Automatic Summarization from Multiple Documents

George Giannakopoulos[1]
University of the Aegean
Department of Information and Communication Systems Engineering
and
NCSR Demokritos
Institute of Informatics and Telecommunications
Software and Knowledge Engineering Laboratory
(ggianna@iit.demokritos.gr)

June 3, 2009

[1]Under the supervision of Prof. George Vouros (University of the Aegean), Dr. Vangelis Karkaletsis (NCSR Demokritos) and As. Prof. Panagiotis Stamatopoulos (National and Kapodistrian University of Athens)

Contents

List of Figures

List of Tables

Abstract

This work reports on research conducted on the domain of multi-document summarization using background knowledge. The research focuses on summary evaluation and the implementation of a set of generic use tools for NLP tasks and especially for automatic summarization. Within this work we formalize the n-gram graph representation and its use in NLP tasks. We present the use of n-gram graphs for the tasks of summary evaluation, content selection, novelty detection and redundancy removal. Furthermore, we present a set of algorithmic constructs and methodologies, based on the notion of n-gram graphs, that aim to support meaning extraction and textual quality quantification.

Σύνοψη

Σε αυτήν την αναφορά περιγράφεται η έρευνα που διεξήχθη στα πλαίσια του διδακτορικού μου στον τομέα της εξαγωγής περιλήψεων από πολλαπλά έγγραφα και τη χρήση υποστηρικτικής γνώσης. Η έρευνα εστιάζει στην αξιολόγηση περιλήψεων και την υλοποίηση ενός συνόλου γενικευμένων εργαλείων με εφαρμογές σε διαδικασίες επεξεργασίας φυσικής γλώσσας, μέσα από ένα σύνολο τελεστών και σχετικών μεθοδολογιών. Παρουσιάζουμε αυτές τις μεθοδολογίες στα πλαίσια της αξιολόγησης περιλήψεων, της επιλογής περιεχομένου, της αναγνώρισης νεωτερισμού και αφαίρεσης πλεονασμού. Περαιτέρω, παρουσιάζουμε ένα σύνολο από αλγορίθμους και μεθοδολογίες, με βάση τους γράφους ν-γραμμάτων, με σκοπό την υποστήριξη της εξαγωγής νοήματος και της ποσοτικοποίησης της κειμενικής ποιότητας.

Acknowledgments

The research described within this report was supported by the research and development project ONTOSUM[1], which is in turn funded by the Greek General Secretariat for Research and Technology.

I would like to deeply thank my supervisors, Dr Vangelis Karkaletsis, Professor George Vouros and Assistant Professor Panagiotis Stamatopoulos for their continuous support and patience. I also owe many thanks to Dr Sergios Petridis, Dr George Paliouras, Dr Stasinos Konstantopoulos and (nearly Dr) Ilias Zavitsanos for a set of excellent discussions and sport events that made coping with the PhD feasible.

Last but definilely not least, I would like to thank my family for their continuous support and irreplaceable influence in all the small bits that make me what I am today. I hope that the conclusion of this small step in my life is a small token of gratitude towards everything we have earned together, day by day, moment through moment, difficulty through difficulty.

[1]See also *http://www.ontosum.org/*

Ευχαριστίες

Η έρευνα που περιγράφεται στα πλαίσια αυτής της αναφοράς υποστηρίχθηκε από το ερευνητικό και αναπτυξιακό έργο ONTOSUM, που χρηματοδοτήθηκε από τη Γενική Γραμματεία Έρευνας και Τεχνολογίας.

Θα ήθελα αρχικά να ευχαριστήσω τους επιβλέποντες καθηγητές μου. Τον Ερευνητή Α΄ κο Βαγγέλη Καρκαλέτση για την επίμονη και αποτελεσματική προσπάθειά του να πατάω πάνω σε γερό έδαφος, τόσο ως υποδομή όσο και ως χρηματοδότηση, ώστε να μπορέσω απερίσπαστος να προβώ στην έρευνά μου. Τον Καθηγητή κο Γιώργο Βούρο για την απεριόριστη δυνατότητά του να διαβάζει και να διορθώνει, μέσα στο κυκεώνα των υποχρεώσεών του, καθώς και για την απαίτησή του για ποιότητα δουλειάς. Τον Επίκουρο Καθηγητή κο Παναγιώτη Σταματόπουλο, γιατί με την εμπιστοσύνη, τη στήριξη και τη συνέπειά του έχει αποτελέσει από την αρχή της πορείας μου στον χώρο της έρευνας παράδειγμα προς μίμηση και βασικό έρεισμα στο σύνολο της προσπάθειάς μου.

Θα ήθελα επίσης να ευχαριστήσω από βάθους καρδιάς όλους τους φίλους και συνεργάτες που έκαναν το παρόν κείμενο εφικτό μέσα σε τρία και μισό χρόνια: Ευχαριστώ Γιώτα, Ηλία και Τάσο, Άρη και Βασιλική γιατί χωρίς εσάς δε θα γινόταν ο Δημόκριτος ο χώρος μου για τα τρία αυτά χρόνια και γιατί θα είχα καταρρεύσει πολλές φορές χωρίς την βοήθεια και την υποστήριξή σας. Ευχαριστώ Γιώργο Παλιούρα που έβρισκες πάντα καιρό για μία εξαιρετική συζήτηση. Ευχαριστώ Σέργιο Πετρίδη που από την πρώτη στιγμή αποτέλεσες φίλο και συζητητή για όλα τα περίεργα που περνούσαν από το μυαλό μου. Ευχαριστώ Στασινέ Κωνσταντόπουλε για το χρόνο που περάσαμε συζητώντας τα νευρωνικά δίκτυα, τα DLs. Ευχαριστώ Αλέξανδρε Αρτίκη για τη βοήθεια και... τους αγώνες ποδοσφαίρου. Ευχαριστώ Κώστα Σταματάκη για την ήρεμη αποτελεσματικότητά σου. Ευχαριστώ επίσης και όλους τους άλλους που πλαισίωσαν την τριετή μου πορεία στο Δημόκριτο με τις παρουσίες τους, διανθίζοντας το χώρο με την ύπαρξή τους.

Ντίνο ευχαριστώ και εσένα για όλες τις φορές που άκουσες να μιλάω για τους... γράφους μου αδιαμαρτύρητα.

Τέλος, θα ήθελα να εκφράσω την ευγνωμοσύνη μου στην αναντικατάστατη οικογένειά μου, που ανέχτηκε μία καθ΄ όλα παράδοξη πορεία προς το όνειρό μου. Τον βράχο δύναμης και αγάπης πατέρα μου, που άντεξε να με βλέπει μέσα σε όλα τα ξενύχτια και τις δυσκολίες. Την αεί παρούσα και υποστηρικτική μητέρα μου, χωρίς την πίστη της οποίας δε θα είχα τη δύναμη να περπατήσω το δρόμο που με έφερε εδώ. Τον αδελφό μου, που με έχει κάνει τον άνθρωπο που είμαι με την απόρθητη διαφορετικότητά του και την απαράμιλλη και ανέλπιστη αγάπη του.

Chapter 1

Introduction

Since the late 50's and Luhn [Luh58] the information community has expressed its interest in summarizing texts. The domains of application of such methodologies are countless, ranging from news summarization [WL03, BM05, ROWBG05] to scientific article summarization [TM02] and meeting summarization [NPDP05, ELH$^+$03].

Summarization has been defined as a reductive transformation of a given set of texts, usually described as a three-step process: selection of salient portions of text, aggregation of the information over various selected portions, abstraction of this information to a more general level, and finally presentation of the final summary text [MB99, Jon99]. The summarization community nowadays includes a significant number of scientists with increasing interest in the multi-document aspect of summarization. Major issues towards multi-document summarization that have been identified by the researchers are the following.

- How can one detect and select salient information? How does the evaluation of salient information change when the summarization task is driven by a query?
- How can one condense or compress text preserving linguistic quality and coherence? Furthermore, how can linguistic quality and coherence be defined and measured?
- How can one assure that the final summary does not contain redundancy or repeated information, especially when multiple documents are used as input for summary composition?
- Can one develop methods that will be partially language-dependent or fully independent from a specific language?

Up to date, many summarization systems have been developed and presented, especially within such endeavours as the Document Understanding Conferences (DUC) and Text Analysis Conferences (TAC)[1]. The DUC and TAC have strengthened the summarization community and have helped in identifying tasks, problems and corresponding solutions concerning the summarization

[1]See `http://duc.nist.gov/` and `http://www.nist.gov/tac/`.

domain. Within these conferences and other related endeavours, such as NTCIR[2], SUMMAC and TREC, the summarization process has grown more specific. However, a full set of new challenges has arisen and led to the update of tasks that constitute the overall summarization process.

The summarization community appears to have moved from single to multi-text input and has also studied opinion summarization and "trend" summarization, as in the case of NTCIR. However, evaluations performed in recent years have proved that the summarization task is highly complex and demanding and that automatic summarizers have a long way to go to perform equally well to humans [Dan05, Dan06, DO08].

Furthermore, even the current methods for evaluating summaries are under heavy criticism [Jon07, CD08, GKV08]. It has been shown by a number of researchers that have faced the problems apparent in the evaluation process that evaluating different aspects of a summary text is far from being a trivial task — even if the evaluation is performed manually [VHT03, Nen06, LH02, RJB00, Mar00, SL02]. The basic questions summarization evaluation research needs to answer are:

- Which are the textual qualities of a summary, or the qualities of a good text in general?
- How do humans evaluate a summary?
- Can there be an automatic grading that would correlate to human judgement?

Within this thesis we present the results of a three year research endeavour, that has made the following basic contributions.

- A statistical, language-neutral and generic representation — named n-gram graphs — which offers richer information than widely used representations such as the vector space model. The representation is accompanied by a set of theoretical tools and an implemented toolkit for the application of the n-gram graph representation and algorithms in NLP tasks. (Part I)
- An automated evaluation system, named AutoSummENG, aiming to capture the textual quality of given summaries in a language-neutral way, by using the n-gram graph representation. AutoSummENG has achieved state-of-the-art performance, while maintaining language neutrality and simplicity. Contributing towards establishing quality-indicative measures, we propose the String Sequence Statistical Normality, which is based on the statistics of character sequences within a given text. (Part II)
- An automatic summarization system based on the use of n-gram graphs, focusing on addressing the content selection and redundancy removal problems in a language-neutral manner. (Part III)

Throughout the course of the presented research a number of important by-products were generated. Part IV of this thesis is dedicated to all the results obtained and initiatives undertaken in parallel to the conducted research. For

[2]See `http://research.nii.ac.jp/ntcir/`

instance, a part of the research was dedicated to promoting the collaboration between summarization community researchers through common frameworks and tools. Namely we present:

- The FABLE framework, aiming to support the AESOP (Automatically Evaluating Summaries Of Peers) task of the Text Analysis Conference upcoming in 2009, by providing a common framework for the integration and evaluation of summary evaluation techniques.
- The JINSECT toolkit, which is a Java-based toolkit and library that supports and demonstrates the use of n-gram graphs on a whole range of Natural Language Processing applications, ranging from summarization and summary evaluation to text classification and indexing. The toolkit is a contribution to the NLP community, under the LGPL[3] licence that allows free use in both commercial and non-commercial environments.

The first part of the thesis offers a literature overview, aiming to introduce the reader to the important problems and existing research efforts rather than to completely and elaborately review the domain. We then present the devised representation, algorithms and tools and indicate their application potential in NLP tasks. The second and third parts of this work are dedicated to the summary evaluation part of our reseach and to the summarization system we have implemented, correspondingly. The fact that the evaluation precedes the summarization system was a strategic decision at the start of this research, as it is very important to understand the qualities of a summary text, before actually producing a summary. The ordering of the sections reflects this train of thought and aims to provide the intuition for the evaluation of summarization methods presented in the summarization system-dedicated part of the report. We conclude this thesis with the discussion of secondary results and community-oriented proposals, as well as the overall conclusions.

[3]See `http://www.gnu.org/licenses/lgpl.html` for more information on LGPL.

Part I

Summarization and n-gram graphs

Chapter 2

Literature Overview and Discussion

Software or human agents accessing a multitude of data sources for satisfying specific information needs would be keen on a system that would retrieve and sum up information for them in an effective and friendly way (what ever this means). This is particularly useful for people who constantly need to be aware of latest advances in their respective discipline, such as medicine or engineering. This need drives research towards information summarization in domains ranging from news [Nen06, BM05, ROWBG05] to medical [AKS05a, ENW04] and other areas of interest [SYGH05].

What has become obvious during the last decades is the overwhelming volume of information, either being structured or unstructured, appearing in all domains of one's e-life[1]. This has been termed to be the *information overloading problem*, which indicates the huge amount of available information, the redundancy of information, as well as the existence of possible contradictions and apparent inconsistencies in information extracted from different sources. Along with the 'mushrooming'[2] of information, the initial desire to extract information was transformed into the need to acquire filtered, customized, pre-evaluated and ranked information, in ways that satisfy people's information needs: Automatic summarization could be a solution to these problems, given that it satisfies certain qualities, which are further discussed in the sections that follow.

The chapter begins with a brief overview of definitions for the summarization process (section 2.1). Following that, through a synthesis of approaches, we specify the steps that the summarization process comprises and show how these steps have been realized by existing research endeavours (section 2.2). Then, the chapter elaborates on summarization from multiple documents (section 2.4) and, in sections 2.5,2.5.1, we furthe examines in detail the incorporation of background knowledge and meta-data into the process. Specifically, section 2.5.1 discusses current research efforts towards multi-document automatic summarization with the use of background knowledge. Section 2.6 concludes by indicating existing open problems for the multi-document summarization pro-

[1]E-life here is meant to describe the human activities utilizing electronic systems of information and communication.

[2]Used in [MB97] to indicate sudden growth.

cess and describes a visionary ‘gold standard’ for systems that consult ontologies to enhance produced summaries, proposing directions on how this standard can be approximated.

2.1 Definitions

Although the summarization process seems to be a commonly understood task, appealing to our intuition, the formalization of this process and the definition of what constitutes a summary is a complex issue. In this section we examine existing definitions of summary, approaches to the summarization process and we propose a generic specification of the latter.

2.1.1 Definition and Qualities of a Summary

Sparck Jones in [Jon99] defines a summary as ‘a reductive transformation of source text to summary text through content reduction by selection and/or generalization on what is important in the source’. According to [RHM02], a summary is ‘loosely defined’ as a ‘text that is produced from one or more texts, that conveys important information in the original text(s), and that is no longer than half of the original text(s) and usually significantly less than that’. What is established, is that summarization should indeed be regarded as a reductive transformation. What is not final is the actual method by which this reduction should be performed.

There are a number of major issues that need to be taken into account, regarding the summarization process:

The source of a summary can be either single-modal or multi-modal. For instance, there can be only textual documents, or mixed documents combining text and images. The cardinality of input documents is also a characteristic of the summarization process (see also section 2.4).

The content and form of a summary has its own aspects as well. According to some researchers, a summary need not be in textual form. A textual summary would imply the need of qualities such as well-formedness, coherence, understandability and informativeness of text. However, as indicated in [RHM02], summaries may also include other modalities such as speech, images, video, or other non-, semi-, or fully-structured text representations. This indicates the necessity for advancing textual qualities to summaries with multimedia content.

The purpose of a summary to satisfy specific communication or information needs. In [Jon07] we find the purpose of a document to be specified by its use, its intended audience and a set of other parameters. The purpose of a summary is a factor that is very important when it comes to evaluating a summary.

The presentation of a summary can vary. It can provide information through various means, like coloring and structure. For instance multimedia [CNP06], and in particular treemaps, have been used to present summarized information. A treemap represents trees by using nested rectangles, where a child-node linked with its parent node is illustrated by having a ‘child’ rectangle contained within a 'parent' rectangle. Also information like the colour and the size of a rectangle is used to code other features, such as the type or the importance of a node. In [CNP06] this approach is used to provide feedback on the intensity of an extracted (*i.e.* summarized) subjective opinion, which is mapped to a rectangle in the treemap, on a specific subject (*e.g.* customer review on a specific product), along with the actual text extract expressing the opinion. We also find the notion of thumbnailing [SDC04] introduced in the context of summarizing document collections. Semantic thumbnails are defined as directed acyclic graphs, where the set of nodes correspond to document terms and the set of weighted edges reflects intra-sentential and inter-sentential significance. Sengupta et al. argue that these thumbnails are a type of human-friendly summary representation.

A thorough reference on the 'context' factors specifying the major aspects of a summary can be found in [Jon07], where three main groups of factors concerning the *input*, the *purpose* and the *output* are elaborated. According to Sparck-Jones, these factors can be used to drive both the summary extraction *and* the summary evaluation processes.

In our enquiry into the automatic summarization process we need to explore the required summary qualities. Different researchers have posed during their research a number of such qualities, which we briefly present in the paragraphs that follow. Then, we aggregate these qualities and explain them to depict an overall quality feature space aligned to what cited research has supported.

Throughout this document, we will mostly focus on the textual aspect of summaries, but without ignoring features and issues that other modalities impose. Due to this approach we may sometimes use the concepts *text* and *data* interchangeably, even though *text* will emphasize the textual aspect of a document, while *data* will emphasize the complex aspect of documents (*e.g.* images and text).

Niggemeyer in [EN00], stating that 'summaries are derived texts', discusses the 'textuality' (*i.e.* the quality of being textual) of automatic summaries, based on the notion of text well-formedness. According to Niggemeyer, for a summary to be textual, it needs to satisfy the qualities of

cohesion linguistic, syntactic and anaphoric integrity,

coherence semantic and functional connectedness, which serves communication,

acceptability the communicative ability of the text from the perspective of its addressees,

intentionality the ability of the text to convey the intention of the writer, *e.g.* exaggeration or question,

situationality the ability of the text to result into the expected interpretation within a specific context,

intertextuality the ability of the text to link to other texts, preserving the presented information) and

informativity the novelty of the textual information [Tai05]

According to Niggemeyer, summaries being produced by automatic summarization systems do not satisfy several of these qualities, which seems to be a common ground, being evidenced by other research efforts as well [Dan05, Nen06]. This means that summaries produced by automatic summarization systems do not have the traits people expect to find in a textual summary.

In the Document Understanding Conference (DUC)[3] of 2005, qualitative human-centered assessment emphasized on qualities such as grammaticality, non-redundancy, referential clarity, focus (which refers to the relation of the data to the subject of the summary), structure and coherence, as well as responsiveness [Nen06, Dan05]. Other required qualities for a summary include predefined length, focus, granularity, coherence (also called cohesiveness) and coverage as presented in [SS05]. Some of these qualities will be further discussed in the section dealing with the evaluation of summaries (section 2.3).

According to the above, we consider a list of qualitative characteristics of a textual summary. Although the list is not considered to be complete, it indicates those qualities that are being emphasized in the literature and describe in an intuitively sufficient way the qualities of a good (either manually or automatically generated) summary:

Structural well-formedness which is analysed into *length*, *grammaticality*, *cohesion*, *referential integrity* and *syntactic quality*. The *length* of the summary defines whether the transformation process was actually reductive or not, and to what degree. *Grammaticality* and *syntactic quality* represent to which extent the text conforms to the grammatical and syntactic rules of the required language. *Referential integrity* indicates whether any anaphora within the span of text is clear. In other words the sentences 'Me and John made a great discovery of an Indian manuscript and an Egyptian mummy. It was more than 4,000 years old!' have an anaphora resolution problem: Is the mummy, or the manuscript very old? Such text would score low on referential integrity. *Cohesion* is a somewhat overall measure, indicating the existence of structural integrity of the text mostly at a sentential level or at closely placed sentences. In [HMR05] we find the following informal definition of a perfectly cohesive sentence:

> 'This (perfectly cohesive) sentence is either fully related to the previous one, or clearly indicates that it addresses a new topic. The relationship of this sentence to the previous one is clear. It can stand in the given position directly after the previous sentence.'

This definition may seem to approximate what coherence (which mostly stands for a meaningful train of thought) is thought to be. The main difference between cohesion and coherence is at the level of analysis: the former refers to local integrity, usually within a few sentences, whereas coherence, emerges throughout the document as a whole. In [MH91] cohesion is described as a term for 'sticking together', while coherence refers to 'making sense'.

Informativeness which is analysed into *focus*, *informativity*, *granularity*, *coverage* and *non-redundancy*. *Focus* defines the degree to which the text

[3]More at `http://duc.nist.gov/`.

remains relevant to the required set of information. *Informativity* indicates the novelty of presented information. *Novelty* is defined in [Tai05] as a corpus-document relation, where the document contains propositions that do not exist in the corpus. *Non-redundancy* indicates that there is no repetition of information within the text.

Granularity refers to the specificity of information contained within a summary text. Granularity may be modelled by assigning a granularity value to every term of a given text, given an underlying ontology. The depth of terms within the given ontology can be an indication of specificity and, thus, granularity. For example, the phrase 'the results of the research were excellent' has lower granularity (*i.e.* more generality) than 'the results were 97 percent precision 95 percent recall after 5 well defined experiments between January and March'.

Coverage refers to the percentage of desired information from the source documents that is expressed in the summary.

Semantic quality which is analysed into *intentionality*, *intertextuality* and *coherence*. *Intentionality* indicates the success of keeping the original author's informative intention in the final summary. So, if the author wrote 'It seems inadequate to say that the mummy was very old', it is a failure to summarize by 'The mummy was very old'. *Intertextuality* indicates whether the arguments or facts in a text are well-founded on other sources, and whether the text itself can be used as a reference to support a new set of arguments or facts. Therefore, a text with low intertextuality does not have adequately justified information and cannot be used as a source. This criterion is mostly applicable in domains where justification is an issue, like research or law papers. Consider the case where some documents have been summarized, using different parts of argumentation from the source documents. Even though the argumentation of each document can be sufficient, the argumentation provided in the summary is not necessarily sufficient, because some arguments apparent in the original documents may have been omitted from the summary. This is a typical example of an intertextuality problem in the summery text.

Representativeness [Tai05] is another semantic quality, which refers to the ability of an abstract to convey the same meaning as its original sources. Representativeness can be quantified by applying reasoning to the propositions contained in the summary and the original texts. For example, one can model the meaning of a set of texts as a set of propositions. Applying the same method to a summary, one can model representativeness as the coverage of propositions found in the original texts by the propositions contained in the summary.

Finally, semantic *coherence* is the quality describing overall quality of text and is directly related to understandability (even though a highly technical text can be cohesive but not easily understandable). Coherence actually indicates whether the overall stitching of phrases, sentences, paragraphs and larger parts of text makes sense. As already stated above, it surely stands upon cohesion (see p. 428 [All87]), but refers to the overall text and not at a local level (*i.e.* within a small number of sentences/phrases).

Pragmatic quality which refers to the satisfaction of constraints implied or explicitly set by the user or by the user's environment. Such a quality is *situationality*, which helps determine whether the information given by the summary is indeed suitable for the target context (the context comprises the triple 'text, user, environment' as indicated in [EN00]). In other words, trying to use a multimedia summary in a text-only output system, or using an English summary for Chinese users will score low in terms of situationality. In [FRTF], the proposed system considers such restrictions for the output form, either text, synthesized voice or audio and video segment. Situationality mainly includes constraints on the size and the presentation of a summary. The constraint on size is presented as the 'imperative' for 'succinct description' in [YP06] to indicate the need for compactness in *data summarization.* Pragmatic quality finally also includes the usability of a given summary for a given usage requirement, which is directly connected to the purpose of a summary.

For more information on the specifics of textual quality we suggest reading [SS05, EN00, Nen06, Dan05]. Given these required qualities, we can evaluate (as will be indicated in section 2.3) any given summary. However, there is still an open question as to whether these qualities can be formalized quantitatively and, thus, be measured.

Until today, there have been various efforts to quantify and measure textual quality, often outside the summarization community. Qualities that have been quantified or approximated using automatic techniques include grammaticality [Kel00], coherence [LB05, MK04], responsiveness [Dan05, Dan06] and overall responsiveness [DO08], which is a measure including an overall judgement of quality and purpose completion (*i.e.* if the text does well in the task it was judged on).

Types of Summary

There are many types of summaries, which can be viewed from many different points of view. For a detailed overview of summary types one may consult [Jon07, Jon99, SL02, AKS05a, BDH$^+$00]. One could suppose that each summary quality (as described in section 2.1) can provide a different dimension for the categorization of summaries, but this is not true. Summary types, as suggested by the existing bibliography, do not correspond to the aforementioned qualities of texts.

Summaries, as already indicated, can be *single-* or *multi-document*, which means they are derived from a single or multiple texts. On the other hand, they can be categorized as *informative*, conveying the core data of their source(s), or *indicative*, providing a 'teaser' reading to the reader, indicative of the content.

Another aspect that differentiates summaries is their *extractive* or *abstractive* nature. Summaries of the former type, *i.e.* extractive, are entirely composed of chunks of text from the original document(s), while abstractive ones may comprise sentences that can not be found in the original text. Abstractive summaries are much more difficult to create automatically [Nen06, Dan05]. Nevertheless there are a number of approaches that offer abstractive summaries; for instance [BKM90], where a set of text-derived and world, *i.e.* common, information are combined to retrieve and express questions to answers. Another abstractive method in the context of spoken dialog summarization can be found in [RKEA00], where spoken dialog is recognized via a recognizer module and through a set of intermediate steps the information communicated is represented and integrated within the dialog context.

The true potential of abstractive summarization has yet to be discovered and the ‘non-extractive’ methods, as Sparck-Jones refers to them in [Jon07], are few. In 2002 the research community expressed its belief that abstraction has a long way to go by such statements as ‘true abstractive summarization remains a researcher’s dream’ [RHM02]. In recent research we see that the barrier distinguishing extractive from abstractive methods appears to weaken, because the meaning of ‘abstraction’ cannot be formalized easily (also see [Jon07]).

