constraints are exactly the same.

To test this, for *sentenceLimit* mode we set *N* so that in both *wordLimit* and *sentenceLimit* summaries we obtain more or less the same number of words. Since our training data contains on average 17 words for each sentence we set *N* to 12 (12*17=204). However, this is only the case for the evaluation on training data. For testing data for both *wordLimit* and *sentenceLimit* we generate summaries with the same word limit constraint i.e., we use the condition *wordLimit* to constraint the summary length limit in the testing. This allows us to have a fair comparison between the ROUGE recall scores and assess whether training in one length constraint scenario and testing in another influences the results.

The *regression* summaries are our baseline. In these summaries the sentences are ranked based on the weighted features produced by Support Vector Regression (SVR) (Joachims 2002a).[10]

Ouyang et al. (2011) use multi-document summarization and regression methods to rank sentences in the documents. As a regression model they used SVR and showed that it outperformed classification (Kupiec et al. 1995, Chuang & Yang 2000, Mani & Bloedorn 1998, Zhou & Hovy 2003, Hirao et al. 2002) and Learning To Rank methods (Amini et al. 2005, Amini & Usunier 2009, Fisher & Roark 2006, Toutanova et al. 2007, Wang et al. 2007, Metzler & Kanungo 2008) on the DUC 2005 to 2007 data. For this reason we use SVR as a baseline system for learning feature weights, instead of the linear regression we used in Chapter 7. However, both these regression-based approaches are comparable, as the weights are learned based on single sentences. To have a fair comparison between all our summary types we use these weights to generate summaries using A* search with the word limit as constraint. We do so for both results on training and testing data sets.

The results for the training data set are shown in Table 8.1. The table shows ROUGE recall numbers obtained by comparing model summaries against automatically generated summaries on the training data. Because in training we used two different metrics (R2, RSU4) to train weights we report results for each of these two different ROUGE metrics. In the table for each summary type ($wordLimit$, $sentenceLimit$ and $regression$) we have two rows. Each row

[10] We use the term *regression* to refer to SVR.

Table 8.1: ROUGE scores obtained on the training data.

Type	metric trained on	R2	RSU4
wordLimit	R2	**0.3208**	0.3510
	RSU4	0.3197	**0.3585**
sentenceLimit	R2	**0.3601**	0.3890
	RSU4	0.3546	**0.3929**
regression	R2	0.1949	0.2413
	RSU4	**0.2031**	**0.2562**

Table 8.2: ROUGE scores obtained on the testing data.

Type	metric	R2	RSU4
wordLimit	R2	**0.0895**	**0.1423**
	RSU4	0.0794	0.1340
sentenceLimit	R2	0.0717	0.1251
	RSU4	0.0778	0.1312
regression	R2	0.0560	0.1043
	RSU4	0.0668	0.1226

indicates on what metric (R2, RSU4) the prediction model is trained. For each of these metrics we report both R2 and RSU4 results – obtained only on the training data.

As shown in Table 8.1 the scores for *wordLimit* and *sentenceLimit* summaries are always at maximum for the metric they were trained on (this can be observed by following the main diagonal of the result matrix). This confirms that MERT is maximizing the metric for which it was trained. However, this is not the case for the regression method. The scores obtained with RSU4 metric trained weights achieve higher scores on R2 compared to the scores obtained using weights trained on that metric. This may be due to SVR being trained on sentences rather than over entire summaries, and thereby not adequately optimising the metric used for evaluation.

The results for the test data set are shown in Table 8.2.

The *wordLimit* summaries score highest compared to the other two types of summaries. This is different from the training results where *sentenceLimit* summary type summaries are the top scoring ones. As mentioned earlier the *sentenceLimit* summaries contain exactly 12 sentences, where on average each sentence in the training data has 17 words. We picked 12 sentences to achieve roughly the same word limit constraint ($12 \times 17 = 204$) so they can be compared to the *wordLimit* and *regression* type summaries. However, these *sentenceLimit* summaries have an average of 221 words, which explains the higher ROUGE recall scores seen in training compared to the *wordLimit* summaries (where a 200 word limit was imposed).

The *wordLimit* summaries are significantly better than the scores from the other summary types, irrespective of the evaluation metric.[11] It should be noted that these summaries are the only ones where the training and testing had the same condition in A* search concerning the summary word limit constraint – in both training and testing we used 200 words as the summary length threshold. The scores in *sentenceLimit* type summaries are significantly lower than *wordLimit* summaries, despite using MERT to learn the weights. This shows that training the true model, i.e. the model, which learns to predict the output of the same kind as training input, is critical for getting good performance as indicated by ROUGE. If the goal is to generate high scoring summaries under a length limit in testing, then the same constraint should also be used in training.

The *regression* type summaries, where feature weights are trained on single sentences using SVR achieved the lowest ROUGE metric scores. Therefore we conclude that summary-level training using discriminative learning leads to better prediction models than the commonly used sentence-level training. Note, in both training types (single sentence using SVR vs. full summaries with MERT) we used the same set of features to score the sentences.

8.6.2 Integration of Redundancy Features

In this section we compare results obtained using the above summarization system with and without redundancy reduction features. We show that both proposed methods for globally optimizing the summaries by reducing redundancy (cf. Section 8.3) lead to improved ROUGE scores compared to a setting where redundancy is not addressed ($wordLimit$). The results are given in Table 8.3.

Introducing an additional feature SF (see Section 8.3.1) to reduce the redundancy within the summary leads to moderate improvement compared to the original setting ($wordLimit$) where such a feature is not used to score the sentences/summary.[12] This shows that MERT tries to learn to use SF in a way to improve the summary quality. However, the results show that this does not lead to the best performance that can be achieved. The best overall results are achieved

[11] Significance is reported at level $p < 0.001$. We used Wilcoxson signed ranked test to calculate significance.

[12] The R2 and RSU4 results for the original setting are lower than from those shown in Table 8.3. The reason for this is that the results shown in Table 8.2 are obtained using summarization features excluding the dependency patterns, whereas for the results shown in Table 8.3 we used all the summarization features used to obtain the results shown in Table 8.2 as well as the dependency pattern feature.

Table 8.3: Experimental results. In each row the results were obtained with the prediction model trained on the metric of that row.

Recall	wordLimit	SF	JRT
R2	.094	.096	**.109***
RSU4	.146	.152	**.167***

Table 8.4: Example summary about the *Akershus Fortress*. This summary is generated using the setting JRT.

The Akershus castle and fortress are located on the eastern side of the Oslo harbor. The fortress was first used in battle in 1306. The original Akershus Castle is located inside the fortress. Akershus Fortress (Norwegian: Akershus Festning) is the old castle built to protect Oslo, the capital of Norway. The fortress was built in 1299, and the meaning of the name is 'the (fortified) house of (the district) Aker'. In the 1600s a castle (or in norsk, "slott") was built. The fortress was reconstructed several times to withstand increasing fighting power. The fortress has successfully survived many sieges, primarily by Swedish forces. The fortress was strategically important for Oslo and therefore for Norway as well. The castle is well positioned overlooking Oslo's harbour. The fortress has survived many sieges, primarily Swedish, and, with the exception of WWII, never been conquered by a foreign enemy. The fortress was first used in battle in 1308, when it was besieged by the Swedish duke Erik of Sdermanland, who later in the same year won the Swedish throne. The fortress was strategicly important for the capital, and therefor, Norway aswell.

using the JRT method where a redundancy threshold R is automatically learnt. It significantly ($p < 0.001$) outperforms both the SF and the setting without redundancy detection.[13]

The values of the learnt redundancy threshold R differ for different ROUGE metrics: for R2 the threshold is 0.5338 and for RSU4 0.4675. Different R values are to be expected given the different properties of R2 and RSU4. Compared to R2 the redundancy threshold for RSU4 is stricter, which reflects the way RSU4 works. Since RSU4 measures the uni-gram overlap between two text units and also bi-grams where gaps of up to four words are allowed between the words, it is able to capture more similarities between sentences than R2, where single word overlaps are not captured. In R2 gaps within a bi-gram are not allowed. For example bi-grams AB and $A??B$ are identical in RSU4, but not in R2. Consequently, a stricter redundancy threshold is required

[13] We have also studied different alternative methods to the JRT one to be used in the jump$(.,.)$ function such as favouring the following sentence to the current one if it is less redundant than the current one or combining the redundancy scores with the actual raw scores of the sentences and jumping only over the current sentence if the combined score is less than the combined score of the following sentence. However, the results by these alternative methods led only to moderate improvement over the baseline. For this reason we do not report those results.

in RSU4 than in R2. This fact illustrates also that there cannot be a single R for every ROUGE metric and highlights the importance of learning it for each ROUGE metric separately.

From the example summary for the geo-located entity *Akershus Fortress* shown in Table 8.4 we can see that the summary does capture a variety of facts about the fortress, such as when it was built, where it is located, etc. This type of essential information about the fortress occurs only once in the summary. What is repeated in most of the sentences are referring expressions such as the name of the entity (*Akershus Fortress*) or the entity type type (*the fortress* or *the castle*). Sentences containing referring expressions are more likely to contain relevant information about the fortress in the model summaries than sentences which do not contain such expressions.

8.6.3 Integration of the Sentence Ordering Feature

Although the summarizer with redundancy features integrated outperforms the baseline without such features with respect to ROUGE scores, the example summary about Akershus castle (Table 8.4) indicates that there is still scope for improvement in the coherence of the output summaries. This summary reads like a bag of sentences, so we aim to improve its readability by adding a sentence ordering feature as described in Section 8.4. We add this feature to the summarizer setting in which the redundancy is addressed (JRT, cf. previous section) and refer to the new setting with both redundancy and sentence ordering as $JRTF$.

Unlike informativeness and redundancy, it is difficult to assess the sentence ordering within the summary using ROUGE scores alone. Human readability evaluation in the style of DUC and TAC on the other hand is a better tool for this, since it explicitly contains a question about coherence of the summaries, which is directly influenced by the sentence order. Therefore, we expect that $JRTF$ summaries will get higher scores on the coherence criterion relative to summaries without sentence ordering addressed. Integrating sentence ordering may also improve clarity and focus of the summary, as it is easier to read a well flowing summary than one which is a bag of sentences such as that shown in Table 8.4.

In addition it is interesting to know to what extent we can approach the Wikipedia baseline summaries regarding sentence order with our fully automatic methods. Our readability evaluation in Chapter 7 showed that none of the automated summaries could outperform the Wikipedia baseline on the *clarity, coherence, focus and redundancy* criteria of the human readability assessment. If our methods for redundancy reduction and sentence ordering can improve the output summaries in relation to these criteria, then we could present a fully automatic system which

Table 8.5: Readability evaluation results.

	5			4			3			2			1		
Criterion	wL	JRT	JRTF	wL	JRT	JRTF	wL	JRT	JRTF	wL	JRT	JRTF	wL	JRT	JRTF
clarity	6.2	22.4	34.0	41.7	73.5	60.0	29.2	2.0	2.0	20.8	0	4.0	2.1	2.0	0
coherence	6.2	28.6	30.0	18.8	42.9	52.0	33.3	24.5	6.0	37.5	4.1	10.0	4.2	0	2.0
focus	6.2	26.5	30.0	33.3	61.2	58.0	29.2	12.2	6.0	29.2	0	4.0	2.1	0	2.0
grammar	4.2	12.2	26.0	58.3	67.3	54.0	12.5	4.1	4.0	20.8	14.3	16.0	4.2	2.0	0
redundancy	4.2	8.2	26.0	8.3	61.2	66.0	2.1	12.2	4.0	41.7	18.4	4.0	43.8	0	0

nevertheless achieves better results than the system which requires manual intervention in pattern categorization.

To assess the $JRTF$ summaries we perform a manual readability evaluation comparable to that described in Chapter 7. In the evaluation we asked three assessors to judge the summaries. The assessors were given the same instructions as in Chapter 7. Apart from $JRTF$ summaries we also assess the summaries produced without any global features (*wordLimit*) as well as the summaries generated using the JRT feature. For all three summarization types we only assessed summaries generated using the R2 trained prediction model, as it leads to the best performance of all ROUGE metrics (see Table 8.2). This was also the case for linear regression trained prediction model in Chapter 7.

In the evaluation we asked three people to assess the summaries. Each person was shown 150 summaries (50 from each summary type selected randomly from the entire test set of 105 places). The summaries were shown in a random way. Each person was asked to assess the summaries according to the DUC and TAC manual assessment scheme described in Chapter 6. The results of the manual evaluation are shown in Table 8.5. Each cell of the table shows the percentage of summaries scoring the ranking score heading the column for each criterion in the row as produced by the summary method indicated by the subcolumn heading – no redundancy or sentence ordering ($wordLimit$ or short wL), redundancy (JRT) and JRT + sentence ordering ($JRTF$). The numbers indicate the percentage values averaged over the three people.

Table 8.6 shows percentage values of summaries which achieved scores at levels four or above. Each cell shows the percentage of summaries scoring the ranking score $>= 4$ for each criterion in the row as produced by the summary method indicated by column heading – no redundancy or sentence order ($wordLimit$), redundancy (JRT) and JRT + sentence ordering ($JRTF$). The numbers indicate the percentage values averaged over the three people. The fifth column shows the Wikipedia baseline results and the last column the results for the best performing system reported in Chapter 7.

Table 8.6: Readability evaluation results showing only the percentage values of summaries which achieved scores at levels four or above.

Criterion	wordLimit	JRT	JRTF	Wikipedia	isStarter + LMSim-2 + DepCat
clarity	47.9	95.9	94	94.3	85
coherence	25	71.5	82	90.7	74
focus	39.5	87.7	88	92.6	76.4
grammar	30.2	79.5	80	81.6	92
redundancy	12.5	69.4	92	91.5	83

We see from Table 8.5 that JRT type summaries perform better than the *wordLimit* setting where summaries are generated without redundancy or sentence ordering. The percentage values at levels 5 and 4 (see Table 8.6) show that the JRT summaries have more clarity (95.9% of the summaries), are more coherent (71.5% of the summaries), have better focus (87.7% of the summaries) and grammar (79.5% of the summaries) and contain less redundant information (69.4% of the summaries) than the ones generated in the $wordLimit$ setting (47.9%, 25%, 39.5%, 30.2% and 12.5%). Integrating sentence ordering ($JRTF$) as an additional jump in the A* search with JRT improves the summary quality further and leads to better readable summaries compared to the setting with only JRT. For the $JRTF$ setting the percentage values at levels 5 and 4 for coherence improve from 71.5% to 82% and for redundancy from 69.4% to 92%. For the other criteria (clarity, focus and grammar) the scores do not differ substantially: clarity for $JRTF$ is 94% (95.9% for JRT), focus for $JRTF$ is 88% (87.7% for JRT) and grammar for $JRTF$ is 80% (79.5% for JRT).

The substantial improvement in redundancy from JRT to $JRTF$ demonstrates that incorporating sentence ordering into a summarization system adds to redundancy reduction. Less redundancy means that the summaries contain more unique information than repetitions, and that should also be reflected positively in the ROUGE scores (as seen between the JRT and $wordLimit$ setting in Table 8.3). The results are shown in Table 8.7. The first column repeats results for the JRT setting and the second column shows results for $JRTF$. The last column shows results obtained using the best performing system reported in Chapter 7. Both the R2 and the RSU4 results are obtained with the prediction model trained using the R2 metric. For the first two columns A* search with MERT is used to train the prediction model, and for the last column linear regression was used (see Chapter 7). In this evaluation we used the entire set of 105 objects. From the results we can see that the integration of sentence ordering as additional jump in the A* search with the JRT setting lead to further slight improvement in R2 score compared to the JRT summaries and stayed stable in RSU4. The last column of Table 8.7 shows the results of the best performing system ($isStarter + LMSim - 2 + DepCat$) reported in Chapter

Table 8.7: Rouge Experimental results after integration of sentence ordering ($JRTF$).

Recall	JRT	JRTF	isStarter + LMSim-2 + DepCat	Wiki
R2	.109	**0.111**	.102	.097
RSU4	**.167**	**.167**	.155	.14

7. These results were obtained using the combination of $isStarter$, $LMSim-2$ and $DepCat$ features and also significantly outperformed both first document and Wikipedia baselines. The $DepCat$ feature that involves manual pre-processing had a positive effect on the results and played a very important role in getting significantly better results than the baseline summaries. We see from the table that the $isStarter + LMSim - 2 + DepCat$ results are significantly outperformed by the JRT and $JRTF$ ones ($p < 0.001$). This shows that both JRT and $JRTF$ settings are a better alternative than when the aim is to generate better entity-based summaries. The $JRTF$ summaries achieve moderately better $R2$ scores than the JRT summaries. When measured by the $RSU4$ metric, $JRTF$ summaries have the same score as JRT ones.

Finally, we compared the JRT and $JRTF$ summaries to the Wikipedia baseline in manual readability assessment, as the best system in Chapter 7, $isStarter + LMSim - 2 + DepCat$ was rated substantially below the strong Wikipedia baseline on all, except grammar, readability assessment criteria. Table 8.6 also shows that the JRT and $JRTF$ summaries are substantially closer to the Wikipedia baseline summaries than the summaries generated with the setting $isStarter+LMSim-2+DepCat$, in which redundancy and sentence ordering were addressed by manual dependency pattern categorization and controlled selection of categories. $JRTF$ for example achieves similar results in clarity and redundancy to the Wikipedia baseline. The coherence and focus are also substantially improved compared to the manual method, but there is still a gap to fill of 8% and 4% respectively in order to achieve the evaluation results of the strong Wikipedia baseline. The grammar score of $isStarter + LMSim - 2 + DepCat$ is higher than in all other settings (JRT, $JRTF$ and Wikipedia summaries). As discussed in Section 7.7.2 the lower grammar scores in Wikipedia articles are due to the non-standard characters used to describe how an entity is pronounced in other languages. Such non-standard characters were not properly displayed in the manual evaluation and therefore assigned lower grammar scores. This Wikipedia problem is inherited by the JRT and $JRTF$ summaries since they are derived from Wikipedia articles. The most common information type is the type definition (e.g. *Westminster Abbey is a church*). Such type definitions occur in almost every Wikipedia article as the first sentence. Such sentences also contain pronunciation information. If in a document set to be summarized there is a Wikipedia article then the JRT and the $JRTF$ models are likely to select a sentence from the Wikipedia article rather than from other non-Wikipedia articles

Table 8.8: Example summary for the Akershus Castle. This summary is generated using the setting JRTF.

Akershus Fortress (Norwegian: Akershus Festning) is the old castle built to protect Oslo, the capital of Norway. The fortress was built in 1299, and the meaning of the name is 'the (fortified) house of (the district) Aker'. The Akershus castle and fortress are located on the eastern side of the Oslo harbor. In the 1600s a castle (or in norsk, "slott") was built. The original Akershus Castle is located inside the fortress. The fortress was first used in battle in 1306. The fortress was reconstructed several times to withstand increasing fighting power. The fortress has successfully survived many sieges, primarily by Swedish forces. The fortress was strategically important for Oslo and therefore for Norway as well. Inside the fortress there is an information centre, Norways Resistance Museum and Akershus Castle and church. The fortress has survived many sieges, primarily Swedish, and, with the exception of WWII, never been conquered by a foreign enemy. The castle is well positioned overlooking Oslo's harbour. Akershus fortress is still a military area, but is open to the public daily until 9pm.

for the type definition. If that sentence contains pronunciation descriptions then the problem with non-standard characters also exists for the JRT and $JRTF$ summaries. Table 8.8 shows another summary about the *Akershus Fortress* generated using the $JRTF$ setting.

8.6.4 Discussion

In this chapter we presented and evaluated a fully automatic framework for addressing challenges of summary composition: informativeness maximization, redundancy reduction and sentence ordering. Posing this task as a search problem we show that it can be solved optimally and efficiently with all three challenges integrated. A further important feature of our proposed framework is that the A* search algorithm is run with the training of prediction model intact, in which the system parameters are learnt in such a way that the best scoring *whole summary* under the prediction model has a high score under the ROUGE metric. We train the prediction model using MERT discriminative training.

Our evaluation of this training and search strategy have revealed several important results.

Firstly, we demonstrated that training the prediction model on summaries leads to significantly better results when the model is trained on single sentences. This implies that when using the single sentence training approach there is only a guarantee for high content overlap between single training sentences and model sentences. However, when these sentences are combined into summaries, it is not guaranteed that these summaries will also have high content overlap with the entire model summaries. This suggests that it is important to train the prediction model on summaries – i.e. on the actual outputs for the summarizer – rather than on single sentences.

Second, our investigations into the influence of length constraints on output summary quality further support the assumption that input to the training needs to be constrained in the same way as the input to the actual working system is. When we used sentence-based length constraints (12 sentence limit) in training and generated summaries with the word limit of 200 words, the summaries obtained were significantly worse than the ones trained with the 200 word limit. This means that MERT does not adapt to different length constraints, so in order to obtain high ROUGE metric scores it is essential to use the same constraints in the training summaries as in the target ones.

Furthermore, we showed that the quality of the automated summaries can be further improved by reducing redundancy in them. To do this we evaluated two different redundancy reduction methods and demonstrated that both of them improve the ROUGE scores compared to the basic system without redundancy reduction. The best performing method was the JRT method, in which sentences are not included in the summary if their redundancy score exceeds an automatically learnt redundancy threshold R. We have seen that the properties of different ROUGE metrics require different redundancy thresholds, so that R must be learned for each ROUGE metric separately. The automatically determined R values appeared to be neither too strict nor too generous as they allow lexical repetitions such as referring expressions to be redundant in the output summary but not whole factual assertions.

Our human evaluation demonstrated that although addressing redundancy leads to improvement in summary quality there is still a gap to fill when the automated summaries are compared to the Wikipedia baseline results. To address this gap we addressed the sentence ordering problem within our framework, which led to more readable summaries. The main expectation from addressing sentence ordering in addition to redundancy within a single summarization framework is that the coherence of the summary will improve. The readability assessment confirms this, showing that the information type flow model helped to improve the summaries with respect to the coherence criterion when compared to the JRT summaries. When turning from JRT summaries to $JRTF$ ones there is also a substantial improvement with respect to the redundancy criterion. However, this substantial improvement in redundancy is not reflected in the ROUGE scores, as one might expect, which requires further investigation.

For summaries with sentence ordering addressed we obtained very similar results to the Wikipedia baseline summaries in four out of the five readability assessment criteria. This is consistent with our results from Chapter 7, where we have shown that the $DepCat$ sentence selection method, which also orders sentences within the summary, contributed most to summary quality. However, $DepCat$ orders the sentences within the summary using the dependency categories obtained manually. In this chapter we achieved better results with a fully automatic method for sentence ordering. However, our results indicate that there is still some room for improvement if automated summaries are to approach the Wikipedia baseline and further methods in sentence ordering are worth investigating for multi-document summarization.

ROUGE results suggest that achieving coherence does not happen at the cost of loosing important information from the summaries. A loss in information content could occur if information rich sentences were jumped over because the information types they convey are not highly likely according to the information flow model. Likewise, sentences could be selected only because they maximize the information flow, regardless of whether they are related to the topic the summary should be about. Our results demonstrate that these problems do not occur in our summarization framework. This may be because the summarization framework we use in this paper integrates all three challenges of automatic summarization: information content maximization, redundancy reduction and coherence optimization into a single framework in which the summarizer is trained to produce the best *whole summary*. This is consistent with what human authors do when they write texts – they organize content in a way that renders a well readable, coherent *whole* (Crossley & McNamara 2010). For this reason, it is important to address all three summarization tasks within a single framework. In this way our work departs from previous work on sentence ordering, which has mainly been done outside the summarization context (see Section 8.4.3). Applying these methods to real summarization problem would mean that summary worthy sentences are first chosen and then reordered in a post-hoc manner. This approach does not model the way people write texts. In particular it does not focus on creating the coherent whole summary as sentence selection works independently from sentence ordering. Therefore, we predict that this would render summaries judged as being lower in coherence than our approach which considers all aspects of summary generation simultaneously.

Finally, the manual and the automatic evaluations show that humans and ROUGE judge redundancy differently. From the example summary shown in Table 8.8 we can see that there are several noun phrase (NP) repetitions. Since ROUGE does not distinguish any word or phrase type it punishes any such repetition. However, it seems that humans allow such repetitions and assess them differently. Their main focus seems to be at repetitions on the information type rather on thee NPs as they assigned higher scores for $JRTF$ summaries than for the JRT ones.

In $JRTF$ summaries the repetition of same information types is reduced due to the ordering behaviour.

Currently, the information flow model is based on information about transitions between pairs of information types. Ideally, one would like to model longer sequences of information types such as tri-grams or even longer sequences. Furthermore, the current approach seeks to maximize the transition probability of information types in adjacent pairs of sentences independently, as opposed to maximizing the probability of the overall sequence of information types. Ideally, we should consider the information type flow from the first sentence in the summary till the last one and take the summary that has the highest flow probability. Addressing each of these weaknesses should lead to more global coherence of summaries, though it is not clear how or whether such considerations can be integrated into an A* search summary composition process.

Our current approach uses add-one smoothing which is not very sensitive to data sparsity. We plan to experiment with methods which are more sensitive to data sparsity such as Good-Turing estimation (Jurafsky & Martin 2008) described in Section 7.2.

In addition, we do not consider grouping of similar information types. If we place similar information types or dependency patterns into the same group, then this may help the problem with data sparsity. We can illustrate this with the following example. Let us assume we have a group X containing three similar information types x_1, x_2, x_3 and another group Y with the information types y_1, y_2. Let us further assume that the last sentence in the summary contains only the information type x_1, and the candidate sentence only the information type y_1. We further assume that our information type model does not contain the pair $< x_1, y_1 >$ but pairs like $< x_1, y_2 >$, $< x_2, y_1 >$, $< x_3, y_1 >$, $< x_2, y_2 >$ and $< x_3, y_2 >$. If we do not have the groups, we are forced to assign the smoothing probability score to the pair $< x_1, y_1 >$. However, if groups are available, we could take the probability of one of the other pairs. We could also take the pair that has the highest probability. This may not only address the data sparsity problem but also lead to better scoring summaries since any sentence jumped over is more informative than the next one (as noted earlier we order the sentences by sentence score divided by their length and visit them in descending order). If we include the candidate sentence in the summary instead of the next one, we would achieve better scoring summaries according to our summarization model (see Section 8.1). Therefore, we plan to investigate ways of grouping information types.

In the human evaluation we have seen that information type flow models help to reduce redundancy. Since the same information type is less likely to follow itself in this way the repetition of the same information type in the summary is reduced. However, the ordering is achieved between pairs of information types of two sentences which lead to a most likely flow. Other information types within these sentences are not considered. Because of this constraint repetitions are not entirely captured. We plan to investigate this further and use in additional to the n-gram similarity approach information types for redundancy reduction. We will allow the inclusion of a sentence only if it brings novel information types to the summary – i.e. similar to the n-gram similarity approach we will measure the overlap of information types of the candidate sentence and the summary and include the sentence in the summary if the overlap is below a threshold.

Finally, in our summarization approach we do not perform spelling correction. As one can see the example summaries shown in Tables 8.4 and 8.8 have sentences containing misspellings. Correcting misspellings will indeed help to achieve better summary quality. Thus we plan to address this issue as well.