As already described in the introduction, abstraction presupposes some kind of transformation, for example filtering, mix or generalization, or different interpretation of the source data. Extractive summaries are more machine-oriented (*i.e.* easily performed and evaluated by machines) while abstractive summaries are more human-oriented. However, humans prefer (good) abstracts to (good) extracts, and this is explained by the fact that humans mostly use abstractive methods [BV04]. This kind of categorization of summaries will be further discussed in the evaluation section of the summarizing process.

At DUC a different distinction between summaries was proposed, based on summary granularity: summaries can be indicated to be either ‘general’ or ‘specific’ [Dan05]. However, this distinction is not well-defined (DUC has decided not to use this distinction after DUC 2005). It seems that the level of granularity of a summary depends on the information needs [Dan05], *i.e.* on the needs related to the summary generation. Granularity provides an example of a text quality that proved to be too ambiguous a criterion for the categorization of summaries, and has been excluded from use.

2.1.2 Specification of the Summarization Process

Let us look at, what at first sight seems to be, a naive question: if someone had me summarize, for example, a number of research papers, what exactly should I do? It seems that summarization, according to Mani and Bloedorn [MB99] is a three-step process (even though in the first footnote Mani and Bloedorn indicate that this process is only a partial consensus between contemporary researchers). Specifically, it is said that summarization consists of *analysis*, *refinement* and *synthesis* steps. Analysis results into a representation of the source text(s), refinement transforms the text into a summary representation through the selection of salient information that needs to be communicated, and synthesis renders the summary representation into natural language (or any type of surface appearance[4]).

According to the view of Niggemeyer and Wansorra [ENW04], summarization is a cognitive task involving the mental representation of ‘a body of mostly external information’, reduction of this representation to a set of most relevant (information) items, and the generation of content. The relevance of the information items in the second step refers to the relevance of information with respect to the information needs of the user. The cognitive aspect of this definition is based on the fact that the steps can be mapped to a set of empirically

[4]For more on surface appearance and Natural Language Generation see [All87, RD00]

identified, human cognitive strategies that reflect the mechanics of summarization performed by humans. Niggemeyer and Wansorra argue that, having systems that imitate people in these methods can yield results similar to those of humans. Sekine and Nobata in [SN03] have argued along the same line of thought, supporting that one should study human behaviour to devise automatic summarization strategies.

In our effort to find commonalities between the two process specifications, we may match analysis with representation, refinement with reduction and synthesis with generation. It is obvious that there is some serious overlap between different definitions of the summarization process. However, the generality of steps' description does not offer a concrete specification, but only an intuitive support on what the summarization process comprises. Of course, this generality allows most of the established existing approaches to be subsumed by the specification, which is a useful if a specification is to be widely used. However, such a specification must also be useful to methodologically comparing, implementing and further advancing the different aspects of the summarization process.

Other approaches specify summarization as a more complex process at a lower level of granularity. For example, the UAM system [TAGMS05] adopts the newer Mani definition [Man01] of (multi-document) summarization including five steps, which seem to be a more refined version of the ones in [MB99]:

- Identification of elements of information.
- Matching instances of these elements.
- Filtering of elements to keep the salient ones.
- Aggregation and generalization of the information from kept elements.
- Presentation of the results.

The aggregation and generalization step does not exist in Mani's first, three-step specification, implying the objective to produce extractive summaries. On the other hand, Niggemeyer seems to have taken into account that it is not only the selection of salient information (independently of the way this is implemented) that transforms text; there can be other types of transformation. Furthermore, the newer, five-step specification of the summarization process by Mani indicates the problem of element identification, which the previous definitions ([MB99] and [ENW04]) seem not to take into account, although it can be part of the analysis step. Also, in [MGB99] we find an elaboration of the condensing (reductive) transformation of text, which is reported to involve the following operations: selection of salient portions of text, aggregation of the information over various selected portions and abstraction of information to a more general level, as is also indicated in [Jon99]. The above indicate that, even though there is serious overlap between definitions, there is no consensus over the detailed description of the process. This lack of consensus has also been noted at footnote 1 of [MB99].

To further support this argument, we have to point that Sengupta et al. in [SDC04] describe summarization as 'the process of extracting keywords or potentially complete sentences that capture the text of the document'. In the aforementioned paper keyword extraction is described as a form of summarization process. But, is this what people think about what a summary should be like? Probably, as stated in section 2.1.1, the answer is negative, as there are a number of textual qualities that are not covered by simple keyword extraction.

Marcu in [Mar00] reduces the problem of summarizing documents to the selection of those nodes (text spans) that form the *nuclei* in the rhetorical structure of the source text. The rhetorical structure of a text has been modeled using RST, which stands for Rhetorical Structure Theory (RST) [MT87]. RST defines a framework for the description of intra-textual relations between text spans, such as elaboration, justification, concession, evidence, contrast, and others. Therefore, a text is mapped to a tree-structure based on the above relations, with each span of the text being a leaf of the tree, while relations hold inner positions. Some relations have child-nodes of equal importance, while others have child-nodes of non-equal importance. For example the relation *contrast* (*e.g.* 'I am dumb but not that dumb') has equally important children ('I am dumb', 'but not that dumb'). On the other hand, the *evidence* relation ('I am dumb. See my school grades!') does not have child-nodes of equal importance ('I am dumb' is the important part, or *nucleus*, while 'See my school grades!' is the less important part, or *satellite*). What is important about Marcu's specification of the summarization process is that no rendering step exists: the prevalent clauses are kept as-is. Thus, after locating the relations between text spans, the rhetorical structure provides a representation which leads to direct indication of salient parts of text.

A recent architecture of a general purpose summarization tool, indicative of recent thoughts on the summarization process, has been defined in [FRTF], where the three main components of the architecture are:

The Relevant Information Detector that returns a set of ranked Text Units (TU), corresponding to chunks of text.

The Content Extractor that processes linguistically the TUs detected by the previous module and selects some of them by means of the Candidates Similarity Matrix Generator (CSMG) and the Candidates Selector (CS) modules. The Extractor uses various analyses to determine the most appropriate TUs for the final summary.

The Summary Composer that creates the final summary text, using one of two approaches: lexical and semantic. The semantic approach uses semantic information to avoid redundancy, whereas the lexical does not.

Concerning the specification of the summarization process, we would like to have one that is detailed enough to indicate how a summarization system can be implemented, but also general enough, to subsume existing established definitions. We will attempt to describe summarization in a multi-layered fashion, providing an overall specification of this process, and further elaborate on its distinct stages. The specification, in order to be generic enough, must allow steps to be omitted without harming the overall function summarization process (an output summary would still be expected).

2.2 The Summarization Process

We define summarization as a reductive process that transforms information from a finite-information source to an output (surface appearance of a) representation that:

- maximizes equivalence of meaning (semantic equivalence) between the represented information and the information from the source.
- maximizes textual quality and informativeness with respect to human judgement.
- minimizes information redundancy, if such redundancy exists in the source.
- selects pieces of information indicated or implied by an information need model (or profile).

The information need model represents a specification of the information needs and preferences of the summary consumer (see section 2.2.2).

Our objective is to specify the concrete steps of the summarization process, specifying those aspects that will drive the design of summarization systems, for any modality, so that the output summary preserves required qualities. We could then check our specification of summarization, making sure that it uses and complies with knowledge from existing approaches, and that it also takes into account views of the process that are novel or uncommon to existing specifications and systems, such as the use of non-textual sources, or the production of non-textual summaries.

The overall specification of the summarization process is meant to allow for a deeper understanding of the problems one may face while developing a summarization system, of existing solutions to these problems, and of open questions requiring further research.

As Figure 2.1 depicts, we consider summarization as a multi-step process involving *domain definition*, *subject analysis and information retrieval*, *data analysis*, *feature generation and representation*, *information aggregation and fusion*, *summary representation*, *summary generation*, *summary reformulation*, and *summary evaluation* (see Figure 2.1). In brief we present each step:

Domain definition sets the context, i.e. data, information and supportive knowledge sources, to be used throughout the summarization process.

Subject analysis and information retrieval processes the information need of the user and retrieves information related to this need.

Data analysis processes the information gathered and extracts structures, relations and features of varying complexity.

Feature generation and representation is the step where extracted structures, relations and features are used to derive more informative features, that we will call *elements*. These elements may be represented by feature vectors, graphs or other such formalisms.

Information aggregation and fusion is the step that combines generated elements, so as to provide less redundant combined pieces of information.

The aggregation and fusion further aims at creating *enriched elements* that are more informative compared to each of their individual source elements.

Summary representation determines the salience of different enriched elements, selecting a subset of the latter and binding them into a coherent whole in a well-defined way.

Summary generation is the rendering of selected enriched elements into a surface appearance, *e.g.* text.

Summary reformulation is a correcting step, aiming to enhance the produced summary, so that the text is of higher quality. We have integrated this step as a separate subprocess, due to the fact that there is much work dedicated to the reformulation process as part of extractive summarization methods [MKH$^+$99, KM00, HB08, HCGL08, Nen06]. In non-extractive methods such a step can be omitted.

Summary evaluation is the step where the enhanced summary is evaluated according to a set of given metrics, in order to determine its quality. The results of the evaluation can then be used as feedback to previous steps.

Each step can have one or more inputs and one or more outputs, creating a multiple pipeline-like structure. In each step, input comprises outputs of previous step(s) as well as other supportive resources, such as domain knowledge, information repositories, linguistic resources, *etc.* Examining how this generic course of actions reflects specific approaches, some steps can be implemented using the identity function, which means they will perform no transformation or processing whatsoever. We claim that this model can describe most existing approaches on summarization, simply by mapping the techniques used in each approach to one or more specific steps in the model.

Subsequently, we introduce the steps that compose the summarization process one by one. We attempt to provide examples that indicate the specific processing at each step, indicating how the described process works. One should note that most of the cited approaches, are derived from the domain of multi-document summarization (which is the main focus of this section), but others originate from single-document summarization research, where similar work has been conducted.

What is also notable is that, although it would be useful to present evaluation methods for each intermediate step in the process, this is not possible. The reason is that in most summarization approaches we only find evaluations of the final result. These evaluations are not necessarily indicative of the success at the subprocess level. As we will discuss in part II of this work, evaluation should focus more on the intermediate methods and results to optimize the overall summarization process.

2.2.1 Domain Definition

Domain definition sets the overall domain of interest in which the summarization process will be carried out. This means that the *domain of interest (input)* leads to the selection of a set of *data sources, metadata sources and probably other information resources (output)*, that subsequent steps consult and process.

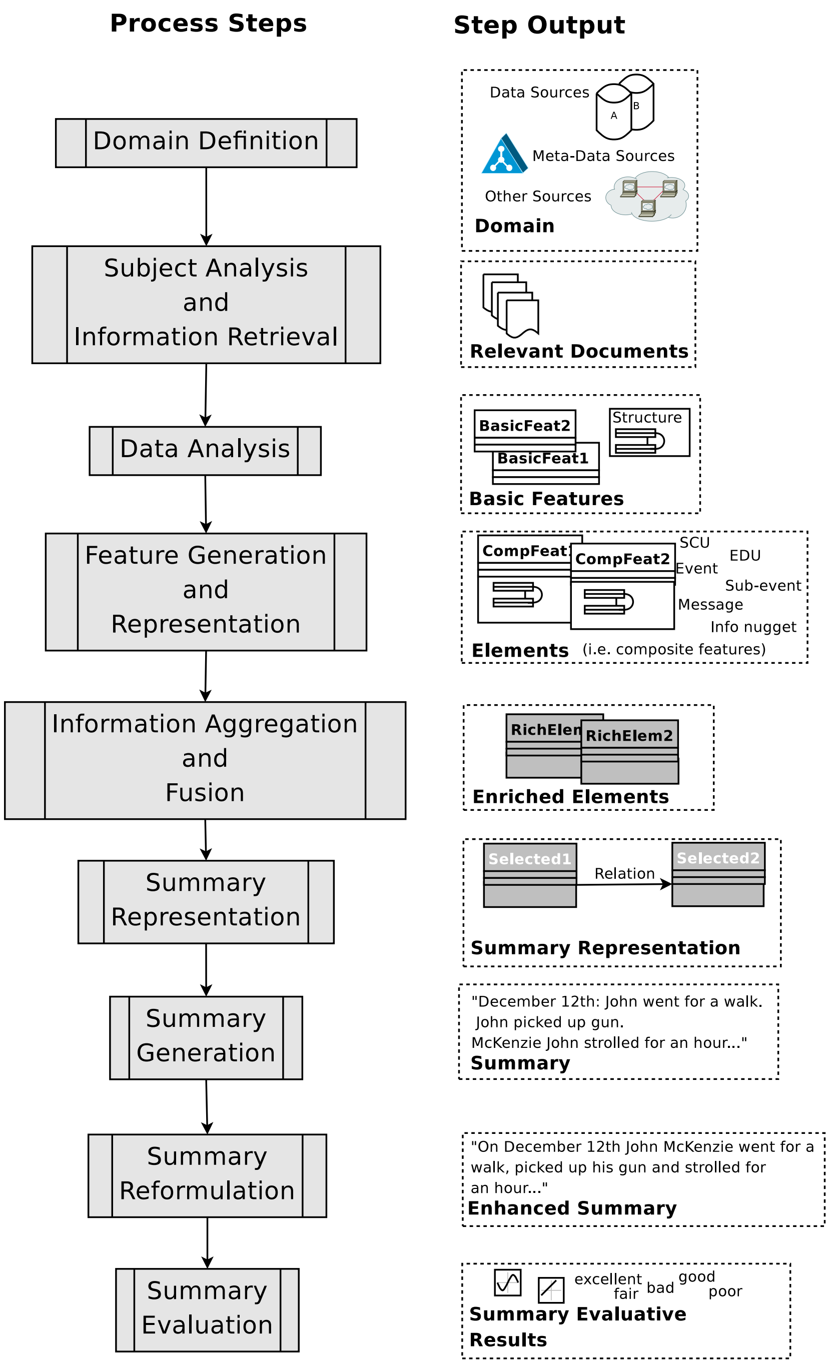


Figure 2.1: *The Summarization Process* The output of each step is also considered input for the next step. Domain data and meta-data, as well as the information need model are used throughout the process. Finally, evaluation output can be used as feedback to improve previous steps by iteration.

The documents to be summarized are being retrieved from the information sources identified, while other resources aim to provide helpful information for the summarizer to perform its tasks. The output of this step functions as input for latter steps.

- Where The Data Come From — Data sources

  Either manually or automatically, one should determine the source of documents as an early step of summarization. Systems aggregating news, for example, give the user the option to select the primary sources of news [ROWBG05]. In most systems, however, data sources are predefined (*e.g.* DUC data).

  Sources can vary in many attributes, like the subject in focus, objectivity, style (*e.g.* of writing), content type (text or multi-modal), volume of information, intention of communication (*e.g.* libel or praise, elaboration on a previous text, support of an idea, news update) and others. This original content limits the maximum informativeness or responsiveness[5] of the system, as existing systems cannot deduce new pieces of information. They only make use of what information is contained within the source documents. Therefore, the quality of the data sources may pose an upper limit in summarization effectiveness, measured by means of the different qualities introduced.

- Metadata sources

  Metadata are described in [BL97] as 'machine understandable information about web resources or other things' or 'information about information'. If, for example, we have an image (which is a piece of information), a machine understandable (*i.e.* conforming to some standards) description of the image provides the metadata for the image.

  Metadata, in general, are used to provide to the system such information or knowledge that the system can exploit to perform reasoning tasks (also see section 2.5). Another use for metadata would be the provision of grammatical, syntactical or semantic information (*e.g.* [MBF$^+$90]), to support mechanisms for textual, or other types of, analysis. Lexica and thesauri may contain metadata concerning. for instance, terms, meanings of terms, semantic relations among them, *etc.* The provided information and knowledge aid the summarization process to increase effectiveness in such tasks as selection of salient information [WL03], or analysis of meaning and verification of extracted meaning from analysed text [ENW04].

2.2.2 Subject Analysis and Information Retrieval

One of the major tasks of the subject analysis step is to specify the subject of the needed summary. This can be done by means of a short query, a set of queries, a narrative question, a relevant document, or by means of some other form of subject specification: using keywords, indicating an event, or a time duration. For indicative literature on the subject specification see Table 2.1. The above facets of specification can form parts of the consumer information need model.

[5]Responsiveness is a quality indicating how informative a summary is, given a question (see [Dan05]).

Method	*Indicative Literature*
Short query	[HMR05, CSS05, DIM05]
Set of queries	[Sag05, DNS$^+$05]
Narrative question	[TAGMS05]
Relevant document	[HS06a]
Other (keywords, time duration)	[NPDP05]

Table 2.1: Subject Analysis - Subject specification approaches

Elaborating on the information need model of the user, the model can define the user need using characteristics such as:

Length, or compression ratio indicating the required length or reduction ratio of the summary.

Source requirements indicating criteria for source selection, *e.g.* scientific journal news only.

Relevance requirements indicating criteria for the domain, the subject within a domain, or more specific content requirements (*e.g.* by means of a query).

Quality requirements referring to the summary qualities, as these have been defined above.

Representational requirements concerning the output, like output format (XML, text, image, treemap, *etc.*).

Use and purpose requirements indicating criteria, specific to the use of the summary. Each different use can actually alter the rest of the requirement set as an overall bias. For example the use of a summary as a scientific report will have different quality requirements than a news bulletin.

Other requirements defining, for example, already acquired information, so as to avoid information redundancy in a communication session.

The information need model can be augmented by other pragmatic (*i.e.* user-context dependent) constraints, complementary to the above characteristics. These may serve as a bias to the summarization subprocesses, driving the satisfaction of specific qualities. Such constraints may be:

time constraints like a requirement for fast response.

space constraints like a constraint for minimal encoding size (due to low speed network).

machine constraints indicating supported protocols like the final representation of the summary (HTML only, XML, *etc.*.).

One should notice that by referring to 'a source' we do not affect the generality of the specification: all individual sources can be considered to be merged into an ideal source, which in turns is provided as input to the summarization process.

It has been argued that users cannot effectively define or learn to define specific information needs [SJWS02], which means that 100 percent of recall (*i.e.* percent of desired information from the input that is contained in the output) and precision (*i.e.* percent of the output information that accounts for desired information from the input) of summary-contained information can only be approached asymptotically. In other words, if the user is not sure about what he wants (or he can not specify his needs in an adequate way), how can the system produce satisfactory summaries? As we argue later (section 13.1), the specification of information needs is more of an interface rather than an algorithmic issue. However, the information need model supplies a specification of how an information need can be defined and used.

In some cases, sources may be so comprehensive that some restriction has to be put, in order a system to use only data that are useful or appropriate for the purposes of the summary. For example, a system may get many interesting pieces of information from the World Wide Web. In order to avoid huge amounts of information that will make the summarization process too time-consuming, we can use subject analysis, aiming to analyze the specification of the subject (as described in the information need model), so as to retrieve only relevant pieces of information. This technique is applicable even in cases where one requires continuous summary updates evolved through time (*e.g.* on a news story and its evolution). By retrieving only relevant documents, the use of extraneous information is limited, probably resulting in a summary which is more focused on the subject of interest.

To retrieve related documents or information, it is quite common to use some kind of metric for content similarity. For instance, clustering methods (see *e.g.* [DHS01] for background on clustering methods) exploit document feature vectors (the process of feature vector creation is described in detail, along with several applicable distance metrics, in sections 2.2.4, 2.2.6) to group documents according to content similarity. The subject of interest can also be modeled accordingly, for example using a feature vector. Doing so, the document cluster that is closest to the subject in the vector space is the most relevant to the purposes of summarization (for some examples consult [RBGZ01, ZSN05]). This document clustering process locates, in other words, the best candidate subset of documents to be used as the source of information for the summarization process.

Another search and retrieval strategy is one retrieving texts based on the existence of a keyword or key-concept within the examined text. As a matter of fact, any document retrieval approach can be used to locate the required and most appropriate subset of documents. The metrics and methods used in the retrieval process can be identified within the information need model. More specifically, appropriate restrictions could be declared within the model, aiming at biasing the process towards a specific metric or retrieval and filtering method.

The subject analysis and information retrieval step outputs a set of documents that are to be used as source documents. We should note at this point that all the analysis methods that we present thoroughly in the following data analysis section, including morphological, syntactical and grammatical, semantic and pragmatic analysis can be used in this subject retrieval step as well. We do not delve further in the retrieval process, as the information retrieval domain is a very broad scientific field.

2.2.3 Data Analysis

After identifying and retrieving objects of interest, the system should analyse them to extract useful information concerning structures, relations, and features (*i.e.* attributes of the data). This step affects the summarization process in that it provides the basis for the next steps to stand upon. For example, if a stemming process is not successful, all the following steps will use wrong terms and the summary text may be misleading as to the original text or meaning. Therefore, each kind of analysis described, even though it is presented as a unitary process, aims at extracting as much useful information from the source documents, with the least amount of noise and errors inserted.

Morphological Analysis and Preprocessing

This type of analysis isolates and identifies features depending on symbols (*e.g.* letters) appearing in a document. A method finding patterns of symbols, of whatever length, is an instance of morphological analysis. Actual examples of such analysis can take into account the word morphemes appearing within a text, as in [BKM90]. Other examples are tokenization and chunking, substring analysis, stemming and matching, as well as lemmatization and matching (*e.g.* in [HMR05]).

Using tokenization and chunking the text is segmented in tokens of various sizes, according to the intuition and theoretic approach (underlying hypothesis) of the researcher. Usually we refer to tokens when we deal with words or small complexes of words (*e.g.* phrasal verbs, named entities). For longer word ensembles we usually use the term *chunk*.

Tokenization and chunking can be performed at any level[6]. One can find in existing literature, as illustrated in Table 2.2, tokens that are single words , sentences , multisentential passages , or even entire texts — even though texts are an extreme of the chunking process. It should be noted that in [FH03] there is a preliminary study that indicates inter-human disagreement in the chunking process, when trying to identify lengths of sentences indicative of an atomic event.

In substring analysis different parts of a string are used instead of the original string, *e.g.* the word 'overload' can be analysed into the representation array ('over', 'verl', 'erlo', 'rloa', 'load'). This kind of analysis offers flexibility in such techniques as matching, because one can match two words even if they are not identical. In [SHW06] such a technique is used for the indexing of concept words (lexicalization of concepts) of an ontology. Also, it is rather obvious that substring analysis can neglect noise within a string, like misspellings, as a number of substrings will match despite the error.

Stemming and lemmatization is used in many systems [ZSN05, TAGMS05, CSO07] to avoid differentiation of word forms of a single word. Stemming is the process where a word is reduced to its 'stemmed' form, removing suffix. In lemmatization the word is reduced to its lexicographic lemma, *i.e.* a canonical form of the word, containing only its identifying part, irrespective of the actual

[6]In p. 398 of [All87] we find that selecting different granularity for segmentation appeared to be a subject of controversy among researchers. It is no longer so, because varying granularity allows different levels and types of analysis.

Method	*Indicative Literature*
Word tokens	[WKB05, HMR05]
Sentence tokens	[KM00, AGK01, RBGZ01, BM05, FH03, LOWS07]
Multi-sentential tokens	[GMCC00]
Text tokens	[HS06a]
Stemming and lemmatization	[ZSN05, TAGMS05, CSO07]

Table 2.2: Morphological Analysis approaches

word form. The lemma usually corresponds to the string one would look up in a dictionary to find the meaning of a word.

All the above methods, summarized in Table 2.2, have one main common attribute: they do not infer or identify the semantics of words; they just use pattern matching, pattern extraction or similar methods as a means to identify symbolic features. These features are then used as either symbols (*e.g.* words) of the output summary, or as input to other processes aiming at capturing the semantics of the surface representation of the original texts. However, this shallow level of analysis can usually be performed with low complexity algorithms, which makes it a useful tool in most cases. On the other hand, the language-dependent nature of many parts of this analysis lessens the analysis' importance in multi-lingual domains. Furthermore, the preprocessing step which usually involves stemming and stop-word removal is of argued value [Led08, LRPN07], in conjunction to their implication for language dependency.

Within this thesis we present a set of methods for morphological analysis (section 3) based on n-gram graphs that is language-independent, which also offers high usability in a variety of summarization applications ranging from content analysis and redundancy reduction to summary evaluation.

Syntactic and Grammatical Analysis

Syntactic analysis , as in [BM05], and grammatical analysis, as in [RJZ89], both offer additional knowledge about tokens (or chunks), which can prove useful in the attempt to reach for semantics. It is very common to apply Part-Of-Speech (POS) tagging to the words of a text, as applied in [CSS05, ZSN05, HMR05], with the aim to retrieve linguistic and functional information about the words of a text. Thus, in [CSS05], the authors use gerund clauses, restrictive relative-clause appositives, and other similar phenomena as an indication of dispensability of a text chunk. Zhou et al. use POS tags in the clustering and summarizing modules of their summarizer [ZSN05].

The features extracted by syntactic and grammatical analysis are of a higher level than the morphological ones, given that they offer information concerning the role and functionality of the terms they describe. Therefore the output summary can make use of this information to create a plausible text in terms of language use; however, syntactic and grammatical information cannot ensure such qualities as cohesion, coherence or focus. Once more, dependence from language is an issue for syntactic and grammatical analysis.

Semantic Analysis

Semantic analysis is usually based on types of analysis like the ones mentioned above, coupled with metadata. So, simple string matching between the nouns of a text and the lexicalization of concepts in an ontology, qualifies as such an analysis. In [ZSN05], we find a classification of analysis methods, in the context of summarization, according to the level of semantic information extracted. This classification proposes three categories:

extractive analysis, where a word-similarity measure is used along with salience measures to decide the original sentences to appear in the output summary.

simple semantic analysis , where representations like chains (or trees) of inter-related words appearing in a document are used to detect salient chunks of text. The representation, according to Zhou et al., offers some semantic information.

deep semantic analysis, where the analysis is deeper in the sense that it extracts even more complex information, like the relations appearing in RST, again aiming at salience detection.