8.7 Summary

In this chapter we presented and evaluated a fully automatic framework for addressing challenges of summary composition: informativeness maximization, redundancy reduction and sentence ordering. Posing this task as a search problem we showed that it can be solved optimally and efficiently with all three challenges integrated. A further important feature of our proposed framework is that the A* search algorithm is run with the training of the prediction model intact, in which the system parameters are learnt in such a way that the best scoring *whole summary* under the prediction model has a high score under an evaluation metric. We demonstrate that training the prediction model with summaries leads to significantly better results than when the model is trained on single sentences. We trained the prediction model using MERT discriminative training. Furthermore, we showed that it is vital to have the same constraints in training as in testing. We showed that our fully automatic summarization system outperforms the one proposed in the previous chapter which involved substantial manual effort. The main improvements in both cases came from addressing sentence ordering within the summary. Although our results from human readability evaluation showed significant improvement compared to the manual system, there is still scope for improvement with respect to the Wikipedia baseline. However, both ROUGE and readability assessment show that the summarization framework we propose in this chapter is a viable solution for creating good automated summaries.

CHAPTER 9

Conclusion and Future Work

9.1 Summary

In this work we have investigated the application of entity type models in multi-document summarization using automatic caption generation for geo-located entity images as an application scenario. Entity type models are automatically derived from texts about entities of the same type and contain sets of patterns that aim to capture the ways the entities are described in natural language. We experimented with three different representation methods for entity type models: signature words, n-gram language models and dependency patterns.

To implement the idea of entity type modeling within multi-document summarization we first investigated which information types (attributes) humans associate with geo-located entities from urban and rural landscape through a Mechanical Turk survey. In our survey we found that there are attributes which are relevant for any entity type but also those specific to particular entity types. However, even within the set of attributes shared between different entity types, not all of attributes were equally important for each entity type. We aimed to find similar entity types, i.e. entity types that share attributes and where importance ranking of attributes is equal. We used popularity ranking of the attributes to measure similarity between attributes. The results show that entity types which are similar in function, design and look are also similar in the ranking of their attributes.

Based on these results we analyzed whether the attributes identified in the survey are also present in existing descriptions of geo-located entities. We have shown that attributes identified in the survey are also found in human generated Wikipedia articles about geo-located entities. Therefore we used Wikipedia articles to construct entity type corpora, each corpus being a collection of Wikipedia articles about geo-located entities of specific types. To do this we implemented *Is-A* patterns to automatically categorize Wikipedia articles by different entity types. In total we found 107 different geo-location related entity types leading to 107 different entity type corpora. The accuracy of assigning Wikipedia articles to particular entity types was 91%. We also discussed similar entity type corpora reported by related work and have shown that our entity types are more specific than these alternatives, which makes them more suitable for our summarization task.

For the summarization task we developed an extractive, query-focused multi-document summarizer which we use to automatically generate summaries for each geo-located entity. The summary is extracted from input web documents related to that entity. The input documents were queried from the web using the name of the entity as query. For each entity we used two baseline summaries: text extracted from the top ranking web-document and also from the Wikipedia article about that geo-located entity.

To evaluate the quality of the automatic summaries generated using our summarization system we collected a corpus of model summaries. In total 307 geo-located entities were used and up to four model summaries were manually created for each entity. The model summaries were generated from existing geo-located entity descriptions provided by VirtualTourist, an online travel site. We showed that the quality scores of our model summaries are comparable to those reported in DUC and TAC for their model summaries.

Central to our approach is the use of entity type models in multi-document summarization. To implement this idea we extended our summarization system with an additional entity type model feature and compared its performance to that of the summarizer that uses only standard features. We investigated signature words, n-gram language models and dependency patterns as approaches to derive entity type models from the entity type corpora. Our evaluations showed that dependency patterns yield summaries that score more highly than approaches that use a simpler representation of an entity type model in the form of a n-gram language model or signature words. When used as the sole feature for sentence scoring, dependency pattern models (*DpM-Sim*) produced summaries with higher ROUGE scores than those obtained using the standard text features and the n-gram and signature word models. These dependency pattern models also achieved a modest improvement in ROUGE SU4 over the Wikipedia baseline.

We used a linear weighted sum of features as a method to combine the scores of each separate feature. To train weights we used linear regression in which single sentences with feature and sentence salience scores measured by ROUGE were used to guide the training. However, we claimed that it is better to train the feature weights using whole summaries rather than single sentences and that therefore it is important to combine the summarization process with the training. We proposed a framework that performs the search for the best summary with the training intact. This means that feature weights are learned on whole summaries in an iterative way. In the first iteration random feature weights are used. Using these initial weights the search then generates the best summaries. Based on these best summaries the feature weights are modified in the training step and new weights are output. This process of generating new features and best summaries continues until the feature weights do not change. For searching for the best summary we proposed an admissible A* search heuristic. The A* search algorithm creates the best summary without violating the summary length limit. For training we used MERT, a discriminative training approach, that uses the output of A* search and trains the weights. We showed that such an integrated framework leads to significantly better results than when the feature weights are obtained based on single sentences.

We also used the dependency patterns (as *DepCat* feature) to reduce redundancy within the summary and to improve sentence ordering. We manually categorized the dependency patterns by six different information types. Using these patterns and their information types we determined the type of information the input sentences contain during the summarization process. In the summary composition we included for each of the six information types a restricted number of sentences in the summary. By doing this we controlled the type of information in the summary and aimed to reduce the redundancy within it. In addition, we also ordered the sentences within the summary using a sentence flow derived from manually created summaries. For this we ordered the information types in the same way in which they appear in model summaries. The evaluation results showed that summaries produced using the *DepCat* feature have less redundancy and were more coherent than those generated without the *DepCat* feature.

These results indicated that the way dependency patterns can be used to structure information or perform sentence ordering within the summary deserves further attention. Therefore, we explored a further way of using dependency patterns for redundancy reduction and sentence ordering and proposed a fully automated approach to addressing these tasks. To reduce redundancy within the summary we proposed an admissible A* search heuristic that is based on the idea of jumping over redundant sentences. During the summary composition (search) the heuristic verifies the inclusion of the current sentence within the summary by checking the similarity between the sentence and the existing summary. If the sentence is too similar, i.e. the similarity score is above an automatically learned threshold, then the heuristic makes sure that the sentence is excluded; otherwise it includes the sentence in the summary. To address the sentence ordering problem we extended this heuristic with an additional conditional jump. The extension makes sure that the heuristic jumps over the current sentence if it is less likely to follow the last sentence in the existing summary than the following sentence (sentence that is analysed after the current sentence). We derived the transition likelihood values using the dependency patterns and the entity type corpora.

Our results show that the automatic approach for addressing both redundancy and sentence ordering problems within the summary is superior to the approach where manual categorization is required. In both ROUGE and manual readability evaluations we obtained better results using the automatic approach than the manual approach. The results also show that the automatic approach led to very similar readability evaluation results as the Wikipedia baseline summaries.

The novelty and contributions of our proposed ideas and research steps are outlined and discussed in the following section. While we experimentally investigate and offer solutions for a number of issues related to entity type model development and its use in multi-document summarization, this work leaves open a number of avenues to pursue in future work (cf. Section 9.3).

9.2 Contributions

As mentioned in Section 9.1 in this work we investigated the application of geo-located entity type models in multi-document summarization using automatic caption generation for geo-located entity images as an application scenario. Our investigation was driven by several hypotheses (cf. Section 1.3) which we analyzed in different experiments. Addressing these hypotheses has lead to a number of novel contributions regarding how to construct entity type models for geo-located entities and apply them in multi-document summarization. In the following we discuss the main findings regarding each of our hypotheses with respect to their novelty and contribution and also explain the practical applications of this work.

9.2.1 Attributes for Geo-located Entity Types

Our first research hypothesis was that geo-located entity types are characterized by sets of typical attributes, some of which are entity type specific and others are shared between entity types or concepts. Cognitive psychology has gathered substantial experimental evidence for the existence of concepts, i.e. categories in which entities from our natural and built environment can be placed. Concepts are characterized by sets of attributes and theories differ in whether they claim that a set of attributes is necessary and sufficient to define a concept (defining-attributes theories) or whether the concepts are more fuzzy in their specification in terms of attributes (prototype theories), so that some entities are more representative of a concept than others (Eysenck & Keane 2005).

Based on this research we can assume that there will be sets of attributes for geo-located entity concepts or entity types and that some of these attributes will be specific, while others are more general and shared between entity types. For example, it could be easily assumed that an attribute such as *location* will be relevant to many entity types, e.g. *church, bridge, lake*, etc., while the event *eruption* is specific to volcanos. However, we could refer to no explicit previous study conducted with humans to identify which attributes exactly are relevant for which entity type and which are specific or shared. A step in this direction, however, is research on ontology matching, which has studied the similarity between entities to some extent (e.g. (Rodríguez & Egenhofer 2003)). This and similar studies tackle the problem of measuring the similarity between two entity types or ontologies. The similarity is computed based on different information including attributes as in our Mechanical Turk survey. However, their main concern is aligning existing ontologies whereas we are interested in identifying the set of attributes relevant for describing an entity, not only in their similarity. Without the set of attributes the matching based on attributes is not feasible. Also the authors use only 12 geo-located entity types in their study, whereas we used 40 different classes, which gives our analysis more empirical strength. Another relevant research direction is the idea of learning ontologies from text resources. In these studies, similar to information extraction, relevant features for a given entity type are extracted (Sabou et al. 2005, 2008). However, none of these studies analyse the problem of extracting attributes for geo-located entities and then using them for similarity computation. Therefore, our investigation of the types of information humans associate with geo-located entity types contributes further empirical support for understanding what information humans associate with geo-located entities of different types.

The results of our online experiment conducted on Mechanical Turk lead us to retain our research hypothesis and conclude that there indeed exist sets of attributes that describe geo-located entity types (Section 4.3.3). We found that some of these attributes were specific to certain entity types, while others were shared between many types. However, even in the set of shared attributes, we found a variation in popularity between the attributes for each entity type, showing that some attributes are more strongly related to some entity types than to others.

We used the results of our Mechanical Turk evaluation to guide the creation of entity type models for groups of entity types. Creating entity type models for groups of entity types can be necessary if there are no resources related to a particular entity type from which entity type models can be derived. Our results showed that entity types correlate highly when they are similar in purpose, look or design (e.g. churches and temples, rivers and lakes, etc.) (Section 4.3.2). We used this result to group entity types into groups and created entity type models for these groups. Our evaluation showed that using entity type models derived from the resources related to the group and using these models in summarization leads to a small decrease in summary quality compared to the case where entity type models are derived from corpora for single entity types (cf. Table 7.10, Section 7.8). Therefore, entity type modeling for groups of entity types is a viable solution in cases where textual resources are scarce.

Another possible application of the findings of our experiment could be the automatic generation of IE style templates as reported in Sudo et al. (2003), Sekine (2006), Filatova et al. (2006), Etzioni et al. (2008), Banko & Etzioni (2008), Li et al. (2010). These templates are descriptions of types of information relevant to a specific domain, with domains such as different events reported in the news (e.g. terrorist attacks, plane crashes, etc.) or person specific information. Currently, there exist no templates for descriptions of geo-located entities, and the questions/attributes collected in the survey could be used as such templates for entity types. For each entity type the information types found in our Mechanical Turk experiment could be used as the entire set of attributes for which an IE system has to find values. This would contribute to populating knowledge data bases with geo-located entity information, which could be used to index images pertaining to geo-located entities. If information relevant for humans is used for indexing, this could lead to better retrieval and organization of those images. In addition, templates can be used in template based automatic summarization, e.g. for purposes of automatically generating descriptions for images showing geo-located entities. Furthermore, the relevant information types could be used in a guided summarization task as organized by DUC and TAC. In this task the automated summaries could aim to answer the questions grouped by the attributes relevant for geo-located entities.

9.2.2 Creating Entity Type Models

Our second hypothesis was that the attributes identified as relevant for characterizing geo-located entities in the experiment will also be found in human generated texts, so that these texts are a good potential resource for deriving entity type models. We called such text resources entity type text corpora. To the best of our knowledge similar investigations do not currently exist, so we used the results of the previous analysis about the entity type attributes to analyse whether the attributes identified as relevant are also present in entity type corpora. We considered Wikipedia articles about geo-located entities for this analysis, since they contain high quality human generated descriptions of geo-located entities. We showed that information types found in Wikipedia articles of a specific type correlate with those found in our Mechanical Turk evaluation for the same type (Section 5.1). We could therefore retain our hypothesis and conclude that Wikipedia can be used for learning/capturing the ways how attributes of different entities of the same type are described.

Once we empirically established that Wikipedia is a suitable resource for deriving entity type corpora, we proposed a method for acquisition of entity type corpora from Wikipedia using Is-A patterns. We have shown that using these patterns it is possible to obtain high quality entity type corpora (cf. Table 5.3, Chapter 5). Related work in categorization of Wikipedia articles has offered several taxonomies, none of which was sufficiently detailed to identify geo-located entity corpora (cf. Section 5.2.3 for discussion). Using our approach we obtained entity type corpora for many specific entity types occurring in real life. In this way we have contributed, and have evaluated, a domain independent method for categorization of Wikipedia articles. Similar *Is-A* patterns can be used to obtain entity type corpora for other domains. As in our work, entity type corpora can be used for the derivation of entity type models about particular entity types, which can then be further used in entity focused automatic summarization.

9.2.3 Entity Type Models for Sentence Scoring and Summary Composition

Our first hypothesis regarding the use of corpus-derived entity type models in multi-document summarization was that entity type modeling improves sentence scoring. To test this hypothesis we have investigated three different ways to create entity type models from entity type corpora: signature words, n-gram language models and dependency patterns. Each method was used within a summarizer to score those sentences that are salient according to the entity type model more highly than those which are not. We have shown that the use of entity type models for scoring the input sentences indeed helps to significantly improve summary quality over using standard text related features (cf. Table 7.4, Chapter 7). Among the three models, the dependency patterns proved superior to both signature words and n-gram language models as entity type models. This is most likely due to their ability to capture long distance dependencies between terms.