The *deep analysis* of Zhou et al. corresponds to our view of *semantic analysis.* Approaches using semantic analysis include [BKM90, SDC04, WL03]. In [BKM90] the authors present a system that uses three main components to get from the surface appearance of a text to a knowledge representation of its facts, which includes discourse information. In [SDC04] the analysis performed over terms' frequency in source texts assigns values to relations between terms to determine semantic connectedness and extract a graph of terms as a document summary. In [WL03] an ontology, already containing semantic information, is used to determine the salient parts of a set of documents. This method offers comparable but slightly better results than a generic statistical feature-based method presented within the same article. In [AKS05b], we find an approach where the cross-document structure analysis, based on Cross-document Structure Theory (CST) [Rad00], uses temporal semantics to locate similarities and differences between documents, motivated by the approach of Allan et al. [AGK01] and others. The basic notion behind the approach presented in [AKS05b] is that the facts (represented as what Afantenos et al. call 'messages', which are further described in section 2.2.4) comprising the description of a evolving event through time, have a time dimension that can function as an axis for comparison. This means that two fact descriptions referring to the same point in time, covering the same fact concerning an event can either declare similar or dissimilar information. To compare the two fact descriptions, one must have identified the time of reference for these descriptions, *i.e.* one must have extracted their temporal semantics.

One can argue that specific approaches extracting structural information of a text [Mar00] or between texts [Rad00, ADKK04] also perform semantic analysis. For example, this is the case in [ZSN05], where the use of rhetorical structure is described as an instance of deep semantic analysis. But is this kind of analysis semantic? It seems to be, as the intra-document or inter-document relations extracted have to do with the emergent structure of a document or corpus, given some understanding or induction of the relation semantics. In simpler words, how can you say *e.g.* if a sentence (or document) elaborates a statement in another sentence (or document), if you cannot extract, either heuristically or otherwise, the semantics of elaboration from the given text(s)?

Semantic analysis, as can be derived from the definition of semantics in [All87], should output pieces of information compliant with a predefined restriction of roles. In other words, semantic analysis uses information about which word can play which role – *e.g.* 'freedom' cannot be 'colored' but it can be 'restricted' – and extracts relations between words that indicate plausible facts.

The predefined restriction of roles is the background knowledge this step requires to complete. The validated pieces of information, on the other hand, are the output of the step. This means that this step will extract validated information, as opposed to a simple mixture of salient (*e.g.* frequently appearing) words or terms from the original documents [FRTF]. However, the complexity of semantic analysis makes this step hard to accomplish and we find that this semantic of analysis is under heavy research. On the other hand, simpler forms of semantic analysis, for example using readily available resources like the WordNet seem to be gaining in use. For example, in [LOSG06] the WordNet is used as background knowledge, along with a function of similarity between words, given existing WordNet definitions, to indicate similarity between the summary query and candidate terms in documents.

Pragmatic Analysis

Pragmatic analysis is an effort to extract real world knowledge (*i.e.* pragmatics) from the analysis of a document[7]. Such analysis should deduce knowledge of the real world (*i.e.* its rules, facts and functions). Pragmatics is the type of information Luhn, in [Luh58], referred to as 'general familiarity with the subject', when he stated that 'preparation of abstracts is an intellectual effort'. On the other hand, [All87] describes pragmatic analysis as a process of using textual context as a means to perform such tasks as anaphora resolution. In other words, in [All87] pragmatic analysis is described as a method to identify and extract information about inter-sentential relations, which allow for validity checking of statements. The distinction between 'world' and 'linguistic' knowledge has been used in [BKM90] as an integral part of the proposed KBNL system that performs deep analysis and language generation to retrieve information.

The difference between pragmatics and semantics is that pragmatics take into account the combination of both general truths and context information within a source. Pragmatic analysis can determine the validity of a fact taking into account all related knowledge, whether background, deduced, induced or reviewed during earlier time in that same analysis process. Semantic analysis, on the other hand, is not enough to handle inconsistencies appearing outside a specific level, *e.g.* a sentence. Look at the following example:

> Ilias is human and has three children. They are all green.

Using semantics, as defined in the domain of linguistics [All87], the above would be accepted, as no inconsistency would be found. Remember that semantic

[7]In 1969, Edmunson described what we call *cue words* as *pragmatic words*, and used these two terms interchangeably [Edm69]. The term *pragmatic* is not used in the same way here. However, it is true that Edmunson's *pragmatic words* usually point to real world situations or contexts.

Analysis Type	*Indicative Literature*
Morphological	[BKM90, HMR05, All87, WKB05, HMR05], [KM00, AGK01, RBGZ01, BM05], [FH03, GMCC00, HS06a, FH03], [SHW06, ZSN05, TAGMS05, CSO07]
Syntactic and Grammatical	[BM05, RJZ89, CSS05, ZSN05, HMR05, ZSN05]
Semantic	[BKM90, SDC04, WL03, AKS05b, Rad00, AGK01]
Pragmatic	[BKM90, NPDP05]

Table 2.3: Indicative approaches per analysis type

analysis takes into account only intra-sentential information (plus background knowledge about the valid combinations of terms). Thus, one would not find out that 'they' cannot be 'green'. On the contrary, pragmatic analysis would have first fathomed that 'they' are 'children' of 'Ilias' who 'is human', which means (by common sense) that they should also be human, and thus cannot be 'green'.

One should also examine carefully [NPDP05] to see that pragmatic analysis may depend on the review of existing knowledge to assess the meaning of a token within pragmatic context (which context in the case of [NPDP05] is a multi-modal one). Thus, pragmatic analysis should output a number of information nuggets that can be supported by background or inferred knowledge and would be expected to be more accurate, *i.e.* close to the real world, than semantic analysis. This is because the role restrictions apparent in pragmatic analysis would be more numerous than in semantic analysis. More restrictions mean more checking, more rejected information pieces and therefore higher precision, even though recall may be lower. However, the modelling of pragmatics and its extraction is non-trivial and requires complex methods of processing, which makes it an open domain for research.

Overall, pragmatic analysis is expected to render more precise knowledge about facts and events related to the original documents and will help produce summaries that model information in the sources more precisely. Pragmatic analysis is therefore expected to output a set of relations that abide by common-sense rules, this way correcting meaningless relations extracted by other types of analysis.

A summary of different methodologies based on the type of analysis they use is presented in Table 2.3.

Analysis Methods

At this point we present a few, quite important methods (tools) for analysis, namely *probabilistic analysis*, *cognitive analysis*, and *structural analysis*. We refer to these analyses in a separate section to acquaint the reader to their utility related to summarization. There are also many other kinds of analysis methods, which are mostly based on heuristics, and will not be elaborated here. Probabilistic, cognitive and structural analysis, can serve as basic approaches when one has to decide upon the *method* of analysis. The *types* of analysis mentioned above (morphological, syntactic and grammatical, semantic, pragmatic) differentiate analyses by their goal, and can all be accomplished by means of the following methods.

The probabilistic analysis method applies statistical methods in order to extract a set of statistical measures from input data. One such approach would be to use the frequency of terms (which dates back to 1958 [Luh58] and is used until today *e.g.* [SEKA05]) or of co-occurrence of terms as a means to identify features of a text [MKH$^+$99]. It should be noted that Nenkova in [Nen06, section 4.1.1, p. 70-74] has studied whether high-frequency of a term means that it will appear in a human written summary, and supports that *some* humans use summarization strategies informed by high frequency terms. On the other hand, she finds that the most frequent terms within a set of documents are most probable to appear in the final summary, while the ones with the lowest frequency are less probable to appear. Thus, frequency may be a measure used by human summarizers, but this is not definite. Consider that stopwords (having very high frequencies) are usually removed by preprocessing, which means that before using frequency have already applied corrective measures, based on heuristics, to improve the usefulness of the frequency measure.

Another measure similar to frequency is TF*IDF (term frequency — inverse document frequency) which is a measure of token importance, taken by the multiplication of the frequency of a term within a document, indicated as *tf*, by the inverse document frequency of the token in a reference corpus (indicated as *df*). TF*IDF appears as $tf \times \frac{1}{df}$ or $tf \times \log \frac{1}{df}$. TF*IDF achieves to assign high values to words that appear often within a text, but less often within a corpus. See [SM86] for more information on this measure, which seems to correct the effect of stopwords, but there is no definite answer about whether it correlates to human selection of words in a summary.

Another family of probabilistic tools is that of topic models, including a variation of the Latent Semantic Indexing method (LSI) [DDF$^+$90] called Probabilistic LSI (PLSI) [Hof99] and the Latent Dirichlet Allocation (LDA) [BNJ03]. These methods attempt to capture the 'latent semantics' of a text by combining terms in higher level features. These features, represented as distributions over words, use information derived from the occurrences of terms within documents to determine relations and *topics*. LSI uses the frequencies of terms in documents in a matrix to determine latent topics, defined as a linear combination of term frequency features. PLSI uses statistical methods and models the text generation process using 'classes' of words that can be used in a text to provide similar semantics. This model manages to tackle such phenomena as synonymy and polysemy when analysing a text [Hof99]. LDA uses a probabilistic generative process to model the generation of texts. This way it can infer the latent topics the words are derived from. In this case, the complex features are distributions over words and express the probability to use a given word given a latent topic. Latent topics have been used in various summarization works to determine salience and to identify the underlying topics in document sets for summarization and evaluation purposes [HMR05, SJ04].

The cognitive analysis method attempts to reformulate the extracted information, in accordance to background knowledge, into elementary facts that can be used as a repository for later phases of analysis in the summarization process. The cognitive aspect of analysis aims at mimicking the representation and reformulation methods supposedly used by humans, instead of other, similar methods. The methods are experimentally discovered, *e.g.* by monitoring expert human summarizers. In [ENW04] we find such an approach, where agents are used to perform human-derived subtasks of the summarization to finally create a summary. For example, an agent is used to locate relevant context, another to represent text into propositions and another to remove redundancy. In [SL02], Saggion and Lapalme use conceptual information in terms of concepts and relations which they have extracted from a large corpus of scientific interdisciplinary articles and summaries. Cognitive analysis is also applied in [SYGH05], where the analysis returns a taxonomy as the required cognitive model. The result is that the summary is created based on either information extraction and processing in a way similar to the human summarization process.

The structural analysis method has been used since the first steps of summarization [Edm69] and has taken advantage of the formal structure of specific types of documents. The structural components used in the summarization process may consist of sentence positions in a text [Edm69, Sag06], word positions in a sentence [Luh58], or other kinds of information inherent in semi-structured documents, *e.g.* HTML, XML documents, where the structure is defined in terms of tags and even paragraph order [Ami01]. This kind of information may be depicted as formatting variation when rendered, but relies on an underlying document structure that can be easily extracted.

In [SL02] we find an approach extracting structural information, depending on the formalisms used in technical articles concerning content order. This kind of formalism suggests that high level structure is in general *Abstract*, *Introduction*, *Previous Work* and so on, while in-paragraph text has a well-formed structure itself, with the first sentence being indicative of the paragraph meaning, and the last being a concluding remark for example. Structure can imply salience of specific information and it can also suggest sentence and content ordering in the output summary.

2.2.4 Feature Generation and Representation

Using the above mentioned types of analysis a system extracts some features either to be used as-is or to be combined into complex, more informative features. Features extracted through grammatical and syntactical analysis offer relations between words in the language model. Semantic analysis offers information about the relations between the concepts the words represent. Pragmatic analysis offers information about things that hold, not because they are directly expressed within a given text, but because they can be deduced through common knowledge and the facts within the text. It is clear that in our attempt to summarize, we need to combine principal knowledge sources into more informative ones to enrich the information we have at our disposal to form the summary.

As an example, having a grammatical analyser indicating that in the sentence 'In this paper we define the problem of using background knowledge', the word 'of' is a preposition by itself, could mean that this word is not important. Even a 'stopword' strategy, excluding common words contained in an appropriate list, would have done the same thing. Let's now see the case of the 'Lord Of The Rings' title. The 'of' in this case would again be omitted, even though it is an indispensable part of the title. If a Named Entity Recognition (NER) process is applied [FH03], it would possibly identify the whole of the title as a token and prevent the harm done. NER is usually based on simple analysis (*e.g.* pattern matching and simple list look-up), indicating informative features (*i.e.* whether a list of words is a named entity, the type of this entity, *etc.*). This is the kind of increase in informativeness we aim at by combining features and different analyses.

Feature Vectors

An aspect of the summarization step at hand is how to actually represent the features identified. It is very common to use what we call the vector space approach (for example see [TAGMS05]), where each feature may be qualified or quantified and appended to a vector as a new dimension. We will call the quantified version of the feature, serving as a vector element, a *vector feature*.

Thus, the word 'of' from the example above, using this kind of representation and having applied simple morphological / lexical analysis may correspond to the uni-dimensional vector of [of]. A grammatical analysis might represent the same word by the vector [preposition]. If we had also used a label indicating indispensability or removability of a token, then the feature vector would have two dimensions, and would be represented as [preposition, indispensable] if we referred to the 'Lord of The Rings', while it would be [preposition, dispensable] if we referred to the sentence 'In this paper we define the problem of using background knowledge'.

But how can we quantify features? One way to do it — amongst infinite ones — is to map each feature value to a numeric representation. For example preposition = 1, noun = 2, article = 3 and indispensable = 1, dispensable = 0 for the features accordingly. Then, the above vectors would map to [1,1], [1,0]. And at this point we have managed to represent features as vectors and, therefore, texts as sequences or sets of vectors.

Due to the fact that in many cases the dimensionality of the feature vector (*i.e.* the number of features) is too high (even thousands or million features) and not all features are of equal importance, some approaches apply dimensionality reduction techniques, like LSI and Independent Component Analysis (ICA) [DHS01]. Through LSI a document vector is mapped from the world (vector space) of words into the world of topics or classes. In ICA one maps the document vector to a vector in a feature space that attempts to represent (statistically) independent aspects of the text representation as different dimensions. Using these methodologies we keep what we consider to be the important features or generate new (fewer) ones that somehow represent the original features in an aggregated way.

So, what is the most indicative subset of (generated or predefined) *vector features* that would allow us to directly differentiate between desired and unwanted data? Unfortunately, this is still an open question. It is common to see some features perform better given a certain task, and worse given another. Thus, it is useful to use heuristics, trial-and-error efforts, information theoretic approaches or even compression techniques to ensure the most important features are kept. For more on feature selection the interested reader may consult [DHS01]. What should be kept in mind is that the evaluation of a feature in the summarization process is not equivalent to the evaluation of a feature in another feature-driven task, such as for example the classification task. This happens because a single, common word (*e.g.* the word 'not') may carry important semantics (negation in this example), while a statistical method may indicate that it is of no significance.

Relationships

Another approach on feature generation and representation is the one where extracted features are relationships between information units (e.g. words or sentences). In this kind of representation, features can consist of the structure of relations interconnecting textual units of information, or both the structure and the information units. An instance of such a representation is that of graphs [MB97, Mih04, Mih05, MR06]. In these graphs, units of information (entities and relations in [MB97], sentences in [Mih05]) are interconnected to form a graph based either on collocation (indicative, for instance, of context [MB97]), or on sentence similarity (indicating for instance presumed semantic similarity [Mih05, ER04a]). Trees, which are actually a subconcept of graphs, like the dependency trees in [BM05], can be used as well to represent a sentence, given the output of a syntactic parser and a set of rules. The representation of structured relationships conveys information that has been utilized either for salience indication and selection of sentences, or for reformulation of a summary. Obviously, the underlying representation, *e.g.* graph or tree or vector, does not define the semantics of the relationships. The semantics are defined by *what* is contained within this representation. However, different representations allow for the use of different tools when, for example, we want to match or cluster extracted relations.

Regarding relationships, an approach may output intra-document, or inter-document relations, as in [TM02] or [Rad00]. In the latter case, text chunks are positioned in a cube representation according to their source (*e.g.* a news site), their time of publishing (*e.g.* September 3rd) and their position within the document. This offers a time-sensitive aspect to the text representation and indicates temporal relations.

In [WKB06] authors propose the *fuzzy coreference chain extraction* methodology, which is applied to anaphora resolution and extraction of inter-document and intra-document references. The underlying intuition is that correference of noun phrases or verbal phrases in different sentences constitute indications that two sentences refer to the same entity. Fuzziness is used to manipulate uncertainty and allow for a less strict match between different references. Also 'lexical chains', first proposed in [MH91] and used in [SM02, ZSN05], can serve as features indicating relations between sentences. A lexical chain is a clustering (*i.e.* grouping) of words sharing the same meaning or being part of a relation (for example subsumption, synonymy, hypernymy) with each other. These chains provide the vocabulary that is indicative of the topics present in a text.

Other Feature Types

Beyond single words, or text spans, other types of features may also be used as well. Cue phrases[8] [Luh58] or lexical cues [CSS05] correspond to seemingly important sentences (or variably sized tokens) and serve as features to the summarization process.

Summarization content units (SCUs), defined in [Nen06], are another alternative. They are annotated, semantically adequate fragments of a text, no longer than a sentence and they are defined by humans. At DUC 2005 we find the notion of Elementary Discourse Units (EDUs) [Dan05], which are subsentential, automatically extracted fragments of a text. At TREC 2003 Voorhees et al used the concept of *information nuggets* [Voo03]. These are defined to be atomic facts (in contrast to SCUs, which are not necessarily atomic) for which the (human) assessor can make a 'binary' decision on whether it is contained in a response or not. These facts are identified based on previous research on behalf of the assessor, *i.e.* are based on human semantic preprocessing and conceptual representation.

All the information representations indicated in this section can also be used at the evaluation step. Doing so, human-oriented features and their representations are of great value towards evaluating machine-oriented approaches. That is why we also refer to types of output that can only be generated by humans.

Another representation is that of an event [AGK01, FH03], which is further analysed in messages according to [AKS05b]. A message is an atomic fact (much like the atomic event in [FH03]) concerning an event, with the latter being composed of several facts. Messages appear in the form of a predicate-like template: *e.g.*

$$\textit{performance}(\textit{entity}_1, \textit{in\_what}, \textit{time\_span}, \textit{value}_1)$$

refers to the performance of a player or team during an event. Arguments take valid values from instances of an ontology (more on ontologies in section 2.5). A seemingly similar concept exists in [SL02], where we find the term *template* describing a structure indicative of a category of predicates. The categories of predicates and the templates themselves have been created using empirical examination based on a set of assumptions relative to the structure of technical articles: they are supposed to map unstructured text to structured, semantically important pieces of information. On the other hand, Daniel et al in [DRA03] define the notion of a sub-event. This is an elementary event (*e.g.* a primitive fact, an elementary action) which describes part of a composite event. The example given by Daniel is one involving the event of the Gulf Air crash, which is decomposed in the sub-events concerning the take-off of the plane, the sub-event of something going wrong, the sub-event of the crash, the sub-event of the release of information from Gulf Air and the sub-event of the government agencies reacting. The *factoid* concept [VHT03] specifies 'atomic' pieces of information that can differentiate summaries in the way two different predicates can differentiate knowledge bases: they declare different facts, or different information about partially matching facts. These *factoids* are of various lengths, they are represented in a manner similar to First Order Predicate Logic, and they are subjective in that there is no well defined way to identify them. In

[8]See also [All87], pages 401-403, for a different view of cue phrases.

fact, the main directives on which factoid annotation was based, as indicated in [VHT03], was that a factoid could generally be represented by a simple First Order Predicate Logic predicate. Additionally, potential factoids (*i.e.* pieces of information that appear to be factoids) always appearing together within a (source) text were to be viewed as one, joined factoid, containing the union of all corresponding information.

As far as the definition of the term *event* is concerned, we find that it is usually presented more intuitively than formally [ACD$^+$98]: an event is loosely defined as 'something happenning at a specific place in a specific time'.

To recapitulate, the output of the feature generation and representation step consists of a set of features, either in vector form, or in some other, more structured representation. This representation is supposed to be a better model (*i.e.* better approximation) of the information contained in source texts than the model of simple features extracted in the analysis step. The fact that we have a better model will allow for better aggregation of the information at the next step and, therefore, result to a better summary.

2.2.5 Information Aggregation and Fusion

The step of information aggregation and fusion corresponds to the process of transforming elements of information to information chunks that are more informative when compared to the individual originally extracted elements. This increased informativeness may correspond, for instance, to a more complete view of an event, additional information about people involved and so on. Although, in an extractive summarization approach this step is implemented by an identity function (*i.e.* omitted), in abstractive techniques it is very important.

Given the output of the previous steps, we will have an, either structured or unstructured, set of features and we will refer to it as *elements* or *element set*. The information integrated within these elements could be unique, fully repeated, or partially redundant. These elements may also be structured including information about different types of relations [Rad00, MT87], like identity, paraphrasing, translation and subsumption. Some of the above relations allow a number of elements to be aggregated or filtered, to make sure that the resulting set of elements is more compact, less redundant, and equally informative. For example, finding a paraphrase relation, which is the case when part of a text is represented using different expressions, means that we can use either constituent element of the relation to convey the meaning. In this case we may choose the element with the most compact form.

Also consider the case where there is an elaboration relation, where an element is an elaboration of another, giving more specific information: 'many people' and '200 to 250 people including a 20% adults and 80% children'. In this case, according to our communication intention, *i.e.* what we aim at succeeding by communicating[9], we may merge the two in a single element, including all the information from the sources, but avoiding redundancy: *e.g.* 'up to 250 people including about 50 adults'. It should be noted that the communication intention may be part of the information need model of a user. The user may, for instance, require that the summary provides argumentation or proof for everything it states, or that the text is just a juxtaposition of phrases existing

[9]Also see *intentionality* and *situationality* in [EN00].

in the original text, aiming at content of high novelty with respect to a set of pre-existing information.

Information fusion comes in many shapes, ranging from the selection of a set of most prominent elements [FL03] and their ordering, to the addition of outlying information from other elements [MLDBGH04], to sentence fusion [BM05] or paraphrasing (see [ZLMH06] for paraphrasing detection).

In sentence fusion, as presented in [BM05], common information between sentences is identified using parse trees' alignment[10]. Then, what is called a *fusion lattice*[11] is calculated. The lattice elements are selected based on common information. The ordering in the lattice is based on the normality of the selected sentences, as well as the length of each proposed alternative. Normality of a sentence was measured upon a language model created through analysis of a corpus of texts. Generation of the text is the next step, which is actually a part of what we call summary generation (section 2.2.7). In paraphrasing, the aggregation process attempts to locate paraphrases of the same information, and select the most promising one, so that the least loss of information is conceded and redundancy is minimized. It is obvious that in this case the criterion for information fusion is not only the minimal size.

Filatova et al in [FH03] compose atomic, that is to say elementary, events by fusing previously extracted information on relationships between named entities, accounting for location, participants and time data types.

Information fusion may also offer an aggregation of information concerning numeric estimates [WKC$^+$00]. These estimates are derived via information extraction techniques from a multitude of sources, referring to specific events. Then, the numeric estimations, which can appear in either literal numeral (*e.g.* 18, 34, *etc.*.) or descriptive form (*e.g.* less that half, more than double, more than 15) are categorized according to the following categories:

- Range, indicating numeric range. For example 'ten to fifteen people'.
- Specific, usually indicating literals. For example 'ten men'.
- Incomparable, which refers to expressions like 'more than half of the region's residents'.

Then, estimates are selected according to specificity and novelty, the latter being decided by time of publication of the source document containing the estimate. Finally, these estimates are encoded by extraction of the minimum reported numeric value, the maximum reported numeric value and any estimates between these extremes. This kind of fusion, thus, aims to offer the most novel and complete set of information for a given numeric estimate.

The output of the aggregation and fusion step of the summarization process is a set of *enriched elements*, that increase informativeness and reduce redundancy of the elements generated in previous steps.

2.2.6 Summary Representation

During this step the system has to select and represent the most salient elements, producing a coherent representation of the summary to-be. The selection pro-

[10]A parse tree is a tree structure generated by means of a (syntactic) parser, and represents the (syntactic) structure of a chunk of text (see [All87] for more on parse trees).

[11]A lattice is a partially ordered set, or poset.

cess is in some way similar to the feature selection step in section 2.2.4, in that it would be ideal to keep all the important information, while filtering out the rest without ignoring the user information needs.

How can one define important information? It seems that the notion of salience is heavily related to the information need model of the summary recipient. In other words, salience is related to the subject analysis step (section 2.2.2), even though we will argue later (section 13) that there are other characteristics that identify importance, like the communication intention, the point-of-view (POV) of the summary recipient[12] and other pragmatic aspects concerning the information.

The selection of salient information also depends on the representations used in the previous steps. Therefore, in the feature vector space representation it is quite common to create groups of feature vectors based on topic clustering, *i.e.* clustering of the sentences (or documents) according to the features that are indicative of their subject. Then some kind of distance metric between a vector representative of the cluster and of the element of the selected granularity (*e.g.* a sentence), is used to get a metric of the coverage or similarity between the topic and the element. This metric can simply be the Euclidean distance, as is used in [SEKA05] for the clustering of paragraphs, or the cosine distance (*e.g.* in [Sag05]). The representative vector of the selected cluster is usually the centroid of the cluster [RJB00]. Therefore, salience of textual units, such as sentences or phrases, is determined based on the textual units' vector representation distance from the cluster representative vector.

For representations based on structural traits, like the ones using rhetorical structure, the distinction between important and less important parts of text can be inherent in that structure. For example, in RST in most cases the important part of a text element is the *nucleus* and the less important part, the *satellite*, can be omitted [Mar00]. In graph models, it is argued that the most salient elements tend to be those that correspond to nodes with the highest degree for undirected graphs or in-degree (number of edges leading towards them) for directed graphs, when elements are mapped to nodes [Mih05, MB97].

For other types of representation, like for instance the Summarization Content Units, the Elementary Discourse Units and other (already described in section 2.2.4), there seems to be no direct indication of importance. On the contrary, salience is evaluated by transforming the information to a different type of representation and using measures applicable to that representation. It would be very interesting at this point, if we could measure importance based on pragmatics. This approach will be further discussed in section 2.6.

To determine salient sentences researchers have used either *positional and structural* properties of the judged sentences with respect to the source texts, or *relation* properties of the sentences. Positional and structural properties include the sentence position in its containing text or the fact that a sentence is part of a title or abstract [Edm69, RJST04]. Relation properties of the sentences have to do with the sentence's relation to a user-specific query or to a topic appearing in a document set [CSS05, VH06, PLA$^+$06].

The properties usually used to determine sentence salience include the distance between the representation of the judged sentence and the representation

[12]The terms *recipient* and *consumer* used instead of the term *reader*, also cover the cases where the reader is a system that uses (consumes) the output.

of the document set, single document, or sentence they are matched against. A sentence is represented as a word-feature vector called a *bag-of-words* representation, e.g. [TAGMS05], where the sequence of the represented words is ignored. The vector dimensions contain such values as the word frequency or the Term Frequency – Invert Document Frequency (TF-IDF) value of a given word in the source texts. In other cases, an analysis is performed prior to the vectors' generation in order to produce vectors in a *latent topic space* [SJ04, FGFdC08], which we consider to encapsulate semantic similarity information.

More recent applications use machine learning techniques and sets of different features to determine whether a source text sentence should be included in the output summary. In that case the feature vector calculated for every sentence may include information like sentence length, sentence absolute position in the text, sentence position within its corresponding paragraph, number of verbs and so forth (e.g. see [TM02]). It has been shown that for specific tasks, like the news summarization task of DUC, simple positional features for the determination of summary sentences can offer very challenging baselines for summarization systems [Dan05]. However, this may falsely lead to the expectation that the 'first-sentence' heuristic, *i.e.* the use of any sentence that appears to be similar in content and properties to the first sentences of a set of training instances in the output summary, can be used as a universal rule. Therefore, experiments in generic text collections have to be conducted, as happens in the case of the opinion track of TAC 2008, to determine features of general use, regardless of the text genre. Moreover, one should try to test the transferability of algorithms and criteria over different languages, which is non-trivial.

In multi-document summarization, different iterative ranking algorithms like PageRank [BP98] and HITS [Kle99] over graph representations of texts have been used to determine the salient terms over a set of source texts [Mih05]. Salience has also been determined by the use of graphs, based on the fact that documents can be represented as 'small world' topology graphs [MOI01], where important terms appear highly linked to other terms.

The step of summary representation as described here relates to the content determination step of Natural Language Generation (NLG) Systems [All87], where the salient information is decided upon and selected. Later on we will find more steps than can be mapped to NLG processes.

Redundancy and Novelty

A problem mostly apparent within multi-document summarization is that of redundancy detection. Whereas salience, which is a wanted attribute for the information in the summary, can be detected via similarity to a query for example, redundancy indicates the unwanted repetition of similar or identical information. Research on redundancy has given birth to such measures as the Marginal Relevance [CG98] and the Maximal Marginal Relevance (MMR) selection criterion, which argues that 'good' summary sentences (or documents) are sentences (or documents) that are relevant to the topic without repeating information already used in the summary. The derived MMR *measure* is a generic linear combination of any two principal functions that can measure relevance and redundancy. Another approach to the redundancy problem is that of the Cross-Sentence Informational Subsumption (CSIS) [RJB00], where one judges whether the information in a sentence is contained in another sentence, that may have already been added to the summary. The informationally subsumed sentence can then be omitted from the summary without problem. The main difference between the two approaches is the fact that CSIS is a binary decision on information subsumption, whereas the MMR criterion offers a graded indication of utility and non-redundancy.

Other approaches, overviewed in [AWB03], use statistical characteristics of the judged sentences with respect to sentences already included in the summary to indicate repetition. Such methods are the NewWord and Cosine Distance methods [LAC$^+$03] that use variations of the bag-of-words vector model to detect similarity between all pairs of candidate and summary sentences. Other, language model-based methods create a language model of the summary sentences, either as a whole or individually, and compare a corresponding language model of the candidate sentence to the summary sentence model [ZCM02]. The candidate sentence model with the minimum KL-divergence from the summary sentences' language model is supposed to be the most redundant.

2.2.7 Summary Generation

This part of the summarization process consists of the steps matching the representation of a summary to the actual summary text. Of course there are approaches where the summary representation and the final text can be the same. Such approaches include extractive approaches and other text-to-text views of the summarization generation process, like the one presented in [BM05]. However, the overall structure of the generated document, even for instance the order of elements' appearance, has to be decided.

This step, along with the next one, corresponds approximately to some of the usual steps of Natural Language Generation, according to p. 491 of [All87], and specifically to the transformation of the semantic content into sentences. According to [RD00] this would include discourse planning, where the overall organization (*i.e.* ordering and structure) of the information is decided, the lexicalization process, where words are chosen to represent concepts, and the linguistic realization process, where syntax, morphology and orthography are assured. In extractive summaries, the lexicalization part of the step is usually not needed, as the actual lexicalization is the one used in the original source.

In the multi-document summarization literature there have been a number of studies concerning the ordering of salient sentences in the output summary. It has been shown that reordering can prove useful [BEM02]. In [BEM02] two ordering strategies are presented, namely the Majority Ordering (MO) and the Chronological Ordering (CO) algorithm.

Majority Ordering is based on the ordering of *themes* — themes are sets of sentences from different source documents that contain repeated information — extracted from the summary source texts to determine the proposed output sentence ordering. In particular, sentences in the summary are ordered according to how their corresponding themes were ordered within the original texts. Theme ordering, in turn, is based on the order of its sentences in original texts: if most sentences from theme T_1 were found to be after sentences from theme T_2 then in the output summary sentences from T_1 will also come after those of T_2.

Chronological Ordering, on the other hand, uses the temporal references in articles to deduce the temporal ordering of events described by the summary sentences (see also [BME99]) and orders the sentences based on this temporal dimension, as well as on their original presentation order when ties are found.

An approach based on the probability of the next sentence, given the sequence of already selected sentences in a summary is presented in [Lap03]. The described methodology represents every sentence in a given training corpus based on various set of 'informative features', *i.e.* verbs in lemmatized and non-lemmatized versions, nouns as named entities of groups like 'company', 'person' and 'location', date references and so forth. A probabilistic ranker orders candidate sentences according to the probability of seeing such a candidate sentence after the sentences that have already been selected.

A Hidden Markov Model-based alternative for sentence ordering [BL04] uses a probabilistic model to iteratively create topic clusters based of text spans with similar word distributions in the original documents. Then, it models topic transitions in source texts based on the extracted clusters and determines optimal summary ordering based on this learnt model.

Up to this point we have not talked about extractive approaches that change the extracted text, as for instance happens in [CSS05]. And how about the use of anaphora and sentence aggregation as is defined in the NLG domain? Well, this is where summary reformulation comes in.

2.2.8 Summary Reformulation

The process of reformulation as part of the summarization process is composed of corrective and enhancing transformations that can be applied to the summary text, so that the output is more cohesive and coherent. Also, this step of the summarization procedure attempts to maximize performance related to other qualities of the summary presented in section 2.1.1.

Mani et al. in [MGB99] speak of the usefulness of reformulation and implement a text revision mechanism using elimination of sentence constituents, aggregation of sentence constituents and smoothing (which is analysed into reference adjustment and reduction of text within sentence limits).

Another similar approach is found in [Nen06], where rewrite techniques for noun phrases and references to people are described. For the part of rewriting noun phrases (NPs) Nenkova suggests that the shortest of the available versions of (co-referring) NPs should be preferred in the summary. As far as it concerns references to people, the paper focuses on the 'cognitive status' of the reader to decide upon the rewrite rules of a person reference within an output summary. Specifically, Nenkova supports that a person reference should be reformatted according to whether the consumer of the summary has already read about that person (hearer-old) or not (hearer-new), and whether that person appears to be important (major) for the event at hand or not (minor). These criteria map to specific aspects of background knowledge and pragmatics that help the reformulation step.

In [ORL02] we find a very interesting taxonomy of reformulation and revision strategies, based on the *pragmatic* (*i.e.* actual) *concerns* these strategies aim to tackle. More specifically, we find five major categories of pragmatic concerns:

Discourse refers to problems in inter-sentential relationships or sentence-to-document relationships. Such problems would be indicated by seemingly incoherent, or out-of-place sentences with respect to adjacent ones, or by the document structure.

Identification of entities mainly refers to anaphora resolution problems in the summary.

Temporal refers to correct temporal ordering and relationships between events. One should not ignore the reformulation aspect of temporal ordering, which is a serious issue especially in multi-document summarization [ORL02, Rad00].

Grammar problems are mostly due to erroneous juxtaposition of sentences.

Location, setting refers to problems in the indication of the spatial context of an event. For instance, having a summary that indicates the Olympics of 2004 took place in Corinth instead of Athens is such a problem.

Otterbacher et al. uses these concerns as a guide to problems where reformulation and revision can be applied [ORL02]. Thus, the aforementioned list of concerns indicates the set of problems reformulation should address.

From a different point of view, in [KM00] we find the reformulation problem to be modeled as a compression problem assuming a noisy channel. The noisy channel approach considers the summarized version of a sentence to be the original (desired) signal and all additional (unwanted) information to be the noise. The problem is to get the noise-free (clean) signal, *i.e.* the summary text. There is no actual representation of the initial text (other than the text itself), and the aim is to remove as many words as possible, without harming the integrity of the text. In the statistical version of the process, a number of statistically determined transformations are applied on the original text to achieve compression. The most probable compressed versions of sentences for a given parse tree probabilistic model derived from a training corpus are used to form the final text.

In a second approach presented in [KM00], we find the compression process to be modeled by a rewriting process of the original text into reduced parse trees. The parse tree is reduced, according to [KM00], to a smaller version of itself by using a sequence of shift-reduce-drop operations, using a decision-based model. In this case too, there are only operations at a shallow level of processing.

What the described summarization step aims to offer is an enhanced version of the initial output summary, optimized over a set of desired qualities. The judgement of how well the output summary conforms to quality rules is performed by the summary evaluation step.

2.3 Summary Evaluation

The step of summary evaluation in most automated methods is very important, because it allows us to identify errors and reiterate or reformulate certain aspects of the process to optimality. While this is common ground, the notion of automatic evaluation is not. For some time now, the domain of automatic evaluation of summaries was only superficially addressed, because many of the required summary qualities could not be automatically measured. Therefore, human judges have been widely used to evaluate or cross-check the summarization processes [MB97, ACD+98, Dan05]. This is obviously a fine way to perform evaluation. But what about automatically computed measures? Which is the feature space describing desired and required qualities of a summary? Can all the desired qualities be measured so as to facilitate comparison between performances of summarization systems? And what about cases where humans themselves have different opinions on the quality of a summary; what should a machine decide? Within this work there will be references to both manual and automatic evaluation methods, in order to indicate the benefits and drawbacks of each approach.

In our quest to answer the above stated questions we need to make sure that we consider all the specific categorizations of evaluation methods. Indeed, an evaluation process may be specified to be either intrinsic or extrinsic (*e.g.* [MB97, VHT03]). Intrinsic evaluation operates on the characteristics of the summary itself, trying for example to capture how many of the ideas expressed in the original sources appear in the output. On the other hand, extrinsic evaluation decides upon the quality of a summary depending of the effectiveness of using the summary in a specific task. For example, when using summaries, instead of source texts, to answer a query and expecting that the results will be equally well to those derived from source texts, we have an extrinsic evaluation case. On the contrary, using a *gold standard* summary, *i.e.* a human-generated summary viewed as the perfect output, and estimating the similarity of the summary to the gold standard, this is an intrinsic evaluation (*e.g.* [LH02]).

Sparck Jones in [Jon07] argues that the classification of evaluation methods as intrinsic and extrinsic is not enough and proposes an alternative schema of evaluation methods' classification. This schema is based on the degree to which the evaluation method measures performance, according to the intended purpose of the summary. Therefore, defining new classes that elaborate on the definitions of extrinsic and intrinsic, Sparck Jones classifies evaluation methodologies as:

- semi-purpose, *e.g.* inspection of proper English.
- quasi-purpose, based on comparison with models, *e.g.* n-gram or information nuggets.
- pseudo-purpose, based on the simulation of task contexts, *e.g.* action scenarios.
- full-purpose, based on summary operation in actual context, *e.g.* report writing.

The higher in the above list an evaluation method is mapped, the more it appeals to the notion of intrinsic, while the lower it maps the more it would be considered extrinsic.

In [BDH+00] we find a comment (part 3.4) referring to intrinsic evaluation, where the authors suggest that 'only humans can reliably assess the readability and coherence of texts'. This statement indicates the difficulty of that kind of evaluation. But do humans perform perfect in the evaluation of summaries? And what does *perfect* account for?

The qualities of a summary, as we have already discussed in section 2.1.1 include:

- structural well-formedness measures, involving length, grammaticality, cohesion, referential integrity and syntactic quality.
- informativeness measures, involving focus, informativity, granularity, coverage and non-redundancy.
- semantic quality measures, involving intentionality, intertextuality and cohesiveness.
- pragmatic quality measures, corresponding to situationality and usability.

Humans tend to be able to identify good texts, in a qualitative manner. There is an issue of how to make human assessors grade the quality of a text in uniform and objective ways (see for instance [VHT03, LH02] for indication of the problem). At this point numerous efforts have been attempted (*e.g.* [Nen06, RJB00, Mar00, SL02]) all of which pointed out the inter-judge agreement problem. In other words, there seems to exist only statistical similarity measures that indicate the well-formedness of summaries. People tend to have similar, but surely not too similar opinions. This led to looking for subjective measures correlated to human subjectivity. In other words, if our measures behave similarly to human evaluation, we will have reached an adequate level of acceptance for our (automatic) quality measures. In [LH02] partial inter-judge agreement is illustrated among humans, but it is also supported that, despite the above, human judgements generally tend to bring similar results. Thus, perfection is subjective in the abstractive summarization process, which means that we cannot identify the perfect summary: we can only identify good enough summaries for a significant percentage of human assessors.

Then, what about other alternatives? Even though we find evaluation measures similar to recall and precision from information retrieval (*e.g.* [Mar00, section 9.2.2] and [LH02]), these measures seem to be rather inadequate and difficult to use in an automated evaluation process. The problem is that we do not know the best means to compare summary semantics to the original semantics of the source documents: shall we look for same words, sentences or shall we look for the 'actual' meaning? And if we shall work using the meaning, how can we *automatically* compare the meaning of two different pieces of text, if we cannot analyse or represent it uniquely and with clarity?

An approach found in [Nen06] makes use of a human-based evaluation process, named *pyramid evaluation*, which consists of a multiple step process. This process tries to identify the segments of the original text, from which pieces of the summary are semantically derived. In other words, the method makes use of a supposed (and argued) mapping between summary sentences and source documents, where summarization content units (SCUs) are identified. We remind the reader that SCUs are minimal units of informative ability that also appear in the summary output. According to the number of human judges agreeing on the origin of an SCU (*i.e.* the text span that corresponds to the SCU), the SCUs are assigned weights, corresponding to pyramid layers. Thus, the SCUs higher in the pyramid are supposed to be the most salient pieces of information in the original sources. A summary is then evaluated by locating the SCUs present in the summary output and using a summing function to account for the weights. Doing so, two measures are defined: the pyramid score, which corresponds to precision, and the modified pyramid score, which corresponds to recall. Nenkova argues that the above evaluation process can suppress human disagreement and render useful results. Pyramid evaluation was also applied in DUC and TAC, and the use of a new set of directives for evaluators in DUC 2006 provided better results than DUC 2005 [PMSG06], though not reaching the effectiveness of automatic methods. This indicates that manual evaluation methods can be highly dependent on the instructions given to the evaluators.

Measuring Correlation – Evaluation Method Performance

In the automatic evaluation of summarization systems we require automatic grades to correlate to human grades. The measurement of correlation between two variables provides an indication of whether two variables are independent or not. Highly correlated variables are dependent on each other, often through a linear relationship. There are various types of correlation measures, called *correlation coefficients*, depending on the context they can be applied. Three types of correlation will be briefly presented here, as they are related to the task at hand:

- The Pearson's product moment correlation coefficient reflects the degree of linear relationship between two variables[13]. The value of Pearson's product moment correlation coefficient ranges from -1 to 1, where 1 indicates perfect positive correlation and -1 perfect negative correlation. Perfect positive correlation indicates that there is a linear relationship between the two variables and that when one of the variables increases, so does the other in a proportional manner. In the case of negative correlation, when one of the two variables increases, the other decreases. A value of zero in Pearson's product moment correlation coefficient indicates that there is no *obvious* correlation between the values of two variables.

- The Spearman's rank correlation coefficient [Spe06] performs a correlation measurement over the ranks of values that have been ranked before the measurement. In other words, it calculates the Pearson's product moment correlation of the ranking of the values of two variables. If two rankings are identical, then the Spearman's rank correlation coefficient will amount to 1. If they are reverse to each other, then the correlation coefficient will be -1. A value of zero in Spearman's rank correlation coefficient indicates that there is no obvious correlation between the rankings of values of two variables. It is important to note that this coefficient type does not assume linear relation between the values, as it uses rankings.

- The Kendall's tau correlation coefficient [Ken62] relaxes one more limitation of the previous methods: it does not expect subsequent ranks to indicate equal distance between the corresponding values of the measured variable.

The above correlation coefficients have all been used as indicators of performance for summary systems evaluation [Lin04, Nen06]. To clarify how this happens, consider the case where an *automatic evaluation method* is applied on a set of summarization systems, providing a quantitative estimation of their

[13]The linear relationship of two correlated variables can be found using methods like linear regression.

performance by means of a grade. Let us say that we have assigned a number of *humans* to the task of grading the performance of the same systems as well. If the grades appointed by the method correlate to the grades appointed by humans, then we consider the evaluation method good.

Automatic Evaluation

Other, more morphologically and statistically oriented approaches of summary evaluation are those inherited from the domain of information retrieval and are namely the ROUGE/BE measures. More specifically, the 'family' of BE/ROUGE[14] [HLZF05, Lin04] evaluation frameworks, uses statistical measures of similarity, based on n-grams of words[15], although it supports different kinds of analysis, ranging from n-gram to semantic [HLZF05]. The intuition behind the BE/ROUGE family is that, for two texts to have similar meaning, they must also share similar words or phrases. Automatic methods like ROUGE, BE can be, somewhat surprisingly, more closely correlated to human judgement on responsiveness [Dan05, HLZF06] than human-based processes, *e.g.* the pyramid evaluation.

Basic Elements (BE) are considered to be 'the head of a major syntactic constituent' and its relation to a single dependent. BEs are decided upon in many ways, including syntactic parsing and the use of cutting rules [HLZF05]. BEs can be matched by simple string matching, or by more complex matching methods, like semantic generalization and matching, according to the proposed framework. According to this approach BEs in summary match to those in initial sources. The intuition underlying this approach is that locating minimal units of information from the initial source into the summary identifies similarity of meaning. The ability of this framework to use different matching operators of various complexity, appears to allow for the handling of paraphrase and similar phenomena.

In [HLZF05] we find that ROUGE is a specific branch of the BE approach, where BEs are word unigrams (*i.e.* single words) or n-grams of a higher order (*i.e.* with more than one words). Thus, ROUGE is a BE framework instance using word identity as a matcher between BEs. The size of the n-grams, as well other factors (like allowing gaps between n-grams) are best described in [HLZF05], specifying different types of ROUGE. ROUGE and BE have been found to correlate significantly to human assessors' judgements according to [Dan05] and [HLZF06] (Table 2.4), and ROUGE has been used in DUC for quite some years.

The responsiveness score of DUC and TAC provides, as Dang states in [Dan05], a 'coarse ranking of the summaries for each topic, according to the amount of information in the summary that helps to satisfy the information need expressed in the topic statement, at the level of granularity requested in the user profile'.

Grammaticality and Fluency

Other than the responsiveness of texts, there has been some research concerning the grammaticality and fluency of texts. Grammaticality is the quality of conforming to a specific grammar. Fluency, on the other hand is considered to be

[14]See also [PRWZ01] for the BLEU method on machine translation.

[15]We remind the reader that *N-grams of words* are groups of *words* with N elements. *N-grams of characters* are groups of *characters* with N elements.

Table 2.4: Correlation To Responsiveness in DUC 2005

Method	*Spearman*	*Pearson*
BE	0.928	0.975
ROUGE	0.951	0.972

a measure related to text production or reading rate — *i.e.* fluent writers write more quickly that less fluent ones [CH01] and fluent readers read faster than non-fluent ones. However, the notion of fluency has also been used to describe well-formed, easily understood text [MDWD07]. Fluency is therefore a derived quality given the qualities we have described in section 2.1.1, given that the reader has good knowledge of the text language.

Researchers aim at quantifying these qualities in a set of different approaches. In the research quest for evaluating fluency and grammaticality, the works of various linguists and philosophers have provided the foundations and a constant source of research. N. Chomsky for many years has delved into the notion of grammaticality [Cho55, Cho05]. Considering statistical as well as non-statistical aspects of grammaticality, it has been argued that there can be a binary decision for the grammaticality of a text segment, or a graded decision.

Recent research towards measuring grammaticality has treated grammars as a set of constraints that are either realized or not within a text. There have been distinctions between soft and hard constraints, referring to how important a constraint is to the acceptability of a clause [Kel00, SK05].

Much of the existing work has been based on Optimality Theory (see [PS04]), which declares that the output language is based on a procedure that uses a 'candidate analysis' generation function, called *GEN* and a *harmonic* evaluation function, called *H-eval*. The GEN function is part of a Universal Grammar that generates candidate alternatives for the analysis of the input, while the H-eval function exploits well-formedness rules of a given language. The methodology using GEN and H-eval describes a loop between the two components, until no analysis generated by GEN can give better harmonical results measured by H-eval.

In [BHR06] authors describe an approach based on Property Grammars, which is also a constraint-oriented syntactic formalism. The method applies constraint weighting, which has been used in other works as well [SK05]. Highly grammatical text chunks have a low number of violations for a given set of evaluated properties. The output is a *grammaticality index* that is shown to correlate to human acceptability evaluations.

From the domain of machine translation, the X-Score evaluation process [HR06] computes the frequency of Part-Of-Speech tags and syntax tags in the target language for a given model (reference) corpus called 'fluency corpus'. The same taggers apply tags to terms in the evaluated text. The assumption used by the authors of X-Score is that the fluency score of a text should be linearly dependent on the frequencies of tags in the target language. A prediction function is estimated based on the fluency corpus and a team of human judges; then, the prediction function is applied on the frequencies of tags of any evaluated text, returning the estimated fluency index of the evaluated text.

In [MDWD07] the prediction of acceptability is viewed as a machine learning problem, where the output of a set of parsers is used as input to a learner, trying to discern human from machine generated sentences. Then, the distance of evaluated texts' representations from the support vectors learned provide a metric that correlates to human judgments of fluency.

Studies that analyze human summaries in order to understand the summarization process have been conducted in the past, revealing the abstractive nature of human summarization. In [Jin02] the section on Corpus Analysis indicates that a number of sentences in human summary texts had no equivalent in the source documents. Furthermore, most of the sentences that indeed had an equivalent in the original texts were transformed and combined in various ways to form the summary.

2.3.1 Evaluation of Evaluation Measures

As an indication of how important the problem of evaluation appears to be, there are articles reporting on methods for the evaluation of evaluation measures. In [AGPV05] for example, a framework concerning the evaluation of evaluation metrics is proposed, which is based on some assumptions relating human performance to the performance of automatic summarization methods. The argumentation for the framework implies that we consider evaluation metrics to be 'good' when they can differentiate human from automatic summaries. In other words, if an automated method creates a summary that is as well as a human summary and an evaluation metric cannot tell the difference, then the evaluation method is considered bad. Even though this approach seems very pessimistic concerning the future of automatic summarization, it addresses two important issues: objectivity of an evaluation method (*i.e.* independence from corpus) and the criteria of a good evaluation.

The AESOP (Automatically Evaluating Summaries of Peers) task of upcoming TAC 2009 aims to focus on summary evaluation techniques, as well as on their evaluation under a common set of tools.

2.4 Performing Summarization over Multiple Documents

In our study so far, we have described the summarization process without distinguishing whether the input comprises a single document or multiple documents. In other words, it appears that we suggest that whether we use multiple sources, or a single source for summarization, the problems we face will be the same. This, unfortunately, is not true.

According to several researchers (*e.g.* [GMCC00, LH01, ZBGR02]), there are problems that are found in the multi-document version of summarization and not in its single-document one:

Redundancy. Many sources describing the same topic are much more prone to repeating information. As we have already indicated, redundancy has been dealt with by means of Maximal Marginal Relevance (MMR) [CG98] and Cross-Sentence Information Subsumption [RJB00].

Maximal Marginal Relevance is a metric that takes into account relevance and novelty of a document, combining relevance and novelty linearly to provide what Carbonell and Goldstein [CG98] refer to as *relevant novelty*.

The metrics of relevance and novelty are not predefined in [CG98], which indicates the generality of the approach. A document has high MMR if it is relevant to a user query (as expressed in the information need model in our case) and, at the same time, if it does not offer redundant information with respect to an existing set of information chunks (in [CG98] these chunks are documents).

Cross-Sentence Information Subsumption, on the other hand, is described as the relation holding between two sentences when the information content of a sentence a (denoted as $i(a)$) is contained within $i(b)$ of another sentence b. In that case a becomes *informationally redundant* and the content of b is said to *subsume* that of a: $i(a) \subset i(b)$. When such a relation holds, a can be omitted as redundant. It is important to note that there is no indication of how $i(x)$ can be measured. In [RJB00], authors use the Cluster-Based Relative Utility (CBRU) (also see [ACD+98]) in analogy to the relevance part of the MMR metric above.