Although the idea of using entity type models in sentence scoring existed in previous work (Biadsy et al. 2008), a systematic evaluation and comparison of different modeling methods with respect to their integration into the summarizer and impact on the resulting summary quality was missing. For example, Lin & Hovy (2000) used signature words and Nenkova et al. (2006) n-gram language models to model the topics within news document collections and used them to perform sentence scoring. Dependency patterns have been described in the work of Nobata et al. (2002) who uses them in single document summarization. However, the authors themselves do not report a systematic evaluation of different modeling methods. Nobata et al. (2002), for example, use dependency patterns in combination with other standard summarization features to score sentences. However, they do not report any performance comparison between these different methods, nor do they mention any positive impact of the patterns on the summary quality. Nenkova et al. (2006) on the other hand use the n-gram language models for a different purpose. In their work n-gram language models are used to test the contribution of word frequency to the summary quality, rather than being used as a domain representation to be tested on unseen domain-related documents. In contrast, we use n-gram language models to capture domain knowledge and use them to generate summaries from unseen documents belonging to the same domain.

By investigating the impact of these different ways of entity type modeling on the quality of automatic multi-document summaries we have made a novel contribution to the field of multi-document summarization.

Unlike the previous work outlined above we applied the three methods for entity type modeling to a domain of geo-located entity descriptions instead of news and used a substantially larger data set for deriving entity type models. For example the corpus for the entity type church contains ∼3000 Wikipedia articles about different churches around the world (e.g. *Westminster Abbey*, *Sagrada Familia*, etc.). In total we use corpora for 107 entity types (cf. Chapter 5, Section 5.2.1). Related work used relatively small number of topics (4 used by Lin & Hovy (2000), 50 used by Nenkova et al. (2006) and 30 used by Nobata et al. (2002)), as well as small number of documents related to each topic (around 100 topic relevant articles in Lin & Hovy (2000) and 10 articles for each topic in Nenkova et al. (2006) and Nobata et al. (2002)). We also compared all three entity type modeling methods on the same data set, so our results offer a clear indication of their relative performance. For this purpose we have collected a corpus of 937 model summaries for 307 different entities. We have shown that the quality of the model summaries is high and comparable to that of summaries provided by DUC and TAC (Table 6.5, Chapter 6). We have made these model summaries freely available[1] to provide evaluation data to others working in the same area.

[1] The summaries can be downloaded from http://staffwww.dcs.shef.ac.uk/people/A.Aker/modelSummaries.rar

Using a greater number of topics, larger sets of articles for each topic and a test corpus of substantial size enables more reliable conclusions to be drawn about a method's general feasibility. Our figures for the topic count and size of the articles in each topic allow an estimation of how well these three entity type modeling methods would perform generally. Our techniques for deriving entity type models and integrating them into a summarizer are suitable not only for image captioning but for any application context that requires summary descriptions of instances of entity classes, where the instance is to be characterized in terms of the features typically mentioned in describing members of the class.

Our second hypothesis regarding the use of entity type models in summarization was that entity type models can be used for redundancy reduction and improving the sentence order within a summary. We have shown that entity type models, in particular dependency patterns, can be used to help to reduce redundancy within a summary and to improve its sentence order. We have shown this for two different approaches. In our first approach we manually categorize the dependency patterns into different information types such as *entity type*, *foundationyear*, *location*, *specific*, *surrounding* and *visiting*. We use these pattern categories to control the type of information to be included into the summary, by which we aim to reduce the redundancy. The ordering of these categories in model summaries was used to manually determine the ordering of the same categories within the output summary and in this way perform sentence ordering.

Our second approach is fully automatic and requires no manual categorization of patterns, nor decisions about types and order of categories for the output summaries. Instead, to reduce redundancy within the summary we proposed a new admissible A* search heuristic based on the idea of jumping over redundant sentences. When A* search composes the summary it always checks whether the current sentence is includable in the summary or too similar to sentences already included in the summary. To decide what to include and what to exclude we used a global summary similarity threshold. Unlike related work (Barzilay et al. 1999, Lin & Hovy 2002, Saggion 2008, Sauper & Barzilay 2009) which sets such a threshold manually, we learn it automatically using MERT. To deal with the sentence ordering, we further extended this heuristic, so that it not only avoids redundant sentences but also those which are less likely to follow the last sentence in the current summary. To compute sentence transition likelihoods we use the entity type models, in particular the dependency pattern representation of the entity type corpora, and compute the frequency counts of dependency pattern sequences in entity type corpora. In the summarization process we use these counts to estimate the transition probability for two different sentences, the last sentence in the summary and the sentence under inspection for inclusion in the summary in order to decide whether the sentence under inspection is likely to follow the last sentence in the summary or not.

We have shown that our second, automatic approach is superior to the first manual approach. In both ROUGE (cf. Table 8.7, Chapter 8) and manual evaluation (cf. Table 8.6, Chapter 8) we obtained significantly higher scores using the automatic approach than the manual one. In addition, the human readability evaluation suggested that the output summaries are very close to the strong Wikipedia baseline (cf. Table 8.6, Chapter 8).

The use of dependency patterns for redundancy reduction and sentence ordering is novel in the field of multi-document summarization. This work's contributions are two-fold. First, we demonstrate that these aspects of automatic summarization can be successfully addressed using entity type models represented as dependency patterns. Our evaluation results demonstrate that the main contribution of dependency entity type models seems to be in redundancy reduction and in particular in sentence ordering. We obtained the best summarization scores when entity type models were used for these purposes, in both the manual and the fully automatic systems. We are not aware of any work which addresses redundancy and sentence ordering using dependency pattern modeling. Second, we offer a fully automatic framework for addressing these summarization challenges simultaneously. No previous work integrates redundancy and sentence ordering criteria in their search process and thus tackles them simultaneously within a single, fully automated framework, as we do by incorporating redundancy and sentence ordering constraints into the A* search.

Our analysis and the proposed full framework with integrated redundancy and sentence ordering can be used as a starting point for an improved framework that fully integrates all summarization criteria. Such an improved framework could, for instance, use different similarity measures to tackle the redundancy problem. This could be done by incorporating semantic information into the computation of the similarity between the summary so far and the candidate sentence to be included in it. Results may be further improved if techniques to detect different expressions with the same meaning are also used (Resnik 1995, Hatzivassiloglou et al. 2001, Erkan & Radev 2004).

9.2.4 Integration of Training and Search

Our final hypothesis in this work was that the integration of training and search in summarization can improve summary quality over methods in which these two steps are decoupled. The idea of integrating training and search is novel, as no currently existing related work appears to use search and training simultaneously to generate multi-document summaries. We have proposed an integrated summarization framework where search and training are used simultaneously to generate a multi-document summary. For the search we have proposed an admissible A* search heuristic that creates for each topic the *n* best summaries. In training we used a discriminative method MERT to learn the feature weights of our extractive summarization system. MERT uses the *n* best summaries to update the feature weights. In our evaluations we have shown that this integrated framework significantly outperforms settings where the training and search are decoupled (cf. Table 8.2, Chapter 8). Furthermore, we highlighted the importance of uniformity in training and testing and argued that if the goal is to generate high scoring summaries under a length limit in testing, then the same constraint should also be used in training.

We used our framework to generate geo-located entity descriptions. However, it can be used in other summarization tasks involving different domains. The only requirement is the availability of model summaries, input documents to be summarized and a summarization feature extraction tool. For instance, in Di Fabbrizio et al. (2011) our framework was used to perform sentiment summarization for hotel reviews. The performance of our framework was significantly better than that achieved with MEAD a competitive single and multi-document summarization system (Radev et al. 2001).

9.2.5 Applications

The summarization framework developed in this work presents a working summarization system which can be used for automatic image captioning or caption augmentation. Automatic captions can be used for applications where users seek information about a place just by taking its picture. Alternatively, these more substantial location image descriptions can be used to improve image indexing and search. We have explored both these application scenarios in related collaborative projects.

The contribution of our summarization system to indexing and search has been investigated in Aker, Fan, Sanderson & Gaizauskas (2012). In this work we evaluated image retrieval effectiveness contrasting conditions when captions generated by our summarizer are used to index images and when existing image captions found in Flickr are used. The generated captions were evaluated by user assessments and subjective measures. The best results were achieved when our summaries were combined with existing keyword captions, i.e. Flickr captions. This indicates that our image captioning technique by multi-document summarization is most useful for image retrievability when employed to augment image descriptions.

The idea of using the summarizer as an application for smart phones which automatically generates image descriptions for an image of any place taken using the phone has been explored in a 3rd year computer science student project.[2] This application would be of interest for all users seeking to find information about a geo-located entity just by taking its picture. It was shown that using our summarization system for this purpose is a viable option.

9.3 Future Work

In Chapter 4 we performed comparison between the attributes of different single entity types in order to find similar entity types. The comparison was performed using Kendall's Tau correlation coefficient. We also demonstrated (Section 7.8) that entity type models derived for grouped types still improve the performance of summary generation for a single geo-located entity. This can be very useful when there is a geo-located entity for which no or not enough textual resources are available. In this case text resources of similar entity types could be used to derive an entity type model for that type. Therefore, one avenue for future research is to identify groups of entity types based on their attributes. In this work we performed the grouping manually. In future work, we will use the results of our comparisons to perform this grouping automatically. One way to do this would be to use the K-nearest neighbors machine learning method that puts similar entity types as measured by the Kendall's Tau to the same group.

We also plan to investigate the use of the entity type related questions gathered through our Mechanical Turk survey to evaluate the summaries. Each automatically generated summary about an entity, could be presented to a user along with a set of questions related to the frequently asked attributes about the entity type. The user would be asked to indicate whether the summary answers those questions. This would be similar to the Content Sentence Units (CSU) presented within the Pyramid method (Nenkova et al. 2007). In the Pyramid method, however, the CSUs are extracted from human generated model summaries and presented to the evaluator along with automated summaries. The evaluator is asked whether the automated summary contains that CSU. In our case, the process of extraction of CSUs from reference summaries would be redundant, as the CSUs would be replaced by the questions related to frequently occurring attributes.

In Chapter 5 we have used *IS-A* patterns to collect geo-located entity type corpora from Wikipedia. We aim to investigate how the same approach can be used to collect corpora in other domains

[2] http://www.androidapps-home.com/photo-captioner-android-2793861.html

such as persons. Such corpora can be used to learn patterns about persons with different professions such as musicians, politicians, scientists and the learned patterns can be used to guide, e.g., a summarization system for biography generation.

In Chapter 8 we discussed the information type flow models. We plan to use these models to generate infoboxes for geo-located Wikipedia articles which do not have infoboxes or extend existing ones with further entries. Based on these models we could select a set of sentences which according to the model build the best flow of information types. The subset of sentences can be either taken from the Wikipedia article or from documents retrieved from the web. Finally, instead of outputting the sentences we could output the patterns that caused the particular sentence order to happen and use them to populate the Wikipedia infobox.

The patterns can be used to find sentences either within the Wikipedia article or from documents retrieved from the web. Sentences talking about the same information type, e.g. location of an entity, can be further processed

In Section 8.6.4 we have discussed several points we would like to improve within our integrated summarization framework. One of the improvements relates to sentence ordering. We aim to investigate alternative methods for modeling the transition between the sentences in the summary and produce more coherent summaries. In addition, we plan to group similar information types into the same group and use the groups in our sentence ordering approach.

Finally, we also aim to extend the previous work on image caption generation (e.g. Kojima et al. (2008), Yao et al. (2010), Yang et al. (2011), Li et al. (2011)) using entity type modeling of entities and spatial relations extracted from the image, which would allow us to caption a wider set of images, not only images of geo-located entities. In caption generation the idea is to generate natural language sentences based on semantic representations of visible objects and spatial relations extracted from the image (“visual predicates”). The common generation process is to link the extracted named objects with terms from a thesaurus, compose the terms into a natural language sentence and output the most likely sentence as an image description. Like related work, we would start with a set of visual objects and relation predicates extracted from the image (though in our case we would have an n-best list of these). However, unlike related work, which uses thesauri-like resources or simple n-gram modeling, we propose to use entity type models derived from large collections of on-line textual descriptions of entities or scenes, using dependency parsing and statistical clustering techniques.

Furthermore, we can also model the interaction between different objects and between objects and scene types. These models could help us to select from the various alternatives supplied by the image analysis components and would also allow us to supplement the visual objects and scenes extracted from the image with more abstract attributes, such as the actions an object might perform in a setting (e.g. we might infer “walking” as the action given a person and dog and a beach). In addition to this, the models could help us to identify the correct relationship between the objects and objects and scenes. This relationship information could be used when constructing the natural language sentences. For instance, the sentence *A dog is taking a person for a walk on a beach* will not be generated because dogs generally do not take persons for a walk but a persons do walk dogs. Moreover, this relationship information could help us to distinguish between main objects and peripheral objects in the image (in an image of a beach scene a dog may be smaller than a rock, but a caption like “Person walking dog on beach” is likely to be more appropriate than “Person, dog and rock on beach”). Based on this distinction a more appropriate and richer caption could be generated, optionally containing mentions of the peripheral objects in relation to the main object(s).

APPENDIX A

Different Heuristics Used in A* Search

We present three different heuristics of increasing fidelity, that is, that bound the cost to a goal state more tightly. Algorithm 5 is the simplest, which simply finds the maximum score per word from the set of unused sentences and then extrapolates this out over the remaining words available to the length threshold. In the algorithm, we use the shorthand $s_n = \phi(x_n) \cdot \lambda$ for sentence n's score, $l_n = \text{length}(x_n)$ for its length and $l_{\mathbf{y}} = \sum_{n \in \mathbf{y}} l_n$ for the total length of the current state (unfinished summary).

Algorithm 5 Uniform heuristic, $h_1(\mathbf{y}; \mathbf{x}, L)$

```
Require: $\mathbf{x}$ sorted in order of score/length
$n \leftarrow \max(\mathbf{y}) + 1$
return $(L - l_{\mathbf{y}}) \max\left(\frac{s_n}{l_n}, 0\right)$
```

The h_1 heuristic is overly simple in that it assumes we can 'reuse' a high scoring short sentence many times despite this being disallowed by the model. For this reason we develop an improved bound, h_2, in Algorithm 6. This incrementally adds each sentence in order of its score-per-word until the length limit is reached. If the limit is to be exceeded, the heuristic scales down the final sentence's score based on the fraction of words that can be used to reach the limit.