Coherence. Extractive approaches for multi-document summaries face serious textual quality deficiencies. This means that it is very difficult to correctly combine parts of text from different sources, while bridging the gap between authoring style, intention of writer, and so forth. This problem is very evident when, for instance, evaluating anaphora resolution on the final summary. Another important aspect of this problem is sentence ordering in the output summary [BEM02]. In [HLY+06] authors prove that the strategy of sentence ordering can offer improvement to (mainly bad) summaries. As *abstractive* systems evolve, one would hope to see improvement in summary coherence. However, due to the embryonic state of such systems, coherence is still an issue.

Intertextuality. The integrity of references in texts, as well as the correctness of the author's train of thought and argumentation, cannot be easily transferred by simple extraction of pieces of text from different documents on the same topic. It requires abstraction and validation to make sure that the output preserves the intertextuality of the original set of texts. The problem is that both abstraction and validation have to be further investigated to provide fruitful summarization results, as they require pragmatic analysis and inconsistency detection.

Differences and similarities. Several texts may have different, often partly opposite views of a single topic or event [Rad00, AGK01, AKS05b, MB99]. For example the case where a document reports on an accident having hundreds of dead people, while another document reports the same accident to have tenths of dead people, indicates an inconsistency that is rarely encountered in a single text. Such relations have been described in the Cross-document Structure Theory, as it has already been reported.

Temporal dimension. The time interval that a group of texts has been published is within a span of time for multi-document summarization. This is in contrast to the single-document summarization, where a document is positioned at a specific point time. This dimension reflects a series of arising problems, such as temporal alignment between reference time and publication time, as well as topic evolution through time [Rad00, GMCC00, ADKK04, ORL02].

Evaluation. Humans have major disagreements as to what a multi-document summary should include. That is due to the fact that for people, multi-document summarization is an abstractive process [VHT03, Nen06]. However, recent research attempts to substitute abstraction with extraction and reformulation ([Nen06]), which appears to be a promising direction in that it can be based upon existing extractive approaches but improve on their results.

User interface. The user interface has been handled by some researchers as an important aspect of multi-document summarization [SDC04, CNP06], indicating the need for a more informative way to render multi-document summaries. Aspects of multi-document summaries like the source of an output sentence should be taken into account, as well as aspects concerning the expression of user needs (*i.e.* information need model). Issues concerning the user interface problem are further discussed in section 2.6.

As it can be understood, these problems have to be addressed by a framework for multi-document summarization. Our proposed definition of the summarization process takes into account these distinguishing aspects (as has been indicated in every step of the process), while not lacking in generality.

2.4.1 State-of-the-art Performance

How well do current systems perform in the multi-document summarization task? This is a question that is rather difficult to answer. What is certain, is that there is still much room for improvement.

The evaluation carried out in DUC 2005, where the given task was indeed a multi-document summarization task, given an information need model (as we have defined it above in section 2.2) showed that most systems did quite well in subproblems of the tasks [Dan05]. More specifically, the redundancy problem, as well as the grammaticality one were faced adequately well, having several systems averaging around the grade of 'barely acceptable'. On the other hand, referential clarity, focus and especially structure and coherence fared badly. The overall results indicate that, though much progress has been done, research has many more problems to face. What is important is that systems tend to improve significantly [Dan06], both on content-related and focus measures, leaving however room for even more improvement over readability and coherence issues.

It is probable that the use of background knowledge will import the semantics required to enhance the efficacy of current systems. But what is the nature of this knowledge and how can it be represented?[16]

2.5 Background Knowledge

We view background knowledge to be that part of real-world knowledge which is inherent in humans, and we aim to transfer it into automated systems to sup-

[16]Readers already familiar with the topic of knowledge representation can skip the following section, which is introductory to the domain.

port and aid their function or evaluation. In a syntactic parser the rules used to identify syntax consist background knowledge. In a dictionary application, the actual dictionary used for look up is background knowledge. In the summarization process, the use of nouns and verbs only in order to extract salient sentences is based on background knowledge and namely on the fact that nouns and verbs carry the main meaning of a sentence.

2.5.1 Using Prior Knowledge

Semantically rich information has during the years been embedded into automatic summarization systems using either rules that describe patterns of language corresponding to rhetorical and discourse phenomena (*e.g.* [Mar00]), heuristics, or more complex representations. In [BKM90] the representation of world and text-derived knowledge uses the CyCL representation language. In the same article we find a proposal to differentiate linguistic from world knowledge (pragmatics), which is achieved using a *lexical database* and a *knowledge base*. This distinction is important in that it implies different aspects of knowledge, requiring different handling.

Recent articles investigate the use of ontologies as a more effective way to handle non-obvious latent information. Niekrasz et al. in [NPDP05] use what is described as a Multimodal Discourse Ontology (MMD): this is a knowledge base that represents the concepts and relations of 'communicative actions performed during multi-modal discourse', *i.e.* speech, gestures, *etc.*. The same approach exploits the so-called Temporal Knowledge Base, which maintains uncertain (speculative) and incomplete probabilistic information, and supports learning of new facts by supplementary information and reinforcement. The use of ontologies here is being used to support filling partial information, which is later completed based on reasoning and slot-filling through information extraction.

Ontologies could also be used to incrementally extract information from a text, so that instances of expected entities are identified. These entities can also be used in the summary. For example, if an ontology has defined a transition event as a tuple containing a medium, an agent, a source and a destination, then a system may gather this information from various parts of source text, or even different texts, and give the complete piece of information for surface representation.

In [ENW04] we find a corpus-based ontology supporting the summarization task performed by agents. An *agent*, according to [VKB$^+$06], is 'an entity that perceives its environment with the help of sensors, is part of this environment, performs reasoning tasks on its' representation of the 'environment and acts upon the environment with the help of action mechanisms (effectors) to achieve some targets'. In [ENW04] the agents used are *software agents*, which are programs operating within a system context. More on agents can be found in [VKB$^+$06]. Each agent, in [ENW04], implements a (rather) trivial human cognitive subprocess of summarization. For example the 'context' agent checks whether a specified document is relevant to a selected domain. At this point the ontology is used to provide the key concepts of the domain. Each agent has its own use for the ontology, according to the former's purpose.

Finally, lexica and thesauri are two specific cases of resources specifying knowledge related to the human-intended meaning of words, synonymy, homonymy, opposition as well as other relations, as for instance in WordNet [MBF$^+$90].

Lexica and thesauri have been extensively used to support the summarization process. Recently, other resources such as Wikipedia are used to aid in the summarization process by expanding user queries [Nas08].

2.6 Conclusion and Motivation

Automatic summarization is moving from single-document extractive methods towards multi-document non-extractive methods. Simple heuristic methods are giving way to more complex methods, that often use machine learning techniques to determine salience of information. At times much background knowledge is used in terms of ontologies and thesauri to support the process, even though we have no direct evidence of the merit of such approaches. Finally, the evaluation of summaries offers a very broad research domain that may be directly linked to the optimization of the summarization process itself.

One should never neglect the tight relation between Information Extraction (IE), Natural Language Processing (NLP) and MDS, which allows research in one field to nourish breakthroughs in the other. The required deduction of meaning from documents, allowing different uses of the information contained therein and in other sources of knowledge, seems to require a multitude of steps forward from different disciplines. Researching representation may offer effective means to combine information and evaluate information using a common set of algorithms.

The analysis given in the previous sections of this study leads to a number of questions and directions that research can focus on. The main directions indicated can be categorized as follows:

- Representation. Is there a kind of text representation that can be generic enough to take part in different steps of the summarization process? Can it be scalable to represent relations of various order? Can this representation be language-neutral?
- Knowledge representation in the summarization process. How can we use background knowledge within automatic summarization?
- Textual Quality. How can we measure textual quality or aspects of it in a language-neutral way?

2.6.1 Representation

Throughout this research we faced a number of very interesting research endeavours that used all kinds of language features, from specific patterns of words to syntax and grammar, to extract and combine information from source documents. However, many of these methods induced noise or could not even be determined as useful or not, as in the case of stopword removal. This led us to devise a representation that would not rely on the underlying language, but it would be able to retain language characteristics even indirectly.

The representation we devised was that of the n-gram graph, presented in section 3. The n-gram graph has a number of desired traits, including generic application, language neutrality and higher expressiveness than feature vectors when expressing relations between chunks of text. Furthermore, it is based on the assumption that humans group individual symbols into small sets, which in turn constitute higher complexity symbols. This grouping and scalability has offered the basis upon which our research on representation was set.

2.6.2 Knowledge Representation in the Summarization Process

Ontologies, which have been utilized to a degree in current research, can have a much more significant role in the summarization process. Thus we propose that ontologies are used as a uniform representational method for all resources supporting the actual summarization process, like thesauri, lexica, conceptual taxonomies, look-up lists or indices. This indicates the need to produce mapping methods from existing representations to ontologies, and brings forward the question of combining different aspects of knowledge into an overall representation containing the mixture of information. In other words, it would seem useful to combine a thesaurus, a lexicon and a taxonomy under a unified structure. What possibilities that approach could offer and what its limitations would be should be the focus of future research.

To use knowledge in summarization, one can map concepts of an ontology to chunks of text, aiming to represent the text's semantics. This way tasks like content selection and redundancy detection could be performed in the semantic and not the surface level of a text. This notion we have developed within out summarization system (part III), trying to bridge the gap of the surface to the semantic representation. However, in our work we have used the WordNet resource, without making use of the reasoning potential contained in a formal ontology.

2.6.3 Textual Quality

It was illustrated throughout this whole section that both the definition of textual measures of quality, as well as their quantification are difficult problems. Even worse, even if we manage to quantify textual qualities, as has already happened in various works, it is non-trivial to automatically measure these quantities.

To that aim we developed a language-neutral evaluation system, presented in part II, that manages to approximate a set of qualities required for the evaluation of summaries.

Chapter 3

The N-gram Graph Representation

In the domain of natural language processing, there have been a number of methods using *n-grams*. An n-gram is a, possibly ordered, set of words or characters, containing n elements (see Example 3.0.1). N-grams have been used in summarization and summary evaluation [BV04, LH03, CS04]. In the automatic summarization domain, n-grams appear as word n-grams, as happens in the ROUGE/BE family of evaluator methods [HLZ05, Lin04].

Example 3.0.1 Examples of n-grams from the sentence: *This is a sentence.*
Word unigrams: this, is, a, sentence
Word bigrams: this is, is a, a sentence
Character bigrams: th, hi, is, s_, _a, ...
Character 4-grams: this, his_, _is_, ...

3.1 Graph-based Methods and Graph Matching

Graphs have been used to determine salient parts of text [Mih04, ER04b, ER04c] or query related sentences [OER05]. Lexical relationships [MR06] or rhetorical structure [Mar00] and even non-apparent information [Lam05] have been represented with graphs. Graphs have also been used to detect differences and similarities between source texts [MB97], inter-document relations [WKB06], as well as relations of varying granularity from cross-word to cross-document, as described in Cross-Document Structure Theory [Rad00]).

Graph similarity calculation methods can be classified into two main categories.

Isomorphism-based Isomorphism is a bijective mapping between the vertex set of two graphs V_1, V_2, such that all mapped vertices are equivalent, and every pair of vertices from V_1 shares the same state of neighbourhood, as their corresponding vertices of V_2. In other words, in two isomorphic graphs all the nodes of one graph have their unique equivalent in the other graph, and the graphs also have identical connections between equivalent nodes. Based on the isomorphism, a *common subgraph* can be defined between V_1, V_2, as a subgraph of V_1 having an isomorphic equivalent graph

V_3, which is a subgraph of V_2 as well. The *maximum common subgraph* of V_1 and V_2 is defined as the common subgraph with the maximum number of vertices. For more formal definitions and an excellent introduction to the error-tolerant graph matching, *i.e.* fuzzy graph matching, see [Bun98].

Given the definition of the maximum common subgraph, a series of distance measures have been defined using various methods of calculation for the maximum common subgraph, or similar constructs like the Maximum Common Edge Subgraph, or Maximum Common Induced Graph (also see [RGW02]).

Edit-distance Based Edit distance has been used in fuzzy string matching for some time now, using many variations (see [Nav01] for a survey on approximate string matching). The edit distance between two strings corresponds to the minimum number of edit character operations (namely insertion, deletion and replacement) needed to transform one string to the other. Based on this concept, a similar distance can be used for graphs [Bun98]. Different edit operations can be given different weights, to indicate that some edit operations indicate more important changes than others. The edit operations for graphs' nodes are *node* deletion, insertion and substitution. The same three operations can by applied on edges, giving *edge* deletion, insertion and substitution.

Using a transformation from text to graph, the aforementioned graph matching methods can be used as a means to indicate text similarity. A graph method for text comparison can be found in [TNI04], where a text is represented by first determining weights for the text's terms using a TF-IDF calculation and then by creating graph edges based on the term co-occurrences. In our method, no term extraction is required and the graph is based directly on the text, without further background such as a corpus for the calculation of TF-IDF or any other weighting factor.

Our method represents texts by using character n-grams positioned within a context-indicative graph. We remain language independent, while allowing for different types of the same word, and try to capture high order relations between words (*i.e.* 'neighbor of a neighbor' and sequence information), The graph that we construct, which holds the aforementioned qualities, we shall call the *n-gram graph*.

3.2 Representation

We now provide the definition of n-gram, given a text (viewed as a character sequence):

Definition 3.2.1 *If* $n > 0, n \in \mathbb{Z}$, *and* c_i *is the i-th character of an l-length character sequence* $T^l = (c_1, c_2, ..., c_l)$ *(our text), then*
a character n-gram $S^n = (s_1, s_2, ..., s_n)$ *is a subsequence of length n of* $T^l \iff \exists i \in [1, l-n+1]\colon \forall j \in [1, n]\colon s_j = c_{i+j-1}$. *We shall indicate the n-gram spanning from* c_i *to* $c_k, k > i$, *as* $S_{i,k}$, *while n-grams of length n will be indicated as* S^n.

The meaning of the above formal specification, is that n-gram $S_{i,i+n-1}$ can be found as a substring of length n of the original text, spanning from the i-th to the $(i + n - 1)$-th character of the original text. For example if the text is the following sentence:

```
Do you like this summary?
```

then the bigram $S_{1,2}$ is the sequence {'D','o'}≡'Do' for display purposes.

The length of an n-gram n, is also called the *rank* of the n-gram.

3.2.1 Extracting N-grams

If we choose to extract the n-grams (S^n) of a text T^l, the (elementary) algorithm is indicated as algorithm 1. The algorithm's complexity is linear to the size $|T|$ of the input text T.

```
Input: text T
Output: n-gram set $SS^n$
// T is the text we analyse
1 $SS^n \leftarrow \emptyset$;
2 for all i in [1,length(T)-n+1] do
3     $SS^n \leftarrow SS^n \cup S_{i,i+n-1}$
4 end
```

Algorithm 1: Extraction of n-grams

The algorithm applies no preprocessing (such as extraction of blanks, punctuation or lemmatization). Furthermore, it obviously extracts overlapping parts of text, as the sliding window of size n is shifted by one position and not by n positions at a time. This technique is used to avoid the problem of segmenting the text. The redundancy apparent in this approach proves to be useful *similarly to a convolution function*: summing similarities over a scrolling window may prove useful if you do not know the exact centre of the match between two subparts of a string. In the case of summary evaluation against a model summary for example, the extracted n-grams are certain to include n-grams of the model summary, if such an n-gram exists, whereas a method where the text would be segmented in equally sized n-grams might not identify similar n-grams.

Example 3.2.2 *Application of our method to the sentence we have used above, with a requested n-gram size of 3 would return:*
{'Do ', 'o y', ' yo', 'you', 'ou ', 'u l', ' li', 'lik', 'ike', 'ke ', 'e t', ' th', 'thi', 'his', 'is ', 's s', ' su', 'sum', 'umm', 'mma', 'mar', 'ary', 'ry?'}
while an algorithm taking disjoint n-grams would return
{'Do ', 'you', ' li', 'ke ', 'thi', 's s', 'umm', 'ary'} (and '?' would probably be omitted).

It is important that segmentation is performed carefully in order to reduce redundancy, without losing information on important sequences of characters. Consider the case where we match character n-grams between two segmented texts. In the given example, the fact that the word 'summary' has been broken down into three disjoint n-grams may cause a mismatch, or not match at all, of the word 'summary'. For n-grams of higher length, or *rank* as it is called, the effect of information loss in the case of a careless segmentation may prove more deteriorating, if we consider two n-grams to match if and only if they are exactly the same. Perhaps other methods of string comparison, like substring comparison, may decrease this loss. However, the method proposed in this thesis uses simple string matching between n-grams.

In the case of the word n-gram extraction, the text is considered to be a word sequence (as opposed to character sequence). The text has been preprocessed, using a simple tokenizer based on punctuation and blanks, to determine word boundaries. However, it is important that this 'simple' tokenizer deprives the method of its complete language neutrality, if used. Therefore, we will prefer the character n-gram version to the word n-gram version, if they produce equivalent results.

The segmentation process by itself, even if one uses our approach, does not keep information concerning the relative position of n-grams in the original text; it only extracts n-grams. What this means is that we do not know if the n-gram 'Do' is next to the n-gram 'you', or not. Thus, words (n-grams) that comprise what is called a 'collocation', *i.e.* that when found together possess a meaning that is not simply a concatenation or composition of each separate word meaning [MS99], will lose their connection when extracted. This is where the graph representation comes in.

3.2.2 The N-gram Graph as a Text Representation

The *n-gram graph* is a graph $G = \{V^G, E^G, L, W\}$, where V^G is the set of vertices, E^G is the set of edges, L is a function assigning a label to each vertex *and to each edge* and W is a function assigning a weight to every edge. The graph has n-grams as its vertices $v^G \in V^G$ and the edges $e^G \in E^G$ (the superscript G will be omitted where easily assumed) connecting the n-grams indicate proximity of the corresponding vertex n-grams (also see Figure 3.2). The edges can be weighted by the distance between the two neighbouring n-grams in the original text, or the number of co-occurrences within a given window. We note that the meaning of distance and window size changes by whether we use character or word n-grams. The labeling function L for edges assigns to each edge the concatenation of the labels of its corresponding vertices' labels in a predefined order: for directed graphs the order is the order of the edge direction while in undirected graphs the order can be the lexicographic order of the vertices' labels. To ensure that no duplicate vertices exist, we require that the labelling function is an one-to-one function.

More formally:

Definition 3.2.3 *if $S = \{S_1, S_2, ...\}, S_k \neq S_l$, for $k \neq l, k, l \in \mathbb{N}$ is the set of* distinct *n-grams extracted from a text T^l, and S_i is the i-th extracted n-gram, then $G = \{V^G, E^G, L, W\}$ is a graph where $V^G = S$ is the set of vertices v, E^G is the set of edges e of the form $e = \{v_1, v_2\}$, $L : V^G \to \mathbb{L}$ is a function assigning a label $l(v)$ from the set of possible labels $\mathbb{L}$ to each vertex v and $W : E^G \to \mathbb{R}$ is a function assigning a weight $w(e)$ to every edge.*

In our implementation, the edges E are assigned weights of $c_{i,j}$ where $c_{i,j}$ is the number of times a given pair S_i, S_j of n-grams happen to be neighbours in a string within some distance D_{win} of each other. Since, probably, not all distances are of importance, and thus two n-grams within a distance of 150 characters probably have no actual relation, we take into account only a window around S_i in the original text, to determine which S_j is useful. The vertices v_i, v_j corresponding to n-grams S_i, S_j that are located within this parameter distance D_{win} are connected by a corresponding edge $e \equiv \{v_i, v_j\}$.

We have tested three methods, concerning how the n-gram graph can be constructed, based on how *neighbourhood* between adjacent n-grams is calculated in a text. In general, a fixed-width window of characters (or words) around a given n-gram $N_0 \equiv S^r, r \in \mathbb{N}^*$ is used, with all characters (or words) within the window considered to be neighbours of N_0. These neighbours are represented as connected vertices in the text graph. The edge connecting the neighbours is weighted, indicating for example the distance between the neighbours or the number of co-occurrences within the text. Based on different types of windows, we can use:

The non-symmetric approach A window of length D_{win} runs over the text. If N_0 is located (*i.e.* begins at) at position p_0, then the window will span from $p_0 - D_{\text{win}}$ to $p_0 - 1$, taking into account *only preceding* characters or words. Each edge is weighted by the number of co-occurrences of the neighbours within a given window of the text.

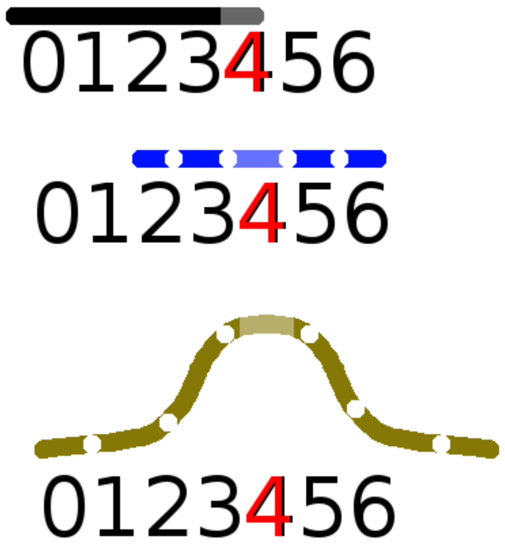
The symmetric approach A window of length D_{win} runs over the text, centered at the beginning of N_0. If the n-gram we are interested in is located at position p_0, then the window will span from $p_0 - [\frac{D_{\text{win}}}{2}]$ to $p_0 + [\frac{D_{\text{win}}}{2}]$, taking into account *both preceding and succeeding* characters or words. Each edge is weighted based on the number of co-occurrences of the neighbours within a window in the text.

The Gauss-normalized symmetric approach A window of length $3 \times D_{\text{win}}$ is placed over the text, centered at the beginning of the current n-gram, N_0. If N_0 is located at position p_0, then the window will span from $p_0 - \lfloor\frac{3\times D_{\text{win}}}{2}\rfloor$ to $p_0 + \lfloor\frac{3\times D_{\text{win}}}{2}\rfloor$ (where $\lfloor x \rfloor$ gives the integer part of x), taking into account *both preceding and succeeding* characters and words. However, in this case the *distance* of a neighbour n-gram to the current n-gram is taken into account. In other words, an n-gram N_1 with distance d_1 from the beginning of N_0, positioned at p_0, is considered to be 'less of a neighbour' than n-gram N_2, positioned at distance $d_2, d_2 < d_1$ from p_0. Therefore, each edge is weighted based on the number of co-occurrences of the neighbours within the text *and* the neighbours' distance at each occurrence. Also, the Gauss-normalized symmetric approach takes into account neighbours *outside* the given window size D_{win}, to a full distance of $3 \times D_{\text{win}}$. This distance was selected given the fact that this accounts for 99.99% of the mass under the Gaussian distribution, given that we consider a standard deviation of D_{win}[1]; that is to say, n-grams outside that distance have almost no effect. Thus, it is better in terms of complexity to just ignore those outliers.

Figure 3.1 provides schematic representations of the three approaches. The numbers indicate adjacent n-grams, which can either be word n-grams or character ones. The line over a number indicates that the n-gram has been taken into account as a neighbour. In the third part of the figure, the bell-shaped

[1] This can be easily derived by using the probability mass function of the Gaussian distribution: $\text{pdf}(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp-\frac{(x-\mu)^2}{2\sigma^2}$. See also [DHS01, Appendix A]

Figure 3.1: Different types of n-gram windows (top to bottom): non-symmetric, symmetric and Gauss-normalized symmetric. N-gram *4* is the n-gram of interest.

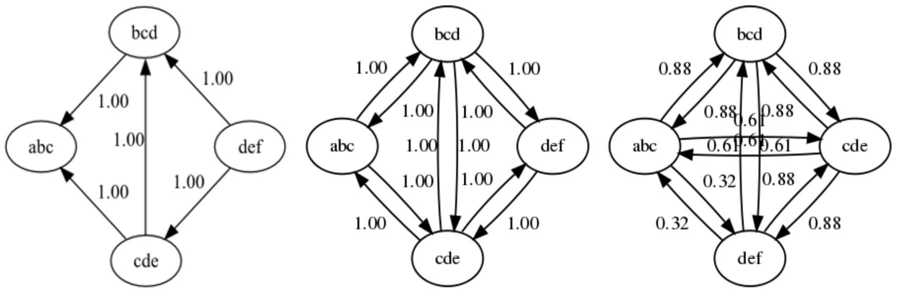


line indicates the different weights assigned to different distances from the n-gram positioned at p_0. The current n-gram, also called the *n-gram of interest*, is indicated by the emphasized number in the figure. We found, through a set of experiments, that the most promising approach was the symmetric one, which may indicate that there is indeed a maximum distance, outside which relations do not hold. The algorithmic complexity for the extraction of edges is $O(|T| \times D_{\text{win}})$, because for every n-gram of interest we extract a number of neighbours based on the D_{win} parameter.

Following this method of representation, we have reached a point where we have kept some information for a determined n-gram length n and distance D_{win}. It is non-trivial, though, to choose a single $\{n, D_{\text{win}}\}$ pair, that can be optimal for n-gram extraction, independent of the text: if one chooses a very low value for n, then the relation between different n-grams can be taken into account only by augmenting the D_{win} parameter. However, in the case of a high D_{win} value, given that we only take into consideration whether the n-grams are neighbours and not their actual distance, may prove detrimental to the information we keep. In other words, if our D_{win} is 50, then a neighbour by 1 character will be considered equally close to our current n-gram to a neighbour with a distance of 50 characters.

If n, on the other hand, is too high, then the information we gather for each text will be extremely redundant and will definitely cause consumption of more memory, as well as make the application of any process with a graph size-based complexity more time-consuming. This happens because there are much more unique 10-grams than 2-grams in any selected (adequately long) text of a

Figure 3.2: Graphs extracted from the string *abcdef*, based on the three types of windows (left to right): non-symmetric, symmetric and Gauss-normalized symmetric. The n-grams are character n-grams of rank 3. The D_{win} parameter has a value of 2.



natural language, like English or Greek. Furthermore, the number of vertices of the graph G^n will increase exponentially to the rank n of n-grams[2].

In order to tackle these problems we take into consideration n-grams of *various ranks*, with a rather small maximum distance between them. What is considered to be 'rather small' can be calculated by statistical analysis of the corpus used in any given task, as will be shown in section 7. The low D_{win} value avoids the pitfall of considering 'neighbouring' n-grams that are far apart within the original text. However, the selection of an optimal n-gram rank range $[r_{\min}, r_{\max}]$ for a given task proved to be an issue worth investigating. We discuss this issue in section 7.

3.2.3 Attributes of the n-gram graph

The n-gram graph has a set of inherent differences to existing representation methods, such as the vector space model, nevertheless keeping useful traits of existing methods:

Indication of n-th order relations. The graph in itself is a structure that maintains information about the 'neighbour-of-a-neighbour'. This means that if A is related to B through an edge or a path in the graph and B is related to C through another edge or path, then if the proximity relation is considered transitive we can deduce that A is related to C. The length of the path between A and C can offer information about this indirect proximity relation. This can be further refined, if the edges have been assigned weights indicating the degree of proximity between the connected vertices.

This attribute of the n-gram graphs primarily differentiates them from the vector space representation. If one creates a feature vector from an n-gram graph, where the *edges* correspond to dimensions of the feature vector, the indirect relation between vertices is lost. Keeping information on an n-th order relation between a number of l distinct elements in a

[2]The grammar of each language does not allow all combinations of alphabet characters, and thus the possible 5-grams of a language with 26-letter alphabet are not 26^5, but somewhat lower. See also [MS99, sections 2.2.1-2.2.2].

vector will require adding a dimension for every relation between two elements. Therefore, even though one may construct a vector in a way that it will represent the same information as the n-gram graph, there is high complexity in the transformation process. However, further study of the equivalence of the two representations would be interesting to perform in the future.

Parametrically determined generality of the model. The n-gram graph, if viewed as a language model, recognizes sequences of symbols based on the way they are found to be neighbours. The set of sequences the model accepts is actually the solution to a problem of constraint satisfaction [Tsa93], where the constraints are based upon the neighbourhood relation. The laxity or strictness of these constraints is based upon the D_{win}, L_{min},L_{MAX} parameters. We plan to research the exact expressive power of the n-gram graph as a language model in future work.

A constraint satisfaction problem is a triple $< V, D, C >$, where $V = \{v_1, v_2, ..., v_n\}$ is a set of variables, $D_{v_i} = \{D_1, D_2, ..., D_n\}$ is the domain of values for $v_i \in V$, and $C = \{c_1, c_2, ..., c_m\}$ is a set of constraints. Every constraint is in turn a pair $< t, R >$, where t is a tuple of variables and R is a set of equally sized tuples of values. An evaluation of the variables is a function e from variables to domains, $e : V \rightarrow D$. Such an evaluation satisfies a constraint $< (x_1, \ldots, x_n), R >$ if $(v(x_1), \ldots, v(x_n)) \in R$. A solution is an evaluation that satisfies all constraints.

In our case, in the simplest n-gram graph form where we use unigrams only, $L_{\text{min}} = L_{\text{MAX}} = 1$, we consider the variables of our constraint satisfaction problem to be matched to the letters of a given text T of length $|T|$ in characters, i.e. $n = |T|$. Then, the edges of the graph form the constraints on the variables, based on the counts of co-occurrences of unigrams. The value of D_{win} changes the number of edges in the graph, thus changing the number of constraints per variable. This gives us the specification of the texts the given n-gram graph can generate as a language model.

Language neutrality. When used in Natural Language Processing, the n-gram graph representation makes no assumption about the underlying language. This makes the representation fully language-neutral and applicable independent even of writing orientation (left-to-right or right-to-left) when character n-grams are used. Moreover, the fact the the method enters the sub-word level has proved to be useful in all the cases where a word appears in different forms, *e.g.* due to difference in writing style, inflection of word types and so forth.

Generic application. The n-gram graph need not be applied to text representation only. The existence of small-length symbols, like character n-grams, that are related via a neighbourhood relation can also be found in such applications as image analysis and data mining or bioinformatics.

The graph in itself is a mathematical construct with no predefined semantics, suitable for the representation of relations. This generic utility makes it usable whenever there is a *meaningful neighbourhood relation*. Furthermore, the amount of information contained within a graph is only restricted by the data one desires to annotate graph edges and vertices with. Within the scope of this study we will propose a set of uses for the n-gram graph aiming to offer intuition on possible further applications.

In order to use the n-gram graph representation we need a set of algorithms and tools to perform such tasks as graph matching and similarity measurement, graph merging and graph difference. The algorithms, as will be shown in the following chapters, can then be applied to fuzzy matching between strings, to generating document set models, to classifying documents, to evaluating summaries and a number of other applications.

Chapter 4

N-gram Graph Operators and Algorithms

Given two instances of n-gram graph representation G_1, G_2, there is a number of operators that can be applied on G_1, G_2 to provide the n-gram graph equivalent of union, intersection and other such operators of set theory. For example, let the merging of G_1 and G_2 corresponding to the union operator in set theory be $G_3 = G_1 \cup G_2$, which is implemented by adding all edges from both graphs to a third one, while making sure no duplicate edges are created. Two edges are considered duplicates of each other, when they share identical vertices[1].

The invention of the operator is actually non-trivial, because a number of questions arise, such as the handling of weights on common edges after a union operation or the keeping of zero-weighted edges after the application of an operator. In our implementation we have decided that the union operator will average existing edge weights for every common edge into the corresponding new graph edge. Zero-weighted edges are treated like all other edges, even though they have the same semantics as the absence of an edge (*i.e.* the vertices are not related).

In all the presented algorithms we work with edges only, because the way the graphs have been created does not allow isolated vertices to exist. Throughout this section we consider that information is contained within the relations between n-grams and not in the n-grams themselves. Therefore, our minimal unit of interest is the edge, which is actually a pair of vertices. This use of graphs implicitly defines the properties of the graph's vertices, based on what we do with the corresponding edges.

Overall, we have defined a number of operators most of which are functions from $\overline{\mathbb{G}} \times \overline{\mathbb{G}}$ to $\overline{\mathbb{G}}$, where $\overline{\mathbb{G}}$ is the set of valid n-gram graphs of a specific rank. In other words, unless otherwise noted, the operators function upon graphs of a given rank and produce a graph of the same rank. The operators are the following.

- The similarity function $\text{sim} : \overline{\mathbb{G}} \times \overline{\mathbb{G}} \to \mathbb{R}$ which returns a value of similarity between two n-gram graphs. This function is symmetric, in that

[1]The identity between vertices can be a customized calculation. Within our applications two vertices are the same if they refer to the same n-gram, *i.e.* they share the same label. Thus, identity can be checked by simple string matching.

sim(G_1, G_2) = sim(G_2, G_1). There are many variations of the similarity function within this study, each fitted to a specific task. The common-ground of these variations is that the similarity values are normalized in the $[0, 1]$ interval, with higher similarity values indicating higher actual similarity between the graphs.

- The containment function *contains*, which indicates what part of a given graph G_1 is contained in a second graph G_2. This function is expected to be asymmetric. In other words, should the function indicate that a subgraph of G_1 is contained in another graph G_2, we know nothing about whether the inverse stands.

  In the herein proposed implementations of the containment function values are normalized in the $[0, 1]$ interval, with higher values indicating that a bigger part of a given graph G_1 is contained in a second graph G_2. We consider a graph to be contained within another graph if all its edges appear within the latter. If this is not the case, then any common edge contributes to the overall containment function a percentage inversely proportional to the size of G_1, so that the function value indicates *what part* of G_1 is contained within G_2.

- The merging or union operator $\cup$ for two graphs, returning a graph with all the edges, both common and uncommon, of the two operand graphs. The edges are weighted by the average of the weights of the original edges. The intuition behind averaging the edges' weights is that the merged graphs should be equally close to the two operand graphs in terms of edge weights; this is the effect of an averaging function.

- The intersection operator $\cap$ for two graphs, returning a graph with the common edges of the two operand graphs, with the averaged weights of the original edges assigned as edge weights. Once more, the averaged weights make sure we keep common edges and their weights are assigned to the closest possible value to both the original graphs: the average.

- The delta operator (also called *all-not-in* operator) $\triangle$ returning the subgraph of a graph G_1 that is not common with a graph G_2. This operator is non-symmetric, *i.e.* $G_1 \triangle G_2 \neq G_2 \triangle G_1$, in general.

- The inverse intersection operator $\bigtriangledown$ returning all the edges of two graphs that are not common between them. This operator is symmetric, *i.e.* $G_1 \bigtriangledown G_2 = G_2 \bigtriangledown G_1$.

Finally, the empty graph $\emptyset$ is considered to be a graph with no edges. The size of a graph is the number of its edges and is indicated as $|G_1|$ for a graph G_1.

4.1 Similarity and Containment

To represent *e.g.* a character sequence or text we can use a set of n-gram graphs, for various ranks, instead of a single n-gram graph. To compare a sequence of characters in the form of a chunk, a sentence, a paragraph or a whole document, we apply variations of a single algorithm that acts upon the n-gram graph representation of the character sequences. The algorithm is actually a similarity measure between two n-gram graph sets corresponding to two texts T_1 and T_2.

To compare two texts (or character sequences in general) T_1 and T_2 *e.g.* for the task of summary evaluation against a gold standard text, we need to compare the texts' representations. Given that the representation of a text T_i is a set of graphs $\mathbb{G}_i$, containing graphs of various ranks, we use the *Value Similarity (VS)* for every n-gram rank, indicating how many of the edges contained in graph G^i are contained in graph G^j, considering also the weights of the matching edges. In this measure each matching edge e having weight w_e^i in graph G^i contributes $\frac{\mathrm{VR}(e)}{\max(|G^i|,|G^j|)}$ to the sum, while not matching edges do not contribute (consider that for an edge $e \notin G^i$ we define $w_e^i = 0$). The *ValueRatio (VR)* scaling factor is defined as:

$$\mathrm{VR}(e) = \frac{\min(w_e^i, w_e^j)}{\max(w_e^i, w_e^j)} \tag{4.1}$$

The equation indicates that the *ValueRatio* takes values in $[0, 1]$, and is symmetric. Thus, the full equation for *VS* is:

$$\mathrm{VS}(G^i, G^j) = \frac{\sum_{e \in G^i} \frac{\min(w_e^i, w_e^j)}{\max(w_e^i, w_e^j)}}{\max(|G^i|, |G^j|)} \tag{4.2}$$

VS is a measure converging to 1 for graphs that share both the edges and similar weights, which means that a value of VS $= 1$ indicates perfect match between the compared graphs. Another important measure is the *Normalized Value Similarity (NVS)*, which is computed as:

$$\mathrm{NVS}(G^i, G^j) = \frac{VS}{\frac{\min(|G^i|,|G^j|)}{\max(|G^i|,|G^j|)}} \tag{4.3}$$

The fraction $\mathrm{SS}(G^i, G^j) = \frac{\min(|G^i|,|G^j|)}{\max(|G^i|,|G^j|)}$, is also called Size Similarity. The NVS is a measure of similarity where the ratio of sizes of the two compared graphs does not play a role.

The overall similarity $\mathrm{VS}^O$of the sets $\mathbb{G}_1, \mathbb{G}_2$ is computed as the weighted sum of the VS over all ranks:

$$\mathrm{VS}^O(\mathbb{G}_1, \mathbb{G}_2) = \frac{\sum_{r \in [L_{\min}, L_{\mathrm{MAX}}]} r \times \mathrm{VS}^r}{\sum_{r \in [L_{\min}, L_{\mathrm{MAX}}]} r} \tag{4.4}$$

where VS^r is the VS measure for extracted graphs of rank r in $\mathbb{G}$, and $L_{\min}$, L_{MAX} are arbitrary chosen minimum and maximum n-gram ranks.

The function *contains()* has a small, but significant difference from the value similarity function. The first difference is that it is not commutative, because the denominator of eq. 4.2 uses the max function. More precisely, if we call Value Containment (VC) the containment function using edge values then VC is:

$$\mathrm{VC}(G^i, G^j) = \frac{\sum_{e \in G^i} \frac{\min(w_e^i, w_e^j)}{\max(w_e^i, w_e^j)}}{|G^i|} \tag{4.5}$$

This small change in the denominator is the cause for the asymmetric nature of the function and makes it correspond to a graded membership function between two graphs.

The similarity function calculation has a complexity of $O(|G_1| \times |G_2|)$, due to the fact that for each edge in G_1 one needs to lookup its identical edge in G_2. If an index is maintained with the edges' labels, this complexity can be diminished at the cost of memory use, which is the case in our implementation. Therefore, for every edge in the smallest of the two graphs, we perform a low complexity lookup in the edges of the biggest graph. If an edge is found we perform the calculation of the edge's contribution to the similarity sum. Otherwise, we continue with the next edge from the small graph. This gives a real complexity that is closer to $O(\min(|G_1|, |G_2|) \times \log(\max(|G_1|, |G_2|))$.

4.2 Graph Union (or Merging) and Intersection

The union, or merging, operator $\cup$ has two important aspects. The first deals with unweighted edges as pairs of labeled vertices $e = (v_1, v_2)$, while the second manages that in conjunction to the weights of the edges.

When performing the union of two graphs we create a graph $G_1 \cup G_2 = G^u = (E^u, V^u, L, W^u)$, such that $E^u = E_1^G \cup E_2^G$, where E_1^G, E_2^G are the edge sets of G_1, G_2 correspondingly. In out implementation we consider two edges to be equal $e_1 = e_2$ when they have the same label, *i.e.* $L(e_1) = L(e_2)$[2], which means that the weight is not taken into account when calculating E^u.

To calculate the weights in G^u there can be various functions depending on the effect the merge should have over weights of common edges. One can follow the fuzzy logic paradigm and keep the maximum of the weights of a given edge in two source graphs $w^u(e) = \max(w_1^G(e), w_2^G(e))$, where $w_1^G(e), w_2^G(e)$ are the weighting functions of the corresponding graphs. Another alternative would be to keep the average of the values so that the weight represents the expected value of the weights of the original weights. Given these basic alternatives, we chose the average value as the default union operator effect on edge weights. It should be noted that the merging operator is a specific case of the graph update function presented in section 4.4. Formally, if E^1, E^2 are the edge sets of G_1, G_2 correspondingly, W^i is the result graph edge weighting function and W_1, W_2 are the weighting functions of the operand graphs with $e \notin E^i \Rightarrow W_i(e) = 0, i \in 1, 2$, then the edgeset $E^\cup$ of the merged graph is:

$$E^\cup = E^1 \cup E^2, W^i(e) = \frac{W^1(e) + W^2(e)}{2}, e \in (E^1 \cup E^2) \tag{4.6}$$

The intersection operator $\cap$ returns the common edges between two graphs G_1, G_2 performing the same averaging operation upon the edges' weights. Formally the edgeset $E^\cap$ of the intersected graph is:

$$E^\cap = \{e | e \in G_1 \wedge e \in G_2\}, W^i(e) = \frac{W_1(e) + W_2(e)}{2}, , e \in (E^1 \cap E^2) \tag{4.7}$$

The merging operator defines the basic operation required to perform updates in the graph model, when for example one uses the n-gram graphs to model a class of documents. The intersection operator, on the other hand, can be used to determine the common subgraphs of different document graphs. This use has a variety of applications, such as common topic detection in a set of documents

[2]We consider the labeling function to be the same over all graphs.

(see part III) or the detection of 'stopword-effect' edges. The 'stopword-effect' edges are edges that are apparent in the graphs of most texts of a given language and have high frequency, much like stopwords.

As an example, the detection of stopword-effect edges can be accomplished by simply applying the intersection operator upon graphs of (adequately big) texts of various different subjects. The resulting graph will indicate the representation of that part of language that has appeared in all the text graphs and can be considered 'noise'. More on the notion of noise in n-gram graphs can be found in section 4.5.

The complexity of the merging and the intersection operators is similar to that of the similarity function calculation, because one needs to look for identical edges between G_1 and G_2 and then determine what to do with the weights. The main difference is that uncommon edges play a part in the merging operator. Thus, one can start from cloning the biggest graph, which has a complexity linear to the size of the biggest graph. Then, continuing with the edges of the small graph, one determines which of the edges already exist in the cloned graph to update their weights and which edges should be added to the cloned graph as new. The overall complexity for the merging operator, therefore, becomes for $|G_1| <= |G_2|$, $O(|G_1|) + O(|G_1| \times \log(\max(|G_1|, |G_2|))) + O(|G_1| \times i)$, where i is the cost of inserting a new edge into a graph, and is related to the implementation of the edges' index.

4.3 Delta (All-not-in) and Inverse Intersection

The Delta or *all-not-in* operator $\triangle$ is a non-commutative operator, that given two graphs G_1, G_2 returns the subset of edges from the first graph that do not exist in the second graph. Formally the edgeset $E^{\triangle}$ is:

$$E^{\triangle} = \{e | e \in G_1 \wedge e \notin G_2\} \tag{4.8}$$

The weights of edges is not changed when applying the delta operator. Obviously, the operator is non-commutative.

A similar operator is the inverse intersection operator $\bigtriangledown$ which creates a graph that only contains the *non-common* edges of two graphs. The difference between this operator and the delta operator is that in the inverse intersection the edges of the resulting graph may belong to any of the original graphs. Formally the resulting edgeset $E^{\bigtriangledown}$ is:

$$E^{\bigtriangledown} = \{e | e \in (G_1 \cup G_2) \wedge e \notin (G_1 \cap G_2)\} \tag{4.9}$$

Both the delta and the inverse intersection operators can be applied to determine the differences between graphs. This way one can *e.g.* remove a graph representing 'noisy' content from another graph (see section 4.5). Another application is determining the non-common part of two texts that deal with the same subject. The algorithmic complexity of the Delta operator is $O(|G_1| \times (\log(|G_2|) + i))$ as for every edge in G_1 one looks up whether the edge is contained in G_2 and, if it is not contained, the edge is inserted in the resulting graph with a cost of i. The complexity of the inverse intersection operator can be calculated based on the fact that the inverse intersection can be analysed in a merge, an intersection and a delta. This means that the, non-primary, intersection operator has a complexity of $O(\cup) + O(\cap) + O(\triangle)$.

4.4 Representing Document Sets - Updatability

The n-gram graph representation specification, as described in section 3.2, indicates how to represent a text using an n-gram graph. However, in Natural Language Processing it is often required to represent a whole document set. The most simplistic way to do this using the n-gram graph representation would be to concatenate the documents of the set into a single overall document. However, this kind of approach would not offer an updatable model, *i.e.* a model that could easily change when a new document enters the set.

In our applications we have used an *update function U* that is similar to the merging operator, with the exception that the weights of the resulting graph's edges are calculated in a different way. The update function $U(G_1, G_2, l)$ takes as input two graphs, one that is considered to be the pre-existing graph G_1 and one that is considered to be the new graph G_2. The function also has a parameter called the *learning factor* $l \in [0, 1]$, which determines the sensitivity of G_1 to the change G_2 brings.

Focusing on the weighting function of the graph resulting from the application of $U(G_1, G_2, l)$, the higher the value of learning factor, the higher the impact of the new graph to the existing graph. More precisely, a value of $l = 0$ indicates that G_1 will completely ignore the new graph. A value of $l = 1$ indicates that the weights of the edges of G_1 will be assigned the values of the new graph's edges' weights. A value of 0.5 gives us the simple merging operator. The definition of the weighting performed in the graph resulting from U is:

$$W^i(e) = W^1(e) + (W^2(e) - W^1(e)) \times l \tag{4.10}$$

The U function allows using the graph in a machine learning process for such tasks as text classification. The model training step for the creation of a class comprises the initialization of a graph with the first document of the class and the updating of that initial graph with the graphs of following class documents. Especially, when one wants the class graph's edges to hold weights averaging the weights of all the individual graphs that have contributed to the class, then the i-th new graph that updates the class graph should use a learning factor of $l = 1.0 - \frac{i-1}{i}, i > 1$. When a graph for each class is created, one can determine the class of a test document by using the similarity of the test document to the class graphs: the class that has the most similar graph to the test document graph is the selected class of the document.

4.5 Defining and removing noise using n-gram graph operators

When representing texts using n-gram graphs *e.g.* for a classification task one faces the presence of noise within the graphs. The noise within the graphs for the given task is the set of common subgraphs over all classes of documents. In traditional text classification techniques stopword removal is usually used to remove what is considered to be the noisy part of the data. Up to this point we have seen that n-gram graphs do not need such preprocessing. However, based on the task, we can usually identify non-interesting parts of data that hinder the task. This 'noise' can be removed via the already proposed n-gram graph algorithms.

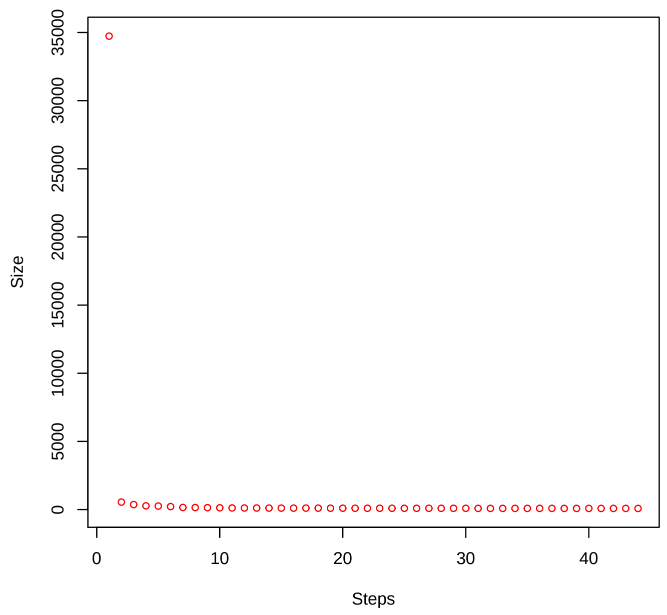


Figure 4.1: Convergence of common graph size over the number of conjunctions

In the case of a classification task, we create a merged graph for the full set of training documents T_c belonging to each class c. After creating the classes' graphs, one can determine the maximum common subgraph between classes and remove it to improve the distinction between different classes. A number of questions arise given this train of thought:

- *How can one determine the maximum common subgraph between classes?* According to our operators, it is easy to see that the maximum common subgraph is the conjunction of all the class graphs.
- *Is this (sub)graph unique?* No, it is not. Even though the conjunction operator is associative if edge weights are omitted, the averaging of edge weights per conjunction causes non-associativeness. In other words, $(G_1 \cap G_2) \cap G_3 \neq G_1 \cap (G_2 \cap G_3)$, in that edge weights are different.
- *Can we approximate the noisy subgraph, without iterating through all the classes?* Yes. As can be seen in figure 4.1 the noisy subgraph can be easily approximated in very few steps.
- *Does the removal of the noisy (sub)graph from each class graph really improve results?* The answer is yes, as will be shown in section 4.5.1.

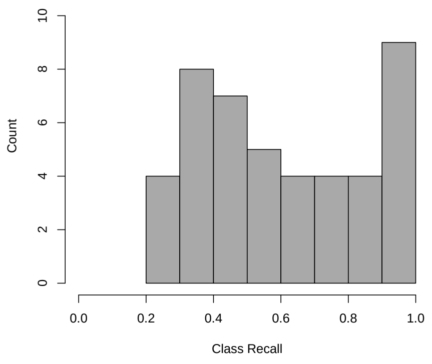


Figure 4.2: Class recall histogram for the classification task *including* noise

4.5.1 Judging the effect of noise within n-gram graphs

To support our intuition that there is a common part of the n-gram graphs that is actually noise, what we called stopword-effect edges, we performed a set of preliminary experiments on classification. We created a graph from all the texts of each of the TAC 2008 topics. Then, we tried to classify each of the training documents of all the topics to a corresponding topic. The classification was performed by measuring the similarity of the judged document to each topic graph. The topic of the graph with the maximum similarity was selected as the topic of the document. This way, we wanted to see if all documents would be found to be maximally similar to their original topic graphs. If that was not the case, then perhaps this would be the effect of common elements between the content of different topics, *i.e.* noise.

The class recall histogram of all topics-classes *before* removing the noise can be found in figure 4.2. The class recall histogram *after* removing the alleged noise can be seen in figure 4.3.

To see whether the results are indeed improved we used a paired Wilcoxon ranked sum test [Wil45], because of the non-normal distribution of differences between the recall of each class in the noisy and noise-free tests. The test indicated that within the 99% statistical confidence level (p-value < 0.001) the results when removing the noise were indeed improved. Further research should be conducted to determine the exact effect of noise for various tasks as well as different settings, *e.g.* various graph sizes.

Given the set of definitions, theoretical constructs and tools presented within this part of the thesis, we are now ready to proceed to the presentation of applications based on the n-gram graphs. Our first attempt was to determine a way to judge the quality of a given summary and we decided to do this *before* creating a summarization system. Paradoxical as it may seem, this order of research was chosen because it is important to know *what is desired*, before determining *how to achieve* it. Therefore, part II offers the AutoSummENG method of automatic summarization system evaluation, followed in part III by our summarization system.

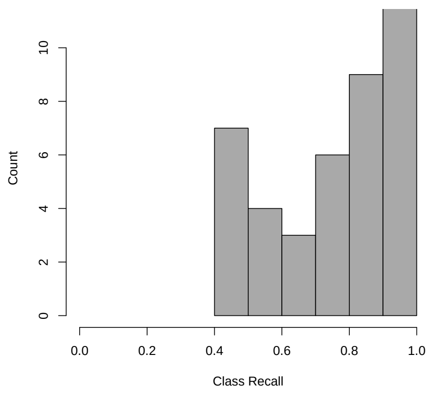


Figure 4.3: Class recall histogram for the classification task *without* noise

Part II

Summary Evaluation Using N-gram Graphs

Chapter 5

Summarization System Evaluation

We focus on the problem of evaluating summarization systems in an automated fashion. In recent scientific attempts to evaluate summarization systems, a multitude of problems arose, concerning mostly the inter-judge disagreement as well as the difficulty to automatically determine the quality of a summary. These problems are met mainly within the domain of multi-document summarization, where the synthesis of summaries appears to be more than mere extraction of text snippets [VHT03, Nen06].