The fractional usage of the final sentence in h_2 could be considered overly optimistic, especially when the state has length just shy of the limit L. If the next best ranked sentence is a long one, then it will be used in the heuristic to over-estimate of the state. This is complicated to correct, and doing so exactly would require full backtracking which is intractable and would obviate the entire point of using A* search. Instead we use a subtle modification in h_3 (Alg. 7) which is equivalent to h_2 except in the instance where the next best score/word sentence is too long, where it skips over these sentences until it finds the best scoring sentence that does fit. This

Algorithm 6 Aggregated heuristic, $h_2(\mathbf{y}; \mathbf{x}, L)$

```
Require: $\mathbf{x}$ sorted in order of score/length
$v \leftarrow 0$
$l' \leftarrow l_{\mathbf{y}}$
for $n \in [\max(\mathbf{y}) + 1, k]$ do
    if $s_n \leq 0$ then
        return v
    end if
    if $l' + l_n \leq L$ then
        $l' \leftarrow l' + l_n$
        $v \leftarrow v + s_n$
    else
        return $v + \frac{l_n}{L - l'} s_n$
    end if
end for
return $v$
```

helps to address the overestimate of h_2 and should therefore lead to a smaller search graph and faster runtime due to its early elimination of dead-ends.

Algorithm 7 Agg.+final heuristic, $h_3(\mathbf{y}; \mathbf{x}, L)$

```
Require: $\mathbf{x}$ sorted in order of score/length
$n \leftarrow \max(\mathbf{y}) + 1$
if $n \leq k \wedge s_n > 0$ then
    if $l_{\mathbf{y}} + l_n \leq L$ then
        return $h_2(\mathbf{y}; \mathbf{x}, L)$
    else
        for $m \in [n + 1, k]$ do
            if $l_{\mathbf{y}} + l_m \leq L$ then
                return $s_m \frac{L - l_{\mathbf{y}}}{l_m}$
            end if
        end for
    end if
end if
return 0
```

The search process is illustrated in figure A.1. When a node is visited in the search, if it satisfies the length constraint then all its child nodes are added to the schedule. These nodes are scored with the score for the summary thus far plus a heuristic term. For example, the value of 4+1.5=5.5 for the $\{1\}$ node arises from a score of 4 plus a heuristic of $(7 - 5) \cdot \frac{3}{4} = 1.5$, reflecting the additional score that would arise if it were to use half of the next sentence to finish the summary. Note that in finding the best two summaries the search process did not need to instantiate the full search graph.

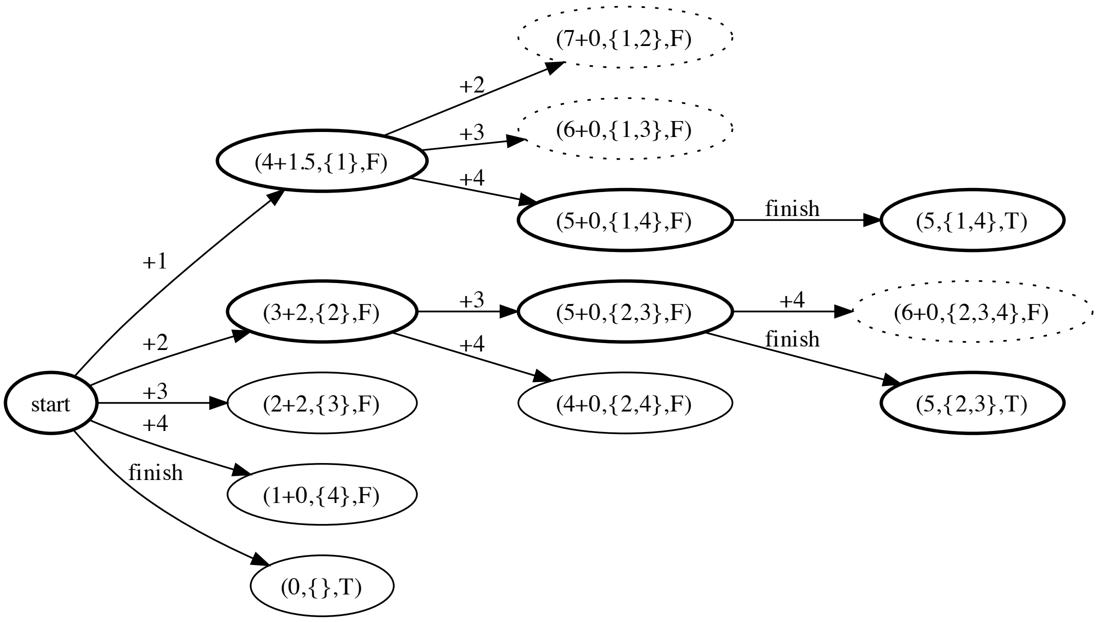


Figure A.1: Example of the A* search graph created to find the two top scoring summaries of length $\leq$ 7 when summarising four sentences with scores of 4, 3, 2 and 1 respectively and lengths of 5, 4, 3 and 1 respectively. The h_1 heuristic was used and the score and heuristic scores are shown separately for clarity. Bold nodes were visited while dashed nodes were visited but found to exceed the length constraint.

To test the efficacy of A* search with each of the different heuristic functions, we now present empirical runtime results. We used the training setting as described in Section 8.5.2 and for each entity or document set generated the 100-best summaries with word limit $L = 200$. Figure A.2 shows the number of nodes and edges visited by A* search, reflecting the space and time cost of the algorithm, as a function of the number of sentences in the document set being summarized. All three heuristics show an empirical increase in complexity that is roughly linear in the document size, although there are some notable outliers, particularly for the uniform heuristic. Surprisingly the aggregated heuristic, h_2, is not considerably more efficient than the uniform heuristic h_1, despite bounding the cost more precisely. However, the aggregated+final heuristic, h_3, consistently outperforms the other two methods. For this reason we have used h_3 in our experiments in Chapter 8.

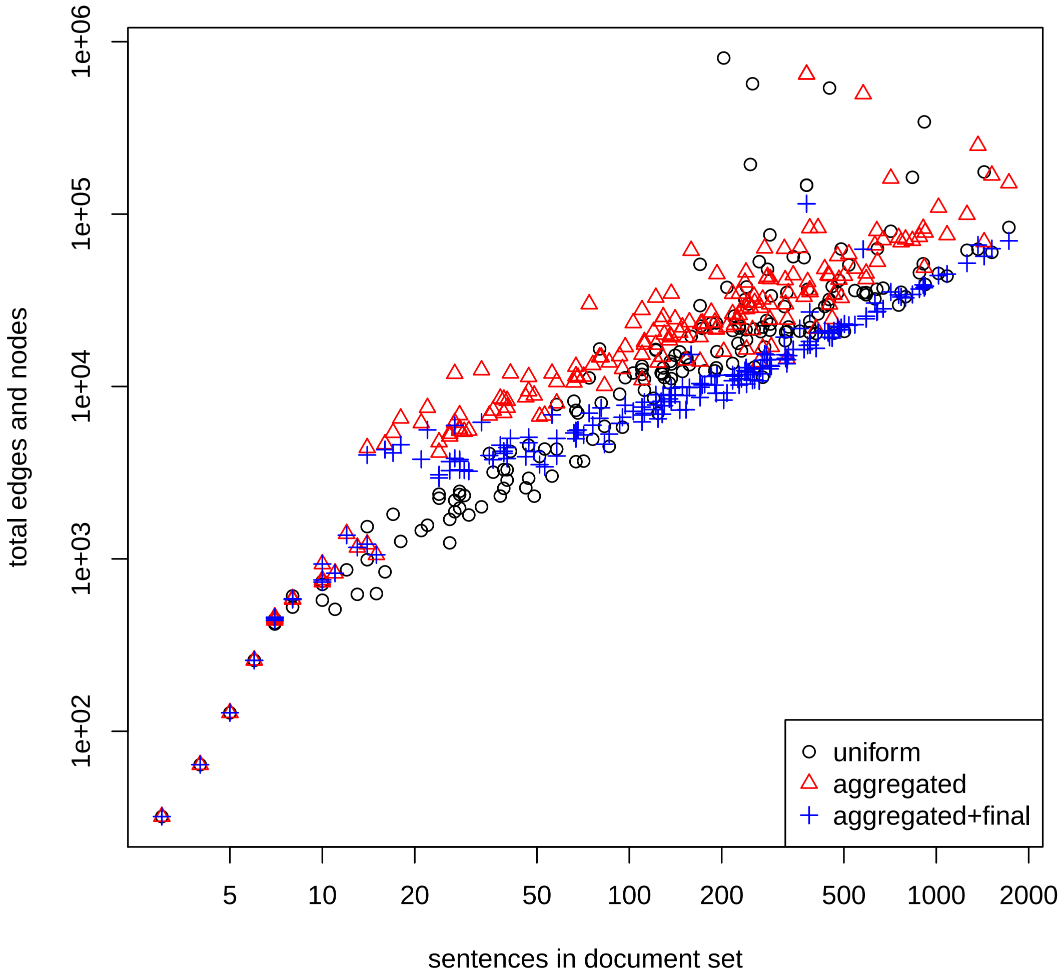


Figure A.2: Efficiency of A* search search is roughly linear in the number of sentences in the document set. The y axis measures the search graph size in terms of the number of edges in the schedule and the number of nodes visited. Measured with the final parameters after training to optimise ROUGE-2 with the three different heuristics and expanding five nodes in each step.

References

Aker, A., Cohn, T. & Gaizauskas, R. (2010), Multi-document summarization using a* search and discriminative learning, *in* 'Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing', Association for Computational Linguistics, Cambridge, MA, pp. 482–491.

Aker, A., Cohn, T. & Gaizauskas, R. (2012), Redundancy reduction for multi-document summaries using a* search and discriminative training, *in* 'Proceedings of the 2nd International Workshop on Exploiting Large Knowledge Repositories, in conjunction with the 1st International Workshop on Automatic Text Summarization for the Future (ATSF-2012)', Universitat Jaume I, Spain, pp. 58–68.

Aker, A., El-Haj, M., Albakour, M.-D. & Kruschwitz, U. (2012), Assessing crowdsourcing quality through objective tasks, *in* 'Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)', European Language Resources Association (ELRA), Istanbul, Turkey, pp. 1456–1461.

Aker, A., Fan, X., Sanderson, M. & Gaizauskas, R. (2012), Investigating summarization techniques for geo-tagged image indexing, *in* 'Advances in Information Retrieval: 34th European Conference on Information Retrieval (ECIR)', Barcelona, Spain, pp. 472–475.

Aker, A. & Gaizauskas, R. (2008), Evaluating automatically generated user-focused multi-document summaries for geo-referenced images, *in* 'Coling 2008: Proceedings of the workshop Multi-source Multilingual Information Extraction and Summarization', Coling 2008 Organizing Committee, Manchester, UK, pp. 41–48.

Aker, A. & Gaizauskas, R. (2009), Summary generation for toponym-referenced images using object type language models, *in* 'Proceedings of the International Conference RANLP-2009', Association for Computational Linguistics, Borovets, Bulgaria, pp. 6–11.

Aker, A. & Gaizauskas, R. (2010*a*), Generating image descriptions using dependency relational patterns, *in* 'Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics', Association for Computational Linguistics, Uppsala, Sweden, pp. 1250–1258.

Aker, A. & Gaizauskas, R. (2010*b*), Model summaries for location-related images, *in* 'Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)', European Language Resources Association (ELRA), Valletta, Malta, pp. 3119–3124.

Aker, A. & Gaizauskas, R. (2011), Understanding the types of information humans associate with geographic objects, *in* 'Proceedings of the 20th ACM international conference on Information and knowledge management', Glasgow, UK, pp. 1929–1932.

Aker, A., Kanoulas, E. & Gaizauskas, R. (2012), A light way to collect comparable corpora from the web, *in* 'Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)', European Language Resources Association (ELRA), Istanbul, Turkey, pp. 15–20.

Aker, A., Plaza, L., Lloret, E. & Gaizauskas, R. (2013), Do humans have conceptual models about geographic objects? a user study, *in* 'Journal of the American Society for Information Science and Technology, In press', Wiley Online Library.

Alfonseca, E. & Rodríguez, P. (2003), Generating extracts with genetic algorithms, *in* 'Advances in Information Retrieval', Springer, pp. 548–548.

Alonso, O. & Mizzaro, S. (2009), Can we get rid of TREC assessors? Using Mechanical Turk for relevance assessment, *in* 'Proceedings of the SIGIR 2009 Workshop on the Future of IR Evaluation', Boston, Massachusetts, pp. 15–16.

Amigo, E., Gonzalo, J., Peinado, V., Peñas, A. & Verdejo, F. (2004), An empirical study of information synthesis task, *in* 'Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume', Barcelona, Spain, pp. 207–214.

Amini, M.-R. & Usunier, N. (2009), Incorporating prior knowledge into a transductive ranking algorithm for multi-document summarization, *in* 'Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval', Boston, Massachusetts, pp. 704–705.

Amini, M., Usunier, N. & Gallinari, P. (2005), Automatic text summarization based on word-clusters and ranking algorithms, *in* 'Advances in Information Retrieval', Springer, pp. 142–156.

Armitage, L. H. & Enser, P. G. (1997), Analysis of user need in image archives, *in* 'Journal of information science', Vol. 23, Sage Publications, pp. 287–299.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R. & Ives, Z. (2007), Dbpedia: A nucleus for a web of open data, Springer, pp. 722–735.

Auer, S. & Lehmann, J. (2007), What have innsbruck and leipzig in common? extracting semantics from wiki content, *in* 'The Semantic Web: Research and Applications', Springer, pp. 503–517.

Baker, K. (2005), Singular value decomposition tutorial, *in* 'Unpublished'.

Balasubramanian, N., Diekema, A. & Goodrum, A. (2004), Analysis of User Image descriptions and Automatic Image Indexing Vocabularies: An Exploratory Study, *in* 'International Workshop on Multidisciplinary Image, Video, and Audio Retrieval and Mining', Sherbrooke, Quebec, Canada.