The problem of inter-judge disagreement, as indicated in [VHT03, LH02], is the result of human subjectivity in terms of evaluation of summaries: it has been noted that human judges, appointed to grade the quality of a summary, disagree notably between each other on the grades assigned to different summaries. Several methodologies have been examined to systematize the grade assignment process, aiming at smoothing or nullifying the disagreement caused by methodology-specific practices [Nen06, RJB00, Mar00, SL02]. If one fails to create a methodology for humans to correlate vigorously to each other on the process of evaluation, then either the process of evaluation cannot be modeled objectively, which would be interesting to examine further by itself, or we need to define the process of summarization and its evaluation more precisely (for a more thorough discussion see [GKV06]).

On the other hand, the rankings posed by human grading over summarization *systems* correlate strongly. This indicates that people tend to prefer the same systems over other systems, which leads, as we will shortly present, to research concerning *automatic* evaluation methods that produce rankings similar to human rankings. In order to achieve the ranking, several attributes of summarization systems have to be examined and graded. These attributes are based on the qualities of the output summaries.

The problem of automatically determining the quality of a given summary, as we have already discussed in part I, appears to be approached using two different perspectives: either by *intrinsic* or *extrinsic* evaluation [MB97, VHT03]. Intrinsic evaluation operates on the characteristics of the summary itself, independent of the domain it may be used, trying for example to capture how many of the ideas expressed in the original sources appear in the output. On the other hand, extrinsic evaluation decides upon the quality of a summary depending on the effectiveness of using the latter for a specified task. Such a measure of extrinsic evaluation, namely *responsiveness*, appeared in the Document Understanding Conference (DUC) of 2005[1]. This extrinsic measure has been used in later DUCs as well.

In DUC 2005, the appointed task was the synthesis of a 250-word, well-organized answer to a complex question, where the data of the answer would originate from 25 documents [Dan05]. In DUC 2005, the question the summarizing 'peers', *i.e.* summarizer systems or humans, were supposed to answer consisted of a topic identifier, a title, a narrative question and a granularity indication, with values ranging from 'general' to 'specific'. We remind the reader that the responsiveness score was supposed to provide, as Dang states in [Dan05], a 'coarse ranking of the summaries for each topic, according to the amount of information in the summary that helps to satisfy the information need expressed in the topic statement, at the level of granularity requested in the user profile'. In other words, the responsiveness grade was meant to result in a partial ordering, indicative of how well a given summary answers a given question, taking into account the specifications of a question. It is important to note that responsiveness was *not* meant to be an absolute grading measure, but rather a partial ordering of the summarization abilities of the peers [Dan05].

The responsiveness grade was appointed by human judges and is therefore a useful measure, which an automatic evaluation system would aim at determining automatically. An automatic measure that could provide a similar ordering should strongly correlate to the responsiveness grade ordering assigned by humans.

Since there appears to be no absolute measure of quality for a summary, even for human judges, an automatic measurement would require at least one *model summary* (*i.e.* human extracted summary produced as a reference for measuring the goodness of the summaries produced by others), also called 'gold standard' or 'reference' summary. The human summaries offer high responsiveness content. This given, it would be possible to judge the *peer summaries* (*i.e.* summaries extracted by peer systems). Such measurements actually determine some kind of distance between the peer and the model summaries. The questions posed for such an automatic measure, having the same utility as the one responsiveness provides, would be:

- What is the kind of information that can be used in order to represent the peer and model summary?
- What should the actual representation method for the extracted information be, in order to retain information valuable in the comparison process?
- What kind of similarity measure can be used or defined, in order to provide meaningful results?

Automatic methods for the evaluation of summaries exist [HLZF05, Lin04, ZLMH06] and correlate highly to the measure of responsiveness. There are, however, some desired characteristics that do not coexist in a single method. More precisely:

[1] Also see *http://duc.nist.gov/*

- Language-neutrality. A method that does not require language dependent resources (thesauri, lexica, etc.) can be applied directly to different languages.
- Full automation. A method should not require human intervention, apart from the human model summaries.
- Context-sensitivity. A method should take into account contextual information, so that well-formedness of text is taken into account. Well-formedness can be loosely defined as the quality of a text that allows easy reading. A text that is a random sequence of words would lack this quality, even if the words are on topic.

Our method, named AutoSummENG (AUTOmatic SUMMary Evaluation based on N-gram Graphs), attempts to hold all these qualities, while bearing results with high correlation to the responsiveness measure, which indicates correlation to human judgement. The results of our experiments indicated that our method outperforms current state-of-the-art systems in that sort of correlation, while remaining strictly statistical, automated and context-sensitive due to the nature of the representation used, namely the n-gram graph (more on this in section 6.1).

Chapter 6

Summary of Method and Background Knowledge

The AutoSummENG method, is based on the concept of using statistically extracted textual information from summaries, integrated into a rich representational equivalent of a text, to measure similarity between generated summaries and a set of model summaries. The novelty of the method concerns the following points:

- The type of statistical information extracted.
- The representation chosen for the extracted information.
- The method of similarity calculation.

We remind the reader that the information extracted from source texts is a set of indicators of neighbourhood between n-grams contained within the source text. In other words, the method proposes the extraction of *relations* between n-grams, given spatial proximity of those n-grams within a given text. Then, a graph is constructed to indicate the full set of relations deduced (as edges between n-gram-labeled vertices), together with additional information concerning these relations. Such representations are extracted from both generated and model (*i.e.* human composed) summaries. An edge in a graph may contain such information as the mean distance between the neighbouring n-grams in all occurrences, or a distance-weighted co-occurrence count for the related pair of n-grams, or a detailed distribution of distances between the pair of n-grams in texts (also see section 3.2).

Finally, a comparison between the graph representation of generated and model summaries is made, returning a degree of similarity between the graphs. At this point, generated summaries that are found to be on average more similar to model summaries are considered better. Systems that generate, on average, better summaries are in turn considered better systems. The comparison methodology varies from vertex-only comparison between graphs, to full comparison including the information attached to the edges.

Given the above, we have evaluated different representation types, based on both the type of represented data (character or word n-grams) as well as the use or not of connectivity information between the data (graph or histogram).

During its evaluation the system was found to perform differently based on its parameters.

In chapter 7 a full study was also conducted, focusing on how these parameters can be a priori optimized, to provide a fully automated evaluation methodology. The study was based on the fact that there are relations between meaningful n-grams, we call 'symbols' and non-meaningful ones, which we call 'non-symbols'. These categories of n-grams are based on statistical criteria and are used to describe how noise can deteriorate the performance of our method, as a function of the methodology parameters. Given this noisy-channel model of our approach, we were able to perform an a priori estimation of the method parameters.

At this point, we briefly review the background related to basic concepts of our methodology, such as n-grams and graphs, also presenting how comparison between graphs is performed.

6.1 Proposed Evaluation Method

Tackling the problem of what kind of information should be used to represent a peer and a model summary in the evaluation of a summary, one should take into account that the *surface appearance* of two equivalent pieces of information conveying the same content need not be identical, as happens in the case of paraphrases [ZLMH06]. Nevertheless, it is quite probable that the words expressing the content will exist in the same context, or that part of the words used will be identical, for example if different inflections are used. For more on the linguistic aspect of this assumption please consult [MS99]. Our method accounts for these assumptions, while retaining language-neutrality, by using only statistical methods and language-independent assumptions for the extraction of information from texts and for the computation of textual similarity.

6.1.1 Representation

Our method uses character n-grams, positioned within a context-indicative graph. Once more, in our analysis we consider that we view neighbourhood with respect to the *current* n-gram, which is a subsequence of the text analysed. In the following analysis, we have also used word n-grams to be able to evaluate the method, as the n-gram graph representation is applicable to both word or character n-grams.

In the research conducted, it was important to see if a histogram offers equally well results with a graph in the process of the evaluation. If that stood, it would mean that the graph representation should not be used altogether.

The n-gram histogram representation is a simple frequency histogram measuring n-grams' occurrences. In other words, it simply indicates the number of times an n-gram appears, without any neighbourhood information. This representation will be used as a baseline to indicate whether neighbourhood information is indeed important in our domain of application.

6.2 Comparison

In order to compare two summaries T_1 and T_2, we need to compare their representations. Given that the representation of a text T_i is a set of graphs $\mathbb{G}_i$, containing graphs of various ranks, we briefly review the similarity measures between graphs G^i, G^j of the same supposed rank n, which were presented in section 4.1:

- *Co-occurrence Similarity (CS)*, indicating how many of the edges contained in graph G^i are contained in graph G^j. We define $e \in E^G \equiv e \in G$. Thus co-occurrence is defined as:

$$\mathrm{CS}(G^i, G^j) = \frac{\sum_{e \in G^i} \mu(e, G^j)}{\max(|G^i|, |G^j|)} \tag{6.1}$$

  μ is the membership function, which returns 1 if e belongs to G, else it returns 0. Also $|G|$ is the number of edges $e \in G$. The definition causes CS to be symmetric, *i.e.*

$$CS(G^i, G^j) = CS(G^j, G^i) \tag{6.2}$$

  Also, CS takes values in $[0, 1]$, with a value of 1 indicating a full match of the two graphs, even though edge weights are not taken into account. On the other hand, a value of 0 means that no edge from one graph exists in the other. In this measure, each matching edge contributes by $\frac{1}{\max(|G^i|,|G^j|)}$ to the sum. The CS is a normalized derivative of common graph distance measures, based on the Maximum Common Subgraph [Bun98].

- *Value Similarity (VS)*, indicating how many of the edges contained in graph G^i are contained in graph G^j, considering also the weights of the matching edges. In this measure each matching edge e having weight w_e^i in graph G^i contributes $\frac{\mathrm{VR}(e)}{\max(|G^i|,|G^j|)}$ to the sum, while not matching edges do not contribute (consider that if an edge $e \notin G^i$ we define $w_e^i = 0$). The *ValueRatio (VR)* scaling factor is defined as:

$$\mathrm{VR}(e) = \frac{\min(w_e^i, w_e^j)}{\max(w_e^i, w_e^j)} \tag{6.3}$$

  The equation indicates that the *ValueRatio* takes values in $[0, 1]$, and is symmetric. It is easy to see that this allows the VS metric to retain the symmetricity inherited from the CS equation part. Thus, the full equation for *VS* is:

$$\mathrm{VS}(G^i, G^j) = \frac{\sum_{e \in G^i} \frac{\min(w_e^i, w_e^j)}{\max(w_e^i, w_e^j)}}{\max(|G^i|, |G^j|)} \tag{6.4}$$

  VS is a measure converging to 1 for graphs that share both the edges and similar weights, which means that a value of VS = 1 indicates perfect match between the compared graphs.

A histogram is actually an illustration of a frequency distribution. In this work we use the term histogram to indicate the feature vector that represents a frequency distribution of n-grams.

The analogous measure of CS from graphs to histograms, which we name CS_H, is actually based upon a binary decision of whether an (n-gram) element h of histogram H_1, with a value of v_1, also exists in a histogram H_2, independent of the value v_2 of the same element in H_2. Each co-existing element h contributes to CS_H a quantity of $\frac{1}{\max(|H_1|,|H_2|)}$ and, therefore, the equation for CS_H is:

$$\mathrm{CS}_H(H_1, H_2) = \sum_{h \in H_1} \frac{\mu(h, H_2)}{\max(|H_1|, |H_2|)} \tag{6.5}$$

where $\mu(h, H)$ equals to 1 if $\exists h \in H : v(h, H) > 0$, otherwise it equals to 0, where v is the value corresponding to h in histogram H. Also, $|H|$ is the number of elements in histogram H.

On the same basis and by setting $v(h, H)$ for each $h \notin H$ to 0, then VS_H is defined as:

$$\mathrm{VS}_H(H_1, H_2) = \sum_{h \in H_1} \frac{v(h, H_2)}{\max(|H_1|, |H_2|)} \tag{6.6}$$

The fact that we have proposed extraction of different ranks of n-grams for the composition of the graphs allows us to take advantage of matching between different ranks. The overall result of both the CS and the VS calculations, here depicted by the superscript 'O', is a weighted average of the ranks. For example, for CS we have:

$$\mathrm{CS}^O = \frac{\sum_{r \in [L_{\min}, L_{\mathrm{MAX}}]} r \times \mathrm{CS}^r}{\sum_{r \in [L_{\min}, L_{\mathrm{MAX}}]} r} \tag{6.7}$$

where CS^r is the CS measure for extracted graphs of rank r, and $L_{\min}$, L_{MAX} are arbitrary chosen minimum and maximum ranks.

The intuition behind equation 6.7 is that matching between higher order ranks is more important between matching in lower level ranks. This intuition relies on the fact that languages rely on the composition of simpler character strings to create more complex ones, which bear richer meaning than their components. This composition occurs in characters, words, sentences, paragraphs and so forth and has been founded both by generative as well as statistical language processing research (e.g. [MS99, Introduction]). Similarly for VS^O, CS^O_H, VS^O_H. In the experiments, these overall values were used as results for the comparison process. It should be noted that the function of weight given the rank has been found empirically, and thus better alternatives can be found[1].

6.3 Experimental Setup – Representation

Based on the definition of the proposed similarity measures we wish to show, by experiments, that systems with summaries getting higher CS or VS than others, are indeed better systems; and so it has proved to be, by correlation to the responsiveness measure (see Table 7.1). We will refer to this correlation to the responsiveness measure as *evaluation performance (EP)*, as opposed to *system or method performance* which refers to the grade appointed to an evaluated system or method.

[1]We have also tried the simple reverse of rank r ($\frac{1}{r}$), with worse results.

The data on which the measurements were conducted was the summary and evaluation corpus of DUC 2006. The corpus consists of summaries for 50 different topics. Each topic had a number of automatically extracted summaries, one for each participating system, and 4 human created summaries. The human summaries were differentiated by means of an identifier, as were the baseline system summaries, which originated from a baseline system created by NIST, which simply took the first 250 words of the most recent document for each topic. It is important to indicate that human summaries were used both as model summaries *and* as peer summaries. All summaries were truncated to 250 words before being evaluated. To verify some of our experiments using a second corpus, we have used the corpora of DUC 2005 and DUC 2007 as well. The corpus of DUC 2005, similarly to the one of DUC 2006, consists of summaries for 50 different topics. The only major difference is that 30 of the topics had 4 human summaries each, while the remaining 20 topics each had either 9 or 10 human summaries. The corpus of DUC 2007 consists of summaries for 45 different topics. Each topic has 4 humanly created summaries, as well as 28 machine generated summaries.

The measure used for evaluation performance was the correlation of the method evaluation metric to the responsiveness measure, which has already been used and accepted by recent research [Dan05, HLZ05, Nen06]. The statistical measures used were Pearson's product moment correlation coefficient, as well as Spearman's rank correlation. We remind the reader that Pearson correlation takes values in $[-1, 1]$, where a value of 1 (or -1) indicates perfect (or negative perfect) correlation between two measures, while a value of 0 indicates no *apparent* correlation. Spearman correlation indicates whether two measures used as grades will provide similar rankings given a set of contesting entities, with values near 1 indicative of a higher correlation. On that basis, we would require our method to approach the maximum correlation value of 1 to the responsiveness measure. In the following tests, we consider representations and measures with Spearman's coefficient values closer to 1 better. It should be noted that the experiments used both *character* and *word* n-grams as granularity levels, to see if there is a difference between the two.

The set of experiments concerning representation, attempted to evaluate simple n-gram histogram representation, in comparison to the graph representation. The measures of Co-occurrence Similarity CS and Value Similarity VS have been used for both representation schemes and the results of the experiments were correlated to the responsiveness measure.

Each peer summary was compared both in terms of *character* n-gram and *word* n-gram graphs, as well as the corresponding character and word n-gram histograms, to all model summaries separately. Then the similarities were averaged to conclude a final similarity of the peer summary to the models. Human summaries, that appeared both as peer and model summaries, were not compared to themselves in the process of comparison.

Requiring that representations are compared between themselves, regardless of the parameters, we conducted a series of experiments within a very wide range of parameter settings. The three parameters used are:

1. Minimum n-gram length, indicated as $L_{\min}$.
2. Maximum n-gram length, indicated as L_{MAX}.

Representation - Measure	*Max*	*Min*	*Mean*	*Std. Deviation*
Graph – Value	0.938	0.803	0.914	0.019
Histogram – Co-occurrence	0.934	0.723	0.912	0.035
Graph – Co-occurrence	0.912	0.738	0.883	0.020
Histogram – Value	0.854	0.502	0.633	0.097

Table 6.1: Histogram and Graph Character N-gram Statistics Ranked by Mean Performance

3. Neighbourhood Window Size, indicated as D_{win}.

The values given to the above parameters were as follows:

- $L_{\text{min}} \in [1, 10]$, which means we have taken into account n-grams from unigrams to ten-grams.
- $L_{\text{MAX}} \in [L_{\text{min}}, 10]$, which means we have taken into account n-grams from the size of the selected L_{min} and up to ten-grams.
- $D_{\text{win}} \in [1, 20]$, which means we have taken into account a window size of one and up to twenty in different iterations of the experiment.

The features that differentiate representations are:

- the word or character nature of the n-grams in the representation.
- the type of the representation: histogram or graph.
- the existence of binary or real values in the representation (co-occurrence vs. value). For example, a histogram indicative of occurrence would only use binary values, while a histogram containing frequency or probability of occurrence would use real values. In the case of a graph, a binary value may indicate co-occurrence, while a real value may also indicate strength of this co-occurrence.

6.4 Results on Characters: Histogram or Graph – Co-occurrence or Value

Table 6.1 depicts the evaluation performance of four different approaches concerning the representation (either graph or histogram) and measure (either co-occurrence or value). For each approach, we have measured the maximum evaluation performance, the minimum evaluation performance, the mean and the standard deviation of evaluation performance. Concerning character n-grams, in Table 6.1 we can see that the most promising representation is that of the graph value, based on the ranking of average performance and robustness (*i.e.* least standard deviation).

Statistical Evidence

In order to statistically support whether different approaches indeed rendered different results and, thus, conclude on which approach is better, we tested whether the distribution of evaluation performance values for each method was normal (Gaussian). If this stood, we would be able to apply a simple one-sided t-test to reach a decision about comparison between approaches. The Anderson-Darling normality test [Ste74] was used to judge normality. This test has the null hypothesis that the samples examined come from a normal distribution. Unfortunately, the distributions of the samples used are not normal (the samples had a probability of less than 2.2×10^{-16} to originate from a normal distribution) and we had to use another test.

At first, we tested whether the distributions of results from all methods were all different from one another, which would indicate that there is not a reason to choose a specific distribution. To do this, the Kolmogorov-Smirnov goodness-of-fit test [MJ51] was used, which has the null hypothesis that two sets of samples come from the same distribution. The test indicated that all distributions are indeed different with a confidence of more than 99%[2].

The test selected as appropriate for the task of deciding which evaluation approach performs better was the *Wilcoxon rank sum test (also called Mann-Whitney test)* [HW73, p. 27-33,68-75]. This test ranks the samples (in our case the evaluation performance values) over a common axis (in our case the $\mathbb{R}$ axis). Then, it uses the *sum of the ranks* of each sample set to see which corresponding distribution *is more probable to give samples that stand higher* than the samples of the other in the given axis. According to this test, the ranking of methods indicated that Histogram-Co-occurrence is probably a little *better* than Graph-Value methods, contrary to what the mean and standard deviation have indicated. In the bottom half of the ranking, Graph-Co-occurrence is worse than Graph-Value but better than Histogram-Value, as expected.

There was a hint that the oddity concerning Histogram-Co-occurrence and Graph-Value may be due to the effect of the D_{win} parameter in Graph-Value results. In other words, there are a lot of samples from the Graph-Value approach that have extreme D_{win} values, which affect the whole distribution of results. To check this possibility we performed the same test between Histogram-Co-occurrence and Graph-Value evaluation performances, where we had kept only performances of the Graph-Value approach for $D_{\text{win}} <= 10$, which is still a high value considering the average size of words in English. This indeed turned things around, and Histogram-Co-occurrence was presented as being *worse* than Graph-Value, even though the confidence level of 95% could not be reached (p-value of 0.1123). We double checked our intuition in the DUC 2005 corpus, once more for values of D_{win} under 10, inclusive, and got a result indicating that Graph-Value is *better* than Histogram-Co-occurrence with a confidence level of 99%. The above indications gave us the incentive to delve further in the optimization of parameters, as can be seen in chapter 7. However, all the analysis pointed that, given non-extreme values of all three parameters (D_{win}, L_{min}, L_{MAX}), the Graph-Value approach can outperform the Histogram-Co-occurrence one.

One should note that the D_{win} parameter does not affect the histogram representation (as there is no D_{win} parameter when using the histogram). This means that the test concerning the histogram only included L_{min}, and L_{MAX} parameters. The experiments concerning the graph representation, on the other

[2]The p-value of the null hypothesis was less than 2.2×10^{-16} for all cases except for the G-V – G-C case where p-value was 0.001015; well over the 99% confidence level.

	$G - C$	$G - V$	$H - C$	$H - V$
G – C	-	No	No	Yes
G – V	Yes	-	No	Yes
H – C	Yes	Yes	-	Yes
H – V	No	No	No	-

Table 6.2: Statistics using Spearman Correlation for Character N-Grams on DUC 2006. Each square indicates whether the representation of the corresponding *row* gives better results than the one in the corresponding *column*, based on a Wilcoxon rank sum test with *95%* confidence level. G stands for Graph, H for Histogram, C for co-occurrence and V for value.

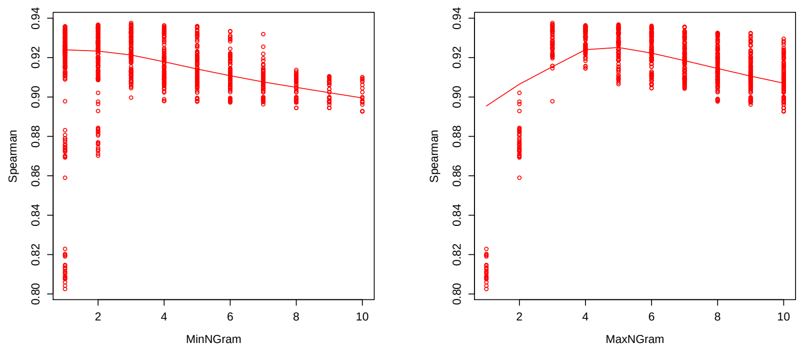


Figure 6.1: Scatterplot of L_{min} (MinNGram) on the left and L_{MAX} (MaxNGram) on the right and Performance (Spearman) - Char N-Grams

hand included D_{win}. We tested the effect of all parameters to the system performance.

To determine the effect of different parameters on the performance of the Graph – Histogram evaluation, we used a scatterplot graph. The scatterplot graph indicates how the values of our evaluation performance (vertical axis), as a value of correlation to human grading, varied between different runs, given different values of a parameter. Grand variation in performance for a single parameter value is indicated by highly dispersed points in the vertical axis, while robustness is indicated by many, closely positioned points in the vertical axis. The smooth line in the graphs was extracted via LOWESS regression [Cle81] and helps identify the trend of the performance given the parameter.

In Figure 6.1 we can see that marginal values of L_{min}, which is the parameter under study, worsen the performance of the system. In low values the deviation is raised, while in high values the average performance is lowered.

In Figure 6.1 we can see that, like L_{min}, marginal values of L_{MAX} worsen the performance of the system. In this case, there is no obvious effect in the robustness of the performance.

Finally, in Figure 6.2 we can see that, while there is no obvious effect in the robustness of the performance by increasing the value of the D_{win} parameter (depicted as Dist in the figure itself), the performance appears to deteriorate gradually.

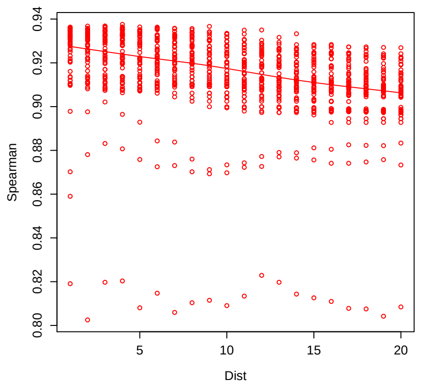


Figure 6.2: Scatterplot of D_{win} (Dist) and Performance (Spearman) - Char N-Grams

Parameter	*Correlation*
D_{win}	-0.316
L_{min}	-0.148
L_{MAX}	0.080

Table 6.3: Pearson Correlation between D_{win} and Performance for Character N-Grams Graph – Value

In order to validate this impression we examined the Pearson correlation between the three parameters and the performance of the system. The results, shown in Table 6.3 with the first column indicating the parameter and the second the correlation of its value to the evaluation performance, indicate that there is indeed negative correlation between D_{win} and performance, even though it is not very strong. The assumption derived from the negative impact of the distance to the performance was that we should be very careful with the selection of D_{win}, as it seems to insert some type of noise in the evaluation process.