Banerjee, S. & Pedersen, T. (2003), Extended gloss overlaps as a measure of semantic relatedness, *in* 'International Joint Conference on Artificial Intelligence', Vol. 18, Lawrence Erlbaum Associates LTD, pp. 805–810.

Banko, M. & Etzioni, O. (2008), The tradeoffs between open and traditional relation extraction, *in* 'Proceedings of ACL-08: Human Language Technologies (HLT)', Association for Computational Linguistics, Columbus, Ohio, pp. 28–36.

Barnard, K., Duygulu, P., Forsyth, D., De Freitas, N., Blei, D. M. & Jordan, M. I. (2003), Matching words and pictures, *in* 'The Journal of Machine Learning Research', Vol. 3, JMLR. org, pp. 1107–1135.

Barnard, K. & Forsyth, D. (2001), Learning the semantics of words and pictures, *in* 'Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on', Vol. 2, IEEE, pp. 408–415.

Barzilay, R. & Elhadad, M. (1997), Using lexical chains for text summarization, *in* 'Proceedings of the Association for Computational Linguistics (ACL) workshop on intelligent scalable text summarization', Vol. 17, Madrid, Spain, pp. 10–17.

Barzilay, R., Elhadad, N. & McKeown, K. (2002), Inferring strategies for sentence ordering in multidocument news summarization, *in* 'Journal of Artificial Intelligence Research', Vol. 17, pp. 35–55.

Barzilay, R. & Lapata, M. (2005), Modeling local coherence: An entity-based approach, *in* 'Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)', Association for Computational Linguistics, Ann Arbor, Michigan, pp. 141–148.

Barzilay, R. & Lapata, M. (2008), Modeling local coherence: An entity-based approach, *in* 'Computational Linguistics', Vol. 34, MIT Press, pp. 1–34.

Barzilay, R. & Lee, L. (2004), Catching the drift: Probabilistic content models, with applications to generation and summarization, *in* 'Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies', Association for Computational Linguistics, Boston, Massachusetts, USA, pp. 113–120.

Barzilay, R., McKeown, K. R. & Elhadad, M. (1999), Information fusion in the context of multi-document summarization, *in* 'Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics', Association for Computational Linguistics, College Park, Maryland, USA, pp. 550–557.

Baxendale, P. (1958), Machine-made index for technical literature: an experiment, *in* 'IBM Journal of Research and Development', Vol. 2, IBM Corp., pp. 354–361.

Berg-Kirkpatrick, T., Gillick, D. & Klein, D. (2011), Jointly learning to extract and compress, *in* 'Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies', Association for Computational Linguistics, Portland, Oregon, USA, pp. 481–490.

Biadsy, F., Hirschberg, J. & Filatova, E. (2008), An unsupervised approach to biography production using wikipedia, *in* 'Proceedings of ACL-08: HLT', Association for Computational Linguistics, Columbus, Ohio, pp. 807–815.

Bollegala, D., Okazaki, N. & Ishizuka, M. (2010), A bottom-up approach to sentence ordering for multi-document summarization, *in* 'Information processing & management', Vol. 46, Elsevier, pp. 89–109.

Bollegala, D., Okazaki, N. & Ishizuka, M. (2012), A preference learning approach to sentence ordering for multi-document summarization, *in* 'Information Sciences', Vol. 217, Elsevier, pp. 78–95.

Bouma, G., Fahmi, I., Mur, J., Van Noord, G., van der Plas, L. & Tiedemann, J. (2007), Using syntactic knowledge for QA, *in* 'Evaluation of Multilingual and Multi-modal Information Retrieval', Vol. 4730, Springer, pp. 318–327.

Brandow, R., Mitze, K. & Rau, L. F. (1995), Automatic condensation of electronic publications by sentence selection, *in* 'Information Processing & Management', Vol. 31, Elsevier, pp. 675–685.

Brown, P., Pietra, V., Pietra, S. & Mercer, R. (1993), The mathematics of statistical machine translation: Parameter estimation, *in* 'Computational linguistics', Vol. 19, MIT Press, pp. 263–311.

Bunescu, R. & Mooney, R. (2005), A shortest path dependency kernel for relation extraction, *in* 'Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing', Association for Computational Linguistics Morristown, NJ, USA, pp. 724–731.

Carbonell, J. & Goldstein, J. (1998), The use of MMR, diversity-based reranking for reordering documents and producing summaries, *in* 'Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval', ACM Press New York, NY, USA, pp. 335–336.

Chen, H. & Ng, T. (1995), An algorithmic approach to concept exploration in a large knowledge network (automatic thesaurus consultation): symbolic branch-and-bound search vs. connectionist hopfield net activation, *in* 'Journal of the American Society for Information Science and Technology (JASIST)', Vol. 46, pp. 348–369.

Choi, Y. & Rasmussen, E. M. (2002), Users'relevance criteria in image retrieval in American history, *in* 'Information Processing and Management', Vol. 38, Elsevier, pp. 695–726.

Chuang, W. & Yang, J. (2000), Extracting sentence segments for text summarization: a machine learning approach, *in* 'Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval', ACM, pp. 152–159.

Crossley, S. A. & McNamara, D. S. (2010), Cohesion, coherence, and expert evaluations of writing proficiency, *in* 'Proceedings of the 32nd annual conference of the Cognitive Science Society', pp. 984–989.

Culotta, A. & Sorensen, J. (2004), Dependency Tree Kernels for Relation Extraction, *in* 'Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04)', Barcelona, Spain, pp. 423–429.

Dakka, W. & Ipeirotis, P. G. (2008), Automatic extraction of useful facet hierarchies from text databases, *in* 'Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on', IEEE, pp. 466–475.

Dang, H. T. (2005), Overview of DUC 2005, *in* 'Proceedings of the Document Understanding Conference (DUC)'.

Dang, H. T. (2006), Overview of DUC 2006, *in* 'Proceedings of the Document Understanding Conference (DUC)'.

Deschacht, K. & Moens, M.-F. (2007), Text analysis for automatic image annotation, *in* 'Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics', Association for Computational Linguistics, Prague, Czech Republic, pp. 1000–1007.

Di Fabbrizio, G., Aker, A. & Gaizauskas, R. (2011), Starlet: Multi-document summarization of service and product reviews with balanced rating distributions, *in* 'Proceedings of the International Conference on Data Mining Workshops (ICDMW)', pp. 67–74.

Dräger, M. & Koller, A. (2012), Generation of landmark-based navigation instructions from open-source data, *in* 'Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics', Association for Computational Linguistics, pp. 757–766.

Dunning, T. (1993), Accurate methods for the statistics of surprise and coincidence, *in* 'Computational linguistics', Vol. 19, MIT Press, pp. 61–74.

Duygulu, P., Barnard, K., de Freitas, J. & Forsyth, D. (2002), Object Recognition as Machine Translation: Learning a Lexicon for a Fixed Image Vocabulary, *in* 'In Seventh European Conference on Computer Vision (ECCV)', Vol. 4, pp. 97–112.

Eakins, J. (1998), Techniques for image retrieval, *in* 'Library & information briefings', number 85, British Library Research and Development Department, pp. 1–15.

Edmundson, P., H. (1969), New Methods in Automatic Extracting, *in* 'Journal of the Association for Computing Machinery', Vol. 16, pp. 264–285.

El-Haj, M., Kruschwitz, U. & Fox, C. (2010), Using Mechanical Turk to Create a Corpus of Arabic Summaries, *in* 'Proceedings of the Language Resources and Evaluation Conference (LREC) Workshop on Semitic Languages', Valletta, Malta, pp. 36–39.

Elsner, M. & Charniak, E. (2011), Extending the entity grid with entity-specific features, *in* 'Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2', Association for Computational Linguistics, pp. 125–129.

Erkan, G. & Radev, D. (2004), LexRank: Graph-based lexical centrality as salience in text summarization, *in* 'Journal of Artificial Intelligence Research', Vol. 22, pp. 457–479.

Etzioni, O., Banko, M., Soderland, S. & Weld, D. S. (2008), Open information extraction from the web, *in* 'Communications of the ACM', Vol. 51, ACM, pp. 68–74.

Eysenck, M. & Keane, M. (2005), Cognitive psychology: A student's handbook, Psychology Press.

Fan, X., Aker, A., Tomko, M., Smart, P., Sanderson, M. & Gaizauskas, R. (2010), Automatic image captioning from the web for gps photographs, *in* 'Proceedings of the International Conference on Multimedia Information Retrieval (MIR)', pp. 445–448.

Farhadi, A., Endres, I., Hoiem, D. & Forsyth, D. (2009), Describing objects by their attributes, *in* 'Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on', IEEE, pp. 1778–1785.

Farhadi, A., Hejrati, M., Sadeghi, M., Young, P., Rashtchian, C., Hockenmaier, J. & Forsyth, D. (2010), Every picture tells a story: generating sentences from images, *in* 'Computer Vision–ECCV 2010', Springer, pp. 15–29.

Farzindar, A., Rozon, F. & Lapalme, G. (2005), CATS a topic-oriented multi-document summarization system at DUC 2005, *in* 'Proceedings of the 2005 Document Understanding Workshop (DUC2005)'.

Feng, D., Besana, S. & Zajac, R. (2009), Acquiring high quality non-expert knowledge from on-demand workforce, *in* 'Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources', Association for Computational Linguistics, Morristown, NJ, USA, pp. 51–56.

Feng, Y. & Lapata, M. (2008), Automatic image annotation using auxiliary text information, *in* 'Proceedings of Association for Computational Linguistics (ACL) 2008', Association for Computational Linguistics, Columbus, Ohio, pp. 272–280.

Feng, Y. & Lapata, M. (2010*a*), How many words is a picture worth? automatic caption generation for news images, *in* 'Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics', Association for Computational Linguistics, pp. 1239–1249.

Feng, Y. & Lapata, M. (2010*b*), Topic models for image annotation and text illustration, *in* 'Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics', Association for Computational Linguistics, pp. 831–839.

Feng, Y. & Lapata, M. (2010*c*), Visual information in semantic representation, *in* 'Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics', Association for Computational Linguistics, pp. 91–99.

Filatova, E., Hatzivassiloglou, V. & McKeown, K. (2006), Automatic creation of domain templates, *in* 'Proceedings of the COLING/ACL on Main conference poster sessions', Association for Computational Linguistics, pp. 207–214.

Fisher, S. & Roark, B. (2006), Query-focused summarization by supervised sentence ranking and skewed word distributions, *in* 'Proceedings of the Document Understanding Conference (DUC)', New York, USA.

Fung, P. & Ngai, G. (2006), One story, one flow: Hidden markov story models for multilingual multidocument summarization, *in* 'ACM Transactions on Speech and Language Processing (TSLP)', Vol. 3, ACM, pp. 1–16.

Gillick, D. & Favre, B. (2009), A scalable global model for summarization, *in* 'Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing', Association for Computational Linguistics, pp. 10–18.

Gillick, D., Riedhammer, K., Favre, B. & Hakkani-Tür, D. (2009), A global optimization framework for meeting summarization, *in* 'Acoustics, Speech and Signal Processing, 2009. ICASSP 2009', IEEE, pp. 4769–4772.

Glickman, O. (2006), Applied textual entailment, Ph.D. thesis, Bar Ilan University.

Goldstein, J., Mittal, V., Carbonell, J. & Kantrowitz, M. (2000), Multi-document summarization by sentence extraction, *in* 'NAACL-ANLP 2000 Workshop on Automatic summarization', Association for Computational Linguistics, pp. 40–48.

Gong, Y. & Liu, X. (2001), Generic text summarization using relevance measure and latent semantic analysis, *in* 'Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval', pp. 19–25.

Greisdorf, H. & O Connor, B. (2002), Modelling what users see when they look at images: a cognitive viewpoint, *in* 'Journal of Documentation', Vol. 58, MCB University Press, pp. 6–29.

Grosz, B., Weinstein, S. & Joshi, A. (1995), Centering: A framework for modeling the local coherence of discourse, *in* 'Computational linguistics', Vol. 21, MIT Press, pp. 203–225.

Gurevych, I. & Strube, M. (2004), Semantic similarity applied to spoken dialogue summarization, *in* 'Proceedings of the 20th international conference on Computational Linguistics', Association for Computational Linguistics, Geneva, Switzerland, pp. 764–770.

Halliday, M. & Hasan, R. (1976), Cohesion in english, Longman Group Ltd.

Hand, T. (1997), A proposal for task-based evaluation of text summarization systems, *in* 'Proceedings of the Association for Computational Linguistics conference', Madrid, Spain, pp. 31–38.

Hatzivassiloglou, V., Klavans, J., Holcombe, M., Barzilay, R., Kan, M. & McKeown, K. (2001), Simfinder: A flexible clustering tool for summarization, *in* 'Proceedings of the Workshop on Automatic Summarization, North American Chapter of the Association for Computational Linguistics: Human Language Technologies', pp. 41–49.

Hearst, M. (1992), Automatic acquisition of hyponyms from large text corpora, *in* 'Proceedings of the 14th conference on Computational linguistics-Volume 2', Association for Computational Linguistics Morristown, NJ, USA, pp. 539–545.

Hearst, M. (1997), TextTiling: segmenting text into multi-paragraph subtopic passages, *in* 'Computational linguistics', Vol. 23, MIT Press, pp. 33–64.

Hirao, T., Isozaki, H., Maeda, E. & Matsumoto, Y. (2002), Extracting important sentences with support vector machines, *in* 'Proceedings of the 19th international conference on Computational linguistics-Volume 1', Association for Computational Linguistics, pp. 1–7.

Hölldobler, S., Karabaev, E. & Skvortsova, O. (2006), Flucap: a heuristic search planner for first-order mdps, *in* 'Journal of Artificial Intelligence Research', Vol. 27, AI Access Foundation, pp. 419–439.

Hollink, L., Schreiber, A., Wielinga, B. & Worring, M. (2004), Classification of user image descriptions, *in* 'International Journal of Human-Computer Studies', Vol. 61, Elsevier, pp. 601–626.

Hovy, E. (1988), Planning coherent multisentential text, *in* 'Proceedings of the 26th annual meeting on Association for Computational Linguistics', Association for Computational Linguistics, pp. 163–169.

Hovy, E. & Lin, C.-Y. (1998), Automated text summarization and the summarist system, *in* 'Proceedings of a workshop on held at Baltimore, Maryland: October 13-15, 1998', Association for Computational Linguistics, pp. 197–214.

Jaimes, A. & Chang, S. (2000), A Conceptual Framework for Indexing Visual Information at Multiple Levels, *in* 'IS&T/SPIE Internet Imaging', Vol. 3964, pp. 2–15.

Janarthanam, S., Lemon, O., Liu, X., Bartie, P., Mackaness, W., Dalmas, T. & Goetze, J. (2012), Integrating location, visibility, and question-answering in a spoken dialogue system for pedestrian city exploration, *in* 'Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue', Association for Computational Linguistics, pp. 134–136.

Jiang, J. & Conrath, D. (1997), Semantic similarity based on corpus statistics and lexical taxonomy, *in* 'Proceedings of the 10th International Conference on Research on Computational Linguistics, Taiwan', pp. 19–33.

Joachims, T. (2002*a*), Learning to classify text using support vector machines: Methods, theory and algorithms, Vol. 186, Kluwer Academic Publishers Norwell, MA, USA.

Joachims, T. (2002*b*), Optimizing search engines using clickthrough data, *in* 'Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 133–142.

Joergensen, C. (1996), Indexing Images: Testing an Image Description Template, *in* 'Proceedings of the annual Meeting-American Society for Information Science', Vol. 33, pp. 209–213.

Jones, G., Fantino, F., Newman, E. & Zhang, Y. (n.d.), Domain-Specific Query Translation for Multilingual Information Access using Machine Translation Augmented With Dictionaries Mined from Wikipedia, *in* 'Proceedings of the 2nd International Workshop on Cross Lingual Information Access: Addressing the Information Need of Multilingual Societies', Hyderabad, India, pp. 34–41.

Jones, K. (1993), What might be in a summary, *in* 'Information Retrieval', Vol. 93, pp. 9–26.

Jones, K. (1999), Automatic summarizing: factors and directions, *in* 'Advances in Automatic Text Summarization', MIT Press, pp. 1–12.

Jörgensen, C. (1998), Attributes of images in describing tasks, *in* 'Information Processing and Management', Vol. 34, Elsevier, pp. 161–174.

Jurafsky, D. & Martin, J. (2008), Speech and language processing, Prentice Hall.

Kaisser, M. & Lowe, J. (2008), Creating a research collection of question answer sentence pairs with amazon's mechanical turk, *in* 'Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)', Marrakech, Morocco.

Kalaycilar, F. & Cicekli, I. (2008), Turkeyx: Turkish keyphrase extractor, *in* '23rd International Symposium onComputer and Information Sciences (ISCIS)', IEEE, pp. 1–4.

Kazai, G. (2011), In Search of Quality in Crowdsourcing for Search Engine Evaluation, *in* 'Proceedings of the 33^{rd} European Conference on Information Retrieval (ECIR)', Vol. 6611 of *Lecture Notes in Computer Science*, Springer, pp. 165–176.

Kim, I., Le, D. & Thoma, G. (2007), Identification of "comment-on sentences" in online biomedical documents using support vector machines, *in* 'Proceedings of the SPIE Conference on Document Recognition and Retrieval', Vol. 68150, pp. X1–X9.

Kittur, A., Chi, E. H. & Suh, B. (2008), Crowdsourcing user studies with mechanical turk, *in* 'Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems', ACM, pp. 453–456.

Kojima, A., Takaya, M., Aoki, S., Miyamoto, T. & Fukunaga, K. (2008), Recognition and textual description of human activities by mobile robot, *in* '3rd International Conference onInnovative Computing Information and Control (ICICIC)', IEEE, pp. 53–53.

Kruengkrai, C. & Jaruskulchai, C. (2003), Generic text summarization using local and global properties of sentences, *in* 'Proceedings of the International Conference on Web Intelligence (WI).', IEEE, pp. 201–206.

Kupiec, J., Pedersen, J. & Chen, F. (1995), A trainable document summarizer, *in* 'Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval', pp. 68–73.

Kutlu, M., Cığır, C. & Cicekli, I. (2010), Generic Text Summarization for Turkish, *in* 'Proceedings of ISCIS'.

Lapata, M. (2003), Probabilistic text structuring: Experiments with sentence ordering, *in* 'Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1', Association for Computational Linguistics, pp. 545–552.

Li, L., Zhou, K., Xue, G., Zha, H. & Yu, Y. (2009), Enhancing diversity, coverage and balance for summarization through structure learning, *in* 'Proceedings of the 18th international conference on World wide web', ACM, pp. 71–80.

Li, P., Jiang, J. & Wang, Y. (2010), Generating templates of entity summaries with an entity-aspect model and pattern mining, *in* 'Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics', Association for Computational Linguistics, pp. 640–649.

Li, S., Kulkarni, G., Berg, T., Berg, A. & Choi, Y. (2011), Composing simple image descriptions using web-scale n-grams, *in* 'Proceedings of the Fifteenth Conference on Computational Natural Language Learning', Association for Computational Linguistics, pp. 220–228.

Liakata, M., Teufel, S., Siddharthan, A. & Batchelor, C. (2010), Corpora for the conceptualisation and zoning of scientific papers, *in* '7th International Conference on Language Resources and Evaluation', pp. 2054–2061.

Liddy, E. D. (1991), The discourse-level structure of empirical abstracts: An exploratory study, *in* 'Information Processing & Management', Vol. 27, Elsevier, pp. 55–81.

Lin, C. & Hovy, E. (1997), Identifying topics by position, *in* 'Proceedings of the fifth conference on Applied natural language processing', Association for Computational Linguistics, pp. 283–290.

Lin, C. & Hovy, E. (2001), Neats: A multidocument summarizer, *in* 'Proceedings of the Document Understanding Workshop (DUC)'.

Lin, C. & Hovy, E. (2002), From single to multi-document summarization: A prototype system and its evaluation, *in* 'Proceedings of the 40th Annual Meeting on Association for Computational Linguistics', Association for Computational Linguistics, pp. 457–464.

Lin, C.-Y. (2004), Rouge: A package for automatic evaluation of summaries, *in* 'Text Summarization Branches Out: Proceedings of the ACL-04 Workshop', Association for Computational Linguistics, Barcelona, Spain, pp. 74–81.

Lin, C.-Y. & Hovy, E. (2000), The automated acquisition of topic signatures for text summarization, *in* 'Proceedings of the 18th conference on Computational linguistics-Volume 1', Association for Computational Linguistics, pp. 495–501.

Lin, D. (1998), An information-theoretic definition of similarity, *in* 'Proceedings of the 15th international conference on machine learning', Vol. 1, pp. 296–304.

Lin, G., Peng, H., Ma, Q., Wei, J. & Qin, J. (2010), Improving diversity in Web search results re-ranking using absorbing random walks, *in* 'Machine Learning and Cybernetics (ICMLC), 2010 International Conference on', Vol. 5, IEEE, pp. 2116–2421.

Lin, H. & Bilmes, J. (2010), Multi-document summarization via budgeted maximization of submodular functions, *in* 'Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics', Association for Computational Linguistics, pp. 912–920.

Lin, Z., Ng, H. & Kan, M. (2011), Automatically evaluating text coherence using discourse relations, *in* 'Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1', Association for Computational Linguistics, pp. 997–1006.

Liu, D., He, Y., Ji, D. & Yang, H. (2006), Genetic algorithm based multi-document summarization, *in* 'PRICAI 2006: Trends in Artificial Intelligence', Springer, pp. 1140–1144.

Lloret, E., Ferrández, O., Munoz, R. & Palomar, M. (2008), A text summarization approach under the influence of textual entailment, *in* 'Proceedings of the 5th International Workshop on Natural Language Processing and Cognitive Science (NLPCS 2008)', pp. 22–31.

Lopez, C., Prince, V., Roche, M. et al. (2011), Automatic titling of articles using position and statistical information, *in* 'RANLP'11: Recent Advances in Natural Language Processing', pp. 727–732.

Louis, A. & Nenkova, A. (2008), Automatic summary evaluation without human models, *in* 'In Proceedings of the Text Analysing Conference, (TAC 2008)'.

Louis, A. & Nenkova, A. (2012), A coherence model based on syntactic patterns, *in* 'Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning', Association for Computational Linguistics, Jeju Island, Korea, pp. 1157–1168.

Luhn, H. P. (1958), The automatic creation of literature abstracts, *in* 'IBM Journal of research and development', Vol. 2, IBM, pp. 159–165.

Mani, I. (2001), Automatic summarization, Vol. 3, John Benjamins Publishing Company.

Mani, I. & Bloedorn, E. (1997), Multi-document summarization by graph search and matching, *in* 'Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI)', Providence, Rhode Island, pp. 622–628.

Mani, I. & Bloedorn, E. (1998), Machine learning of generic and user-focused summarization, *in* 'Proceedings of the natioanl conference on artificial intelligence', pp. 821–826.

Mani, I., House, D., Klein, G., Hirschman, L., Firmin, T. & Sundheim, B. (1999), The tipster summac text summarization evaluation, *in* 'Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics', Association for Computational Linguistics, pp. 77–85.

Mann, G. (2002), Fine-grained proper noun ontologies for question answering, *in* 'International Conference On Computational Linguistics', Association for Computational Linguistics Morristown, NJ, USA, pp. 1–7.

Mann, W. & Thompson, S. (1988), Rhetorical structure theory: Toward a functional theory of text organization, *in* 'Text-Interdisciplinary Journal for the Study of Discourse', Vol. 8, pp. 243–281.

Manning, C., Raghavan, P. & Schutze, H. (2008), Introduction to information retrieval, Vol. 1, Cambridge University Press Cambridge.

Marcu, D. (1997*a*), From discourse structures to text summaries, *in* 'Proceedings of the Association of Computer Linguistics (ACL) Workshop on Intelligent Scalable Text Summarization', pp. 82–88.

Marcu, D. (1997*b*), 'The rhetorical parsing, summarization and generation of natural language texts.'.

Marsh, E. & White, M. (2003), A taxonomy of relationships between images and text, *in* 'Journal of Documentation', Vol. 59, pp. 647–672.

Mason, W. & Watts, D. J. (2009), Financial incentives and the performance of crowds, *in* 'Proceedings of the ACM SIGKDD workshop on human computation', ACM, pp. 77–85.

McDonald, R. (2007), A study of global inference algorithms in multi-document summarization, *in* 'Advances in Information Retrieval', Springer, pp. 557–564.

McKeown, K., Klavans, J., Hatzivassiloglou, V., Barzilay, R. & Eskin, E. (1999), Towards multidocument summarization by reformulation: Progress and prospects, *in* 'Proceedings of the National Conference on Artificial Intelligence', John Wiley & Sons LTD, pp. 453–460.

McKeown, K., Passonneau, R., Elson, D., Nenkova, A. & Hirschberg, J. (2005), Do Summaries Help? A Task-Based Evaluation of Multi-Document Summarization, *in* '28th Annual ACM SIGIR Conference on Research and Development in Information Retrieval', Salvador, Brazil.

McKeown, K. & Radev, D. (1995), Generating summaries of multiple news articles, *in* 'Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval', ACM, pp. 74–82.

Metzler, D. & Kanungo, T. (2008), Machine learned sentence selection strategies for query-biased summarization, *in* 'SIGIR 2008 Workshop on Learning to Rank for Information Retrieval'.

Mihalcea, R., Corley, C. & Strapparava, C. (2006), Corpus-based and knowledge-based measures of text semantic similarity, *in* 'Proceedings of the National Conference on Artificial Intelligence', pp. 775–780.

Mihalcea, R. & Tarau, P. (2004), Textrank: Bringing order into texts, *in* 'Proceedings of Emperical Methods in Natural Language Processing (EMNLP)', Vol. 4, Barcelona, Spain, pp. 404–411.

Miller, G. (1995), WordNet: a lexical database for English, *in* 'Communications of the ACM', Vol. 38, ACM, NY, USA, pp. 39–41.

Miller, G. A. (1993), 'Five papers on wordnet'.

Mitra, M., Singhal, A. & Buckley, C. (1997), Automatic text summarization by paragraph extraction, *in* 'In ACL/EACL-97 Workshop on Intelligent Scalable Text Summarization', Madrid, Spain.

Mori, Y., Takahashi, H. & Oka, R. (2000), Automatic word assignment to images based on image division and vector quantization, *in* 'Proceedings of RIAO 2000: Content-Based Multimedia Information Access'.

Morris, J. & Hirst, G. (1991), Lexical cohesion computed by thesaural relations as an indicator of the structure of text, *in* 'Computational linguistics', Vol. 17, MIT Press, pp. 21–48.

Murray, G., Renals, S. & Carletta, J. (2005), Extractive summarization of meeting recordings, *in* 'Proceedings of the 9th European Conference on Speech Communication and Technology', pp. 593–596.

Nenkova, A. & McKeown, K. (2011), Automatic Summarization, *in* 'Foundations and Trends in Information Retrieval'.

Nenkova, A. & Passonneau, R. (2004), Evaluating content selection in summarization: The pyramid method, *in* 'HLT-NAACL 2004: Main Proceedings', Association for Computational Linguistics, Boston, Massachusetts, USA, pp. 145–152.

Nenkova, A., Passonneau, R. & McKeown, K. (2007), 'The pyramid method: Incorporating human content selection variation in summarization evaluation', **4**(2).

Nenkova, A., Vanderwende, L. & McKeown, K. (2006), A compositional context sensitive multi-document summarizer: exploring the factors that influence summarization, *in* 'Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval', ACM, pp. 573–580.

Nobata, C., Sekine, S., Isahara, H. & Grishman, R. (2002), Summarization system integrated with named entity tagging and ie pattern discovery, *in* 'Proceedings of the Language Resources Evaluation Conference (LREC)', pp. 1742–1745.

Och, F. J. (2003), Minimum error rate training in statistical machine translation, *in* 'Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1', Association for Computational Linguistics, pp. 160–167.

Orasan, C. (2003), An evolutionary approach for improving the quality of automatic summaries, *in* 'Proceedings of the ACL 2003 workshop on Multilingual summarization and question answering-Volume 12', Association for Computational Linguistics, pp. 37–45.

Ouyang, Y., Li, W., Li, S. & Lu, Q. (2011), 'Applying regression models to query-focused multi-document summarization', **47**(2), 227–237.

Owczarzak, K. & Dang, H. (2009), Evaluation of automatic summaries: Metrics under varying data conditions, *in* 'Proceedings of the 2009 Workshop on Language Generation and Summarisation', Association for Computational Linguistics, pp. 23–30.

Ozsoy, M., Cicekli, I. & Alpaslan, F. (2010), Text summarization of Turkish texts using latent semantic analysis, *in* 'Proceedings of the 23rd International Conference on Computational Linguistics', Association for Computational Linguistics, pp. 869–876.

Paice, D., C. (1980), The automatic generation of literature abstracts: an approach based on the identification of self-indicating phrases, *in* 'Proceedings of the 3rd annual ACM conference on Research and development in information retrieval', Butterworth & Co. Kent, UK, pp. 172–191.

Paice, D., C. & Jones, A., P. (1993), The identification of important concepts in highly structured technical papers, *in* 'Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval', ACM New York, NY, USA, pp. 69–78.

Pan, J.-Y., Yang, H.-J., Duygulu, P. & Faloutsos, C. (2004), Automatic image captioning, *in* 'Multimedia and Expo, 2004. ICME'04. 2004 IEEE International Conference on', Vol. 3, IEEE, pp. 1987–1990.

Panofsky, E. (1972), *Studies in Iconology: Humanistic Themes in the Art of the Renaissance*, Westview Press.

Pollock, J. & Zamora, A. (1975), Automatic abstracting research at chemical abstracts service, *in* 'Journal of Chemical Information and Computer Sciences', Vol. 15, ACS Publications, pp. 226–232.

Ponzetto, S. & Strube, M. (2007), Deriving a Large Scale Taxonomy from Wikipedia, *in* 'Proceedings of the national conference on artificial intelligence', pp. 1440–1445.

Purves, R., Edwardes, A. & Sanderson, M. (2008), Describing the where–improving image annotation and search through geography, *in* '1st Intl. Workshop on Metadata Mining for Image Understanding, Funchal, Madeira-Portugal'.

Radev, D., Blair-Goldensohn, S. & Zhang, Z. (2001), Experiments in single and multi-document summarization using MEAD, *in* 'Document Understanding Conference'.

Radev, D., Jing, H., Stys̀, M. & Tam, D. (2004), Centroid-based summarization of multiple documents, *in* 'Information Processing and Management', Vol. 40, Elsevier, pp. 919–938.

Radev, D. & McKeown, K. (1998), Generating natural language summaries from multiple online sources, *in* 'Computational Linguistics', Vol. 24, MIT Press, pp. 470–500.

Resnik, P. (1995), Using information content to evaluate semantic similarity in a taxonomy, *in* 'In Proceedings of International Joint Conferences on Artificial Intelligence (IJCAI)', Montreal Canada, pp. 448—453.

Riedhammer, K., Gillick, D., Favre, B. & Hakkani-Tür, D. (2008), Packing the meeting summarization knapsack, *in* 'Proceedings of the Interspeech Conference, Brisbane, Australia'.

Rodríguez, M. A. & Egenhofer, M. J. (2003), 'Determining semantic similarity among entity classes from different ontologies', *Knowledge and Data Engineering, IEEE Transactions on* **15**(2), 442–456.

Rosch, E. (1999), 'Principles of categorization', pp. 189–206.

Russell, S., Norvig, P., Canny, J., Malik, J. & Edwards, D. (1995), Artificial intelligence: a modern approach, Prentice hall Englewood Cliffs, NJ.

Sabou, M., dAquin, M. & Motta, E. (2008), Scarlet: semantic relation discovery by harvesting online ontologies, *in* 'The Semantic Web: Research and Applications', Springer, pp. 854–858.

Sabou, M., Wroe, C., Goble, C. & Stuckenschmidt, H. (2005), 'Learning domain ontologies for semantic web service descriptions', *Web Semantics: Science, Services and Agents on the World Wide Web* **3**(4), 340–365.

Saggion, H. (2005), Topic-based Summarization at DUC 2005, *in* 'Document Understanding Conference (DUC)'.

Saggion, H. (2008), A robust and adaptable summarization tool, *in* 'Traitement Automatique des Langues', pp. 103–125.

Saggion, H., Bontcheva, K. & Cunningham, H. (2003), Robust generic and query-based summarisation, *in* 'Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 2', Association for Computational Linguistics, pp. 235–238.

Saggion, H. & Gaizauskas, R. (2004), Multi-document summarization by cluster/profile relevance and redundancy removal, *in* 'Document Understanding Conference (DUC)'.

Saggion, H., Radev, D., Teufel, S., Lam, W. & Strassel, S. M. (2002), 'Developing infrastructure for the evaluation of single and multi-document summarization systems in a cross-lingual environment', **1001**, 48109–1092.

Sahuguet, A. & Azavant, F. (2001), Building intelligent Web applications using lightweight wrappers, *in* 'Data & Knowledge Engineering', Vol. 36, Elsevier, pp. 283–316.

Salton, G. & Buckley, C. (1988), Term-weighting approaches in automatic text retrieval, *in* 'Information Processing and Management: an International Journal', Vol. 24, Pergamon Press, Inc. Tarrytown, NY, USA, pp. 513–523.

Salton, G. & Lesk, E., M. (1968), Computer evaluation of indexing and text processing, *in* 'Journal of the ACM', Vol. 15, ACM Press, New York, NY, USA, pp. 8–36.

Salton, G., Singhal, A., Mitra, M. & Buckley, C. (1997), Automatic text structuring and summarization, *in* 'Information Processing & Management', Vol. 33, Elsevier, pp. 193–207.

Salton, G., Wong, A. & Yang, C. (1975), A vector space model for automatic indexing, *in* 'Communications of the ACM', Vol. 18, ACM, pp. 613–620.

Sauper, C. & Barzilay, R. (2009), Automatically generating wikipedia articles: A structure-aware approach, *in* 'Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1', Association for Computational Linguistics, pp. 208–216.

Sekine, S. (2006), On-demand information extraction, *in* 'Proceedings of Association for Computational Linguistics', Sydney, Australia, pp. 731–738.

Shatford, S. (1986), Analyzing the Subject of a Picture: A Theoretical Approach, *in* 'Cataloging and Classification Quarterly', Vol. 6, pp. 39–61.

Smith, B. & Mark, D. (2001), Geographical categories: an ontological investigation, *in* 'International Journal of Geographical Information Science'.

Snow, R., O'Connor, B., Jurafsky, D. & Ng, A. (2008), Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks, *in* 'Proceedings of the Conference on Empirical Methods in Natural Language Processing', Association for Computational Linguistics, pp. 254–263.

Song, F. & Croft, W. (1999), A general language model for information retrieval, *in* 'Proceedings of the eighth international conference on Information and knowledge management', ACM New York, NY, USA, pp. 316–321.

Soricut, R. & Marcu, D. (2006), Discourse generation using utility-trained coherence models, *in* 'Proceedings of the COLING/ACL on Main conference poster sessions', Association for Computational Linguistics, pp. 803–810.

Sparck Jones, K. & Galliers, J. (1996), Evaluating natural language processing systems(an analysis and review), *in* 'Lecture notes in computer science', Springer-Verlag.

Steinberger, J. & Jezek, K. (2004), Using latent semantic analysis in text summarization and summary evaluation, *in* 'Proceedings of the Information System Implementation and Modeling (ISIM) Conference', pp. 93–100.

Stevenson, M. & Greenwood, M. (2005), A semantic approach to IE pattern induction, *in* 'Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics', Association for Computational Linguistics Morristown, NJ, USA, pp. 379–386.

Stevenson, M. & Greenwood, M. (2009), Dependency Pattern Models for Information Extraction, *in* 'Research on Language and Computation', Vol. 7, pp. 13–39.

Su, Q., Pavlov, D., Chow, J.-H. & Baker, W. C. (2007), Internet-scale collection of human-reviewed data, *in* 'Proceedings of the 16th international conference on World Wide Web', ACM, pp. 231–240.

Suchanek, F., Kasneci, G. & Weikum, G. (2007), Yago: a core of semantic knowledge, *in* 'Proceedings of the 16th international conference on World Wide Web', ACM New York, NY, USA, pp. 697–706.

Suchanek, F., Kasneci, G. & Weikum, G. (2008), 'Yago: A large ontology from wikipedia and wordnet', **6**(3), 203–217.

Sudo, K., Sekine, S. & Grishman, R. (2001), Automatic pattern acquisition for Japanese information extraction, *in* 'Proceedings of the 1st international conference on Human language technology research', Association for Computational Linguistics, pp. 1–7.

Sudo, K., Sekine, S. & Grishman, R. (2003), An improved extraction pattern representation model for automatic ie pattern acquisition, *in* 'Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1', Association for Computational Linguistics, pp. 224–231.

Teufel, S. (2010), The Structure of Scientific Articles: Applications to Citation Indexing and Summarization, *in* 'CSLI Studies in Computational Linguistics', Chicago University Press.

Teufel, S. & Moens, M. (1997), Sentence extraction as a classification task, *in* 'Proceedings of the Association for Computational Linguistics', Vol. 97, pp. 58–65.

Teufel, S. & Moens, M. (2002), Summarizing scientific articles: experiments with relevance and rhetorical status, *in* 'Computational Linguistics', Vol. 28, MIT Press, pp. 409–445.

Teufel, S. & van Halteren, H. (2004), Evaluating information content by factoid analysis: human annotation and stability, *in* 'Proceedings of the Conference on Empirical Methods in Natural Language Processing', pp. 419–426.

Tombros, A. & Sanderson, M. (1998), Advantages of query biased summaries in information retrieval, *in* 'Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval', ACM New York, NY, USA, pp. 2–10.

Toutanova, K., Brockett, C., Gamon, M., Jagarlamudi, J., Suzuki, H. & Vanderwende, L. (2007), The pythy summarization system: Microsoft research at DUC 2007, *in* 'Document Understanding Conference'.

Uzêda, V., Pardo, T. & Nunes, M. (2010), A comprehensive comparative evaluation of rst-based summarization methods, *in* 'ACM Transactions on Speech and Language Processing (TSLP)', Vol. 6, ACM, pp. 1–20.

Vapnik, V. (2000), The nature of statistical learning theory, Springer Verlag.

Voorhees, E. (2003), Overview of the TREC 2003 Question Answering Track, *in* 'Proceedings of the Twelfth Text REtrieval Conference (TREC)'.

Walker, M., Joshi, A. & Prince, E. (1998), Centering theory in discourse, Clarendon Press Oxford, UK.

Wang, C., Jing, F., Zhang, L. & Zhang, H. (2007), Learning query-biased web page summarization, *in* 'Proceedings of the sixteenth ACM conference on Conference on information and knowledge management', ACM, pp. 555–562.

Wentland, W., Knopp, J., Silberer, C. & Hartung, M. (2008), Building a Multilingual Lexical Resource for Named Entity Disambiguation, Translation and Transliteration, *in* 'The sixth Language Resources and Evaluation Conference (LREC)', Marrakech, Morocco.

Wu, F. & Weld, D. (2007), Autonomously semantifying wikipedia, *in* 'Proceedings of the sixteenth ACM conference on Conference on information and knowledge management', ACM New York, NY, USA, pp. 41–50.

Yang, Y., Bansal, N., Dakka, W., Ipeirotis, P., Koudas, N. & Papadias, D. (2009), Query by document, *in* 'WSDM '09: Proceedings of the Second ACM International Conference on Web Search and Data Mining', pp. 34–43.

Yang, Y., Teo, C. L., Daumé III, H. & Aloimonos, Y. (2011), Corpus-guided sentence generation of natural images, *in* 'Proceedings of the Conference on Empirical Methods in Natural Language Processing', Association for Computational Linguistics, pp. 444–454.

Yangarber, R., Grishman, R., Tapanainen, P. & Huttunen, S. (2000), Automatic acquisition of domain knowledge for information extraction, *in* 'Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)', Saarbriicken, Germany, August, pp. 940–946.

Yao, B., Yang, X., Lin, L., Lee, M. & Zhu, S. (2010), I2t: Image parsing to text description, *in* 'Proceedings of the IEEE', Vol. 98, IEEE, pp. 1485–1508.

Ye, S., Qiu, L., Chua, T. & Kan, M. (2005), NUS at DUC 2005: Understanding documents via concept links, *in* 'Document Understanding Conference'.

Yi, L., Liu, B. & Li, X. (2003), Eliminating noisy information in Web pages for data mining, *in* 'Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining', ACM Press New York, NY, USA, pp. 296–305.

Yih, W.-t., Goodman, J., Vanderwende, L. & Suzuki, H. (2007), Multi-document summarization by maximizing informative content-words, *in* 'Proceedings of the 20th international joint conference on Artifical intelligence', Morgan Kaufmann Publishers Inc., pp. 1776–1782.

Zajic, D., Dorr, B., Lin, J., Monz, C. & Schwartz, R. (2005), A Sentence-Trimming Approach to Multi-Document Summarization, *in* 'Proceedings of the Document Understanding Conference'.

Zhou, L. & Hovy, E. (2003), A web-trained extraction summarization system, *in* 'Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1', Association for Computational Linguistics, pp. 205–211.

Zhu, X., Goldberg, B. A., Van Gael, J. & Andrzejewski, D. (2007), Improving Diversity in Ranking using Absorbing RandomWalks, *in* 'Proceedings of the North American Chapter of the Association for Computational Linguistics on Human Language Technology', Association for Computational Linguistics, pp. 97–104.