6.5 Results on Words: Histogram or Graph – Co-occurrence or Value

Applying the same pattern of analysis for the word n-grams we reach the following conclusions:

- In word n-grams the histogram is by far better than the graph (see Table 6.4).
- The two best approaches, both of which concern histograms, do not have statistically supported difference in their performance. Therefore the simplest of the two should be chosen (*i.e.* Histogram-Co-occurrence).
- There is very serious variance in results for different parameter values. The standard deviation is actually an order of magnitude higher that that of

Representation - Measure	*Max*	*Min*	*Mean*	*Std. Deviation*
Histogram – Value	0.898	0.450	0.767	0.108
Histogram – Co-occurrence	0.909	0.306	0.741	0.173
Graph – Value	0.909	0.072	0.478	0.227
Graph – Co-occurrence	0.882	0.046	0.457	0.223

Table 6.4: Histogram and Graph Word N-gram Statistics ranked by Mean Performance

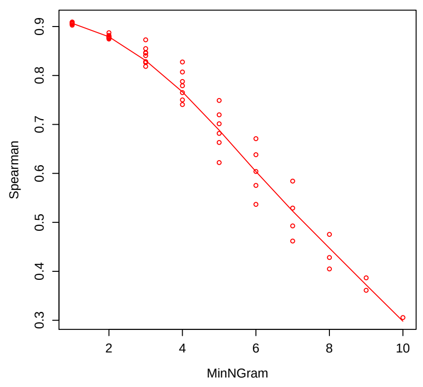


Figure 6.3: Scatterplot of L_{min} and Performance (Spearman) - Word N-Grams

the character n-grams. This has to do with the impact of parameter values on performance. The word-based system seems to be much more sensitive than its character counterpart to L_{min} (see Figure 6.3 and Table 6.5). On the other hand, L_{MAX} seems not to affect the histogram performance (see Table 6.5). The above indicate that for word n-grams one should be very careful in choosing minimal values for the graph approach.

To recapitulate, our experiments indicate that, in the task of summary system evaluation:

- the best representation for *character n-grams* is the Graph – Value representation, even though the Histogram – Co-occurrence representation is almost equally effective, when the distance parameter for the Graph – Value representation is extreme. A low value in the distance parameter of the graph is more likely to produce good results. The L_{MAX} parameter should be chosen to have a non-extreme value, even though further exper-

Parameter	*Correlation*
L_{min}	-0.255
L_{MAX}	-0.038

Table 6.5: Pearson Correlation between D_{win} and Performance for Word N-Grams Histogram – Value

iments, presented in chapter 7, were conducted to show which value can be considered as non-extreme.

- the best representation *word n-grams* is that of Histogram – Value, even though Histogram – Co-occurrence is not much less effective. In word n-grams, the minimum n-gram rank parameter $L_{\min}$ of the histogram plays a serious role, indicating that low-rank n-grams are important and should be used, while the upper limit to the choice of n-gram rank is not directly linked to the overall performance and should therefore be kept low to reduce number of calculations.

- considering the fact that the use of *character n-grams* performs overall much higher than its *word* counterpart (look again at tables 6.1 and 6.4), we should examine the use of character n-grams further. As we discuss in the concluding chapter, chapter 9.3, the *word n-gram* methodology should be examined under another point of view to see whether it correlates to other evaluation measures, something that lies outside the scope of this work.

Chapter 7

Optimizing N-gram Graph Parameters

Despite the robustness of the proposed method, we attempted to delve further in the strengths and weaknesses of the n-gram graph representation and the parameters inherent in the described method. We remind the reader that the three parameters used are:

1. Minimum n-gram length, indicated as L_{min}.
2. Maximum n-gram length, indicated as L_{MAX}.
3. Neighbourhood Window Size, indicated as D_{win}.

The absolute limits of the above three parameters are actually text-driven, since all parameters cannot exceed the size of the text. There is an additional obvious restriction, demanding that the maximum n-gram length should not be lower than the minimum n-gram length. However, since the complexity of the data structure and the number of calculations is exponential to the n-gram lengths, as well the window size, we created a model that can predict near-optimal values for the parameters.

In order to verify the correctness of the model, as well as the deviation of its response from the actual optimal, we conducted a series of experiments in the corpus of DUC 2006 using two approaches:

1. The *exhaustive* approach, where a big number of combinations of the triplet L_{min}, L_{MAX}, D_{win} were evaluated within adequately big limits for each parameter to extract an overall optimum.
2. The *model-based* approach, where the model-predicted values of the parameters were evaluated to indicate whether the response was approximate to the optimum.

During the exhaustive approach the parameters were given values as follows:

- $L_{\text{min}} \in [1, 10]$, which means we have taken into account n-grams from unigrams to ten-grams.
- $L_{\text{MAX}} \in [L_{\text{min}}, 10]$, which means we have taken into account n-grams from the size of the selected L_{min} and up to ten-grams.

‘permanent’, ‘permit’, ‘permits’, ‘persist’, ‘person’, ‘personal’, ‘personal computers’, ‘personnel,’ ‘persons’, ‘persuade’, ‘pesticide’, ‘pesticides.’,
‘permi’, ‘permitt’, ‘pers’, ‘pers and’, ‘person kn’, ‘person or’, ‘perti’, ‘perty’, ‘pes’, ‘pes o’

Figure 7.1: Sample extracted symbols

- $D_{\text{win}} \in [1, 20]$, which means we have taken into account a window size of one and up to twenty in different iterations of the experiment.

The limits of the given values were set arbitrarily, however it was obvious during the experiments that performance of the system near the limits was very low, deteriorating with every step with higher parameter values. Thus, it was obvious that our limits were rational, given the language[1] and the set of texts. It should be reminded that we used the symmetric approach for the extraction of n-gram neighbours (see section 3.2) in all our experiments, because it exhibited the most promising results and it is probably the most language neutral, considering orientation in the writing of texts (left-to-right or right-to-left).

At first we attempted, in order to reduce the number of experiments, to hold the D_{win} parameter constant in the arbitrary, but well-performing, value of 3 and change only the values of $L_{\min}$ and L_{MAX}. This way we planned to find a local-maximum and to investigate a correlation between n-gram size limits and system performance. In the course of our experiments we discovered that the optimal value of D_{win} is correlated to $L_{\min}$, L_{MAX} and cannot be held constant. At that point we formed a model that would contain all our findings.

7.1 Symbols, Non-symbols

We considered our text T to contain *symbols* and *non-symbols*. Let us elaborate in these two types of character sequence:

Symbols They are supposed to carry the meaning of the text, and they should be sequences of characters (letters) that are not neighbours by mere chance. The letters of an existent word should be found neighbouring more often than random characters that do not form a word.

Non-symbols They are sequences of characters (letters) that simply happen to occur near each other and have no actual meaning by themselves. Non-symbols are all the letter sequences from the text that are not symbols.

In Figure 7.1 we have indicated some sample extracted symbols and in Figure 7.2 some non-symbols. We see that symbols may include simple terms and collocations (more on collocations in [MS99, section 1.4.4]). We can also see other sequences of letters, like words lacking their endings (e.g. ‘permitt’), ending themselves (e.g. ‘perty’, ‘pes’) or other sequences of no apparent semantics (e.g. ‘pes o’).

[1]A language where the average word length would be twenty characters may require different limits.

‘permit </HEADLINE>’, ‘permit program’, ‘permit approved’

Figure 7.2: Sample non-symbols

Ideally, we would like our representation to only include symbols. However, based on the given method, the n-gram graph includes information for both symbols and non-symbols, which induces noise in the comparison process. Therefore, we need to identify such values of parameters $L_{\text{min},0}$, $L_{\text{MAX},0}$, $D_{\text{win},0}$ that minimize the noise, while maximizing the quantity of useful information.

The following problems arise:

- How can we formally define and detect symbols and non-symbols?
- Is the usefulness of all symbols equal and how do we measure it?
- Can we define a single, easily understood quantity that we need to maximize in order to achieve the required result?

7.2 Detection of Symbols

As described above, a symbol is a sequence of characters in the case of character n-grams. This sequence abides by a single rule: each letter is more probable to tail its preceding subsequence of characters than a character drawn randomly from a pool of characters. We elaborate on how a symbol is extracted from a text.

We have a text T^l. We denote by s_t the symbol we have composed at step t of the extraction process, and c_t the candidate symbol for the t-th step. A candidate symbol will become a symbol, if and only if it conforms to the rule described in the previous paragraph. In order to determine what is the probability of a given substring X to be followed by a given character y, we construct a corpus containing a set of texts from the domain. In our case, we simply used all the model summaries as the corpus and we created an overall text, T_0^L of length L, formed by the concatenation of the corpus texts. Thus, given a pair (X,y) with X having a length of $|X|$, we can count:

- how many times X appears in T_0, represented by N_X.
- how many times the string Xy appears in T_0, represented by N_{Xy}.
- the total number of n-grams of a given size n within T_0, represented by $|T_{0,n}|$.

We need to calculate the probability $P(y|X)$ of a given suffix y, given the prefix X:

$$P(y|X) = P(X) * P(y,X), \text{where } P(y,X) = \frac{N_{Xy}}{|T_{0,n}|} \text{ and } P(X) = \frac{N_X}{|T_{0,|X|}|} \quad (7.1)$$

On the other hand, the probability $P(y_r|X)$ of a *random* suffix y_r, given the prefix X, is given by:

$$P(y_r|X) = P(y_r), \quad \text{since } y_r \text{ is chosen randomly and } \textit{independently from } X.$$

Thus,

$$P(y_r) = \frac{1}{|A|_1} \tag{7.2}$$

$|A|_n$ is the number of strings of length n, found in the alphabet A of T_0. When we use the term *alphabet of* T_0, we refer to the set of unique characters appearing in T_0. We select this definition of an alphabet because we want our approach to remain language-neutral, and therefore we do not presume any given alphabet.

The extraction of symbols from T_0^L is described as algorithm 2.

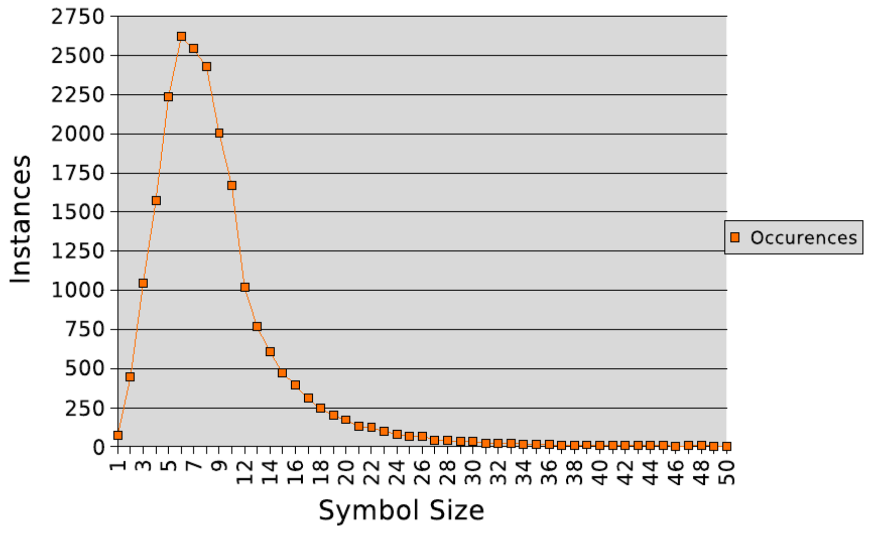
```
   Input: text $T_0^L$
   Output: symbol set $S$
   // t denotes the current iteration, but has no use in the
      algorithm
   // $T[i]$ denotes the $i$-th character of $T$
   // $\epsilon$ is the empty string
   // $P(y_r)$ is the probability of a random suffix $y_r$
   // The plus sign (+) indicates concatenation where character
      series are concerned.
 1 $S = \emptyset$;
 2 $s_t = T_0^L[1]$;
 3 for all i in [2,length(T)] do
 4     $y = T_0^L[i]$;
 5     $c_t = s_t + y$;
 6     if $P(y|s_t) > P(y_r)$ then
 7         $s_t = c_t$;
 8     end
 9     else
10         $S = S + s_t$;
11         $s_t = y$;
12     end
13 end
   // Add last symbol
14 $S = S + s_t$;
```

Algorithm 2: Extracting Symbols

Descriptively, the above algorithm runs through the text, splitting symbols when the next character seems to have been positioned after the current substring by mere chance. Starting with a single-character candidate symbol, the algorithm adds new characters to the candidate symbol, until a split point is reached. Then, the candidate symbol is upgraded to a symbol, and a new candidate symbol is formed using the next character. In Figure 7.3 we can see the distribution of symbol sizes as extracted by the algorithm from the DUC 2006 corpus.

The aware reader may note that this method is related to the Symmetric Conditional Probability used in the LocalMaxs algorithm [DSDGL99], as well as the notion of 'glue' in [HS06b]. The main difference is that we do not evaluate candidate n-grams to keep the most prominent ones, but we consider all n-grams that represent symbols to be important, and all others not important. Additionally, the probabilities used in the extraction of symbols here are different from

Figure 7.3: The Distribution of Symbols per Rank (Symbol Size) in the DUC 2006 corpus



the ones already in previous publications (*e.g.* SCP) and there are no fixed rules based on the n-gram rank, because these would be language-dependent. However, it would be interesting to see whether the use of other existing methods for variable rank n-gram extraction can prove more fruitful than the proposed one. This has not been done in the context of the current work.

7.2.1 Signal to Noise – Symbol Importance

In order to determine a measure of the importance of each symbol in our method, we insisted on the probabilistic approach. We consider any given symbol to be more important, if it is less probable to be generated by a random symbol creator. This symbol creator, in order to create a new n-gram of size n, would choose randomly a $n-1$ rank n-gram from a pool of valid symbols and would randomly select an 1-gram symbol to append, creating the new n-gram. The importance of a symbol is indicated by a weighting factor. On the other hand, we consider non-symbols to be equally (un)important, in that each non-symbol has an importance of 1.

The fact that we have interesting and uninteresting pieces of data that form our input is analogous to a noisy channel model, where a signal (interesting pieces) is transmitted over a medium (algorithm) that adds noise (uninteresting pieces). In this case we would like to change the medium parameters (L_{min}, L_{MAX}, D_{win}), in order to maximize the signal and minimize the noise. A signal-to-noise approach, trying to predict what is the tradeoff between different values of L_{min} and L_{MAX} concerning the signal-to-noise ratio, can be based on an equation like:

$$SN(L_{\text{min}}, L_{\text{MAX}}) = 10 \times \log_{10}(\frac{S(L_{\text{min}}, L_{\text{MAX}})}{N(L_{\text{min}}, L_{\text{MAX}})}) \tag{7.3}$$

where $S(L_{\text{min}}, L_{\text{MAX}}), N(L_{\text{min}}, L_{\text{MAX}})$ are functions returning a measure of signal and noise correspondingly, for a given range ($L_{\min}$, L_{MAX}). SN indicates the function of signal-to-noise. The signal is the useful information we have captured via symbols, while the noise is the redundant or useless information we have captured via non-symbols.

$N(L_{\min}, L_{\text{MAX}})$ is defined as the count of non-symbols appearing in a given corpus for the given range:

$$N(L_{\min}, L_{\text{MAX}}) = \sum_{i=L_{\min}}^{L_{\text{MAX}}} |\text{Non-Symbols}_i| \tag{7.4}$$

where $|\text{Non-Symbols}_i|$ is the number of non-symbols of rank i.

On the other hand, for the case of symbols we wish to take into account the importance of each symbol, and therefore calculate *normalized weighted symbols*. The latter are weighted according to their importance, which is a function of their rank. The normalization step occurs over the weighted symbols to provide a new set of symbols, same in number as the ones found in the texts, which are however rearranged over different ranks in a way that they also illustrate the *importance* of any given rank. The number of weighted symbols for each n-gram rank r is calculated in two steps, within the given range [$L_{\min}$, L_{MAX}]:

1. Calculate the weight w_r of symbols for the specified rank r and sum over all weighted symbols to find the total, *weighted symbol sum* W_r for rank r. The weight w_s is defined to be the inverse of the probability of producing a symbol of rank r given a symbol of rank $r-1$, as longer symbols are less probable to appear as a result of a *random sampling* of characters. This means that we consider more important sequences that are less likely to have been randomly produced. Thus:

$$P(s_r|s_{r-1}) = \begin{cases} \frac{1}{|\text{Symbols}_r|+|\text{Non-Symbols}_r|} & \text{if } r = 1. \\ \frac{1}{|\text{Symbols}_{r-1}|+|\text{Non-Symbols}_{r-1}|} \times \frac{1}{|\text{Symbols}_r|+|\text{Non-Symbols}_r|} & \text{else.} \end{cases}$$

$$\text{So } w_r = 1/P(s_r|s_{r-1}) \tag{7.5}$$

   where $|\text{Symbols}_r|$ is the number of symbols in rank r.

2. Normalize W_r so that the sum of W_r over $r \in [L_{\min}, L_{\text{MAX}}]$ is equal to the original number of symbols in the texts. The normalized, weighted symbols W_r^0 for rank r are calculated by:

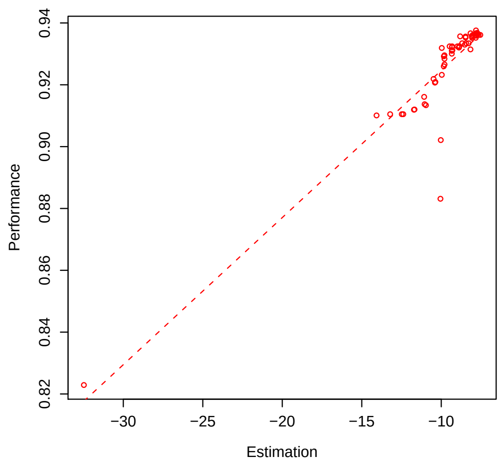
$$W_r^0 = W_r \times \frac{|\text{Symbols}_r|}{\sum_{i=L_{\min}}^{L_{\text{MAX}}} |\text{Symbols}_i|} \tag{7.6}$$

   We indicate once more that the W_r^0 measure actually represents the *importance of symbols* per rank r for the symbols of the texts, instead of the *number of symbols* per rank that is indicated by $|Symbols_r|$.

Thus, $S(L_{\min}, L_{\text{MAX}})$ finally equals to:

$$S(L_{\min}, L_{\text{MAX}}) = \sum_{i=L_{\min}}^{L_{\text{MAX}}} W_i^0 \tag{7.7}$$

Figure 7.4: Correlation between Estimation (SN) and Performance



Having defined our signal-to-noise function, we wish to maximize it and, therefore, we search the space of parameter values for optimum values. However, we have to investigate whether our estimate of signal-to-noise is correlated to the performance of the system, because only then will SN be useful. Indeed, SN offers an important *0.949* rank correlation (Spearman) to the maximum performance that can be achieved by our method (see Figure 7.4). The same correlation holds for the mean performance for a given L_{min}, L_{MAX} pair and different values of D_{win}. In fact, SN is also almost linear to the performance of the system, with a Pearson correlation of *0.918*. Therefore, our estimator is rather good in finding optimal values for L_{min} and L_{MAX}.

However, we have not yet discussed the distance parameter. We have said that it has a rather serious impact on the performance of the system. Up to this point the D_{win} parameter was presumed to be independent from L_{min} and L_{MAX}. However, further evolution and testing of our model indicated a possible connection between D_{win} and L_{MAX}.

We want to know, in the same manner as above, what is the signal-to-noise ratio as a function of distance D_{win}. We shall refer to this ratio as SN_d.

In order to determine SN_d, we can once more count the symbols and non-symbols expected to be found in a given distance D_{win} from our n-gram. Let us consider, without harming generality, the case where we are in the middle of our (infinitely long) text and we have a visibility of D_{win} characters to the right and left. Our current n-gram is of rank s_0. We are extracting n-grams of size r.

During our n-gram extraction, we extract $d_0 = 2 \times D_{\text{win}}$ n-grams (for the symmetric approach). Thus, there are d_0 candidate symbols. In order to calculate the probability of extracting N_s symbols from d_0 attempts, we can model the process of extraction with a binomial success probability, calculated for N_s successes in d_0 attempts. The chance of success for the binomial for a given n-gram rank of r is given by:

$$P_s = \frac{W_r^0}{W_r^0 + |\text{Non-Symbols}_r|} \tag{7.8}$$

The chance that our current n-gram is a symbol is the probability P_s^0 calculated by:

$$P_s^0 = \frac{W_{s_0}^0}{W_{s_0}^0 + |\text{Non-Symbols}_{s_0}|} \tag{7.9}$$

Presuming that only information about neighbouring *symbols* is important, the signal function should take into account only the probability of having both a *symbol* current n-gram and a *symbol* neighbour n-gram. Even though the maximum number of non-overlapping, neighbour symbols we can find within d_0 is $[\frac{d_0}{r}]$, we will not use this limitation in our analysis. We do so, because the analogy of symbols and non-symbols remains the same on average over all our corpus and our estimator can count on this analogy to extract good results *on average.* To extract an estimate E_s^r of the number of symbols that can be extracted for a given rank r, we use the algorithm indicated as algorithm 3.

```
  Input: Distance $D_{\text{win}}$, Success Probability of Single Trial $P_s$
  Output: Expected Number of Symbols $E_s^r$
  // $D(x)$ is a discrete probability distribution
1 for all i in [1,$D_{win}$] do
2 |   $D(x)$ = binomial(i;$D_{\text{win}}$,$P_s$) ;
3 end
  // E(y) is the mean function
4 $E_s^r$=E(D(x)) ;
```

Algorithm 3: Symbol Count Estimation

From algorithm 3, we get an estimated number of symbols. The rest of the d_0 extractions are non-symbols and account for $d_0 - E_s^r$ extractions. Therefore, SN_d can be calculated by:

$$SN_d(L_{\text{min}}, L_{\text{MAX}}) = 10 \times \log_{10} \frac{S_d(L_{\text{min}}, L_{\text{MAX}}, D_{\text{win}})}{N_d(L_{\text{min}}, L_{\text{MAX}}, D_{\text{win}})} \tag{7.10}$$

where S_d, N_d are the signal and noise functions correspondingly, calculated by:

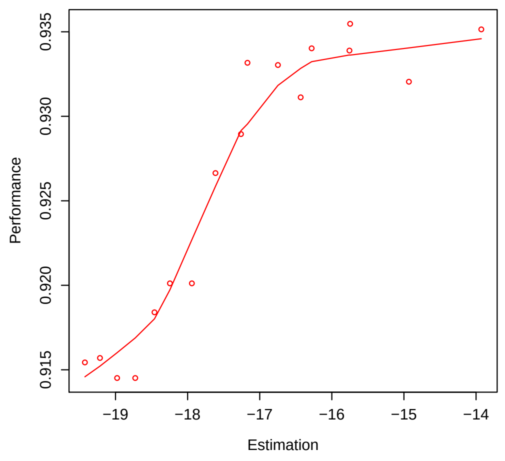
$$S_d(L_{\text{min}}, L_{\text{MAX}}, D_{\text{win}}) = P_s^0 \times \sum_{r=L_{\text{min}}}^{L_{\text{MAX}}} (E_s^r) \tag{7.11}$$

$$N_d(L_{\text{min}}, L_{\text{MAX}}, D_{\text{win}}) = \sum_{r=L_{\text{min}}}^{L_{\text{MAX}}} (d_0 - E_s^r) \tag{7.12}$$

Equations 7.11, 7.12 indicate that:

- the signal, given an n-gram rank range, is the sum of the probabilities over all ranks to create useful edges, *i.e.* edges connecting *symbols* , in the corresponding graphs.

Figure 7.5: Correlation between Estimation (SN) and Performance for Given L_{min}, L_{MAX}



- the noise is the sum, over all graph n-gram ranks, of the number of extracted graph edges that will involve at least one non-symbol.

As a result of this analysis, we conclude that the optimal distance D_{win} is *a function of L_{min}, L_{MAX}* and should be regarded as an independent parameter. In order to evaluate our model we will repeat the extraction of correlation between the SN_d and actual performance for different values of D_{win}.

Indeed, SN_d offers an important *0.920* rank correlation (Spearman) to the maximum performance that can be achieved for the selected optimal n-gram range (see Figure 7.5). SN_d has a promising Pearson correlation of *0.896* to the performance. Therefore, our estimator is good in finding near-optimal D_{win} values[2]. In the given example of the DUC 2006 corpus, the best performance was 0.938 and the one returned using the estimation was 0.935, while the average over all candidate distances was 0.927, with a standard deviation of 0.008.

The near-optimal values for the pair $(L_{\text{min}}, L_{\text{MAX}})$ were according to the estimation values $(4, 4)$, while their optimal values were $(1, 3)$ (DUC 2006 corpus).

7.3 Overall Performance of AutoSummENG

In order to check the performance of our method, compared to other existing methods, we used the Spearman correlation and the Pearson correlation that is used in [Dan06], but for our method we have also calculated the Kendall's tau correlation coefficient, which we consider to be the most fitting coefficient for the given task, based on its definition (see section 2.3).

The Responsiveness measure in DUC 2006 was named Content Responsiveness, because another measure appeared named Overall Responsiveness (see [Dan06]).

[2]This holds for already decided optimal values of L_{min}, L_{MAX} as we found through a series of experiments.

Metric	*Spearman*	*Pearson*	*Kendall*
Overall Responsiveness	0.718	0.833	
Rouge-2	0.767	0.836	
Rouge-SU4	0.790	0.850	
BE-HM	0.797	0.782	
AutoSummENG	*0.870* (0.00)	*0.904* (0.00)	*0.712* (0.00)
AutoSummENG B/S	*0.858* (0.00)	*0.899* (0.00)	*0.712* (0.00)

Table 7.1: Correlation of Measures to the Content Responsiveness Metric of DUC 2006 for *Automatic peers only*. Within parentheses the p-value of the corresponding test.

Evaluated Group	*Rouge-2*	*Rouge-SU4*	*BE-HM*	*AutoSummENG - B/S*
Automatic Peers	0.84 (0.00)	0.85 (0.00)	0.78 (0.00)	0.91 (0.00) - 0.90 (0.00)
Human Peers	0.64 (0.05)	0.69 (0.03)	0.57 (0.09)	0.68 (0.03) - 0.67 (0.00)
All Peers	0.90 (0.00)	0.88 (0.00)	0.88 (0.00)	0.97 (0.00) - 0.97 (0.00)

Table 7.2: Pearson Correlation of Measures to the Content Responsiveness Metric of DUC 2006 for *Automatic peers, Human peers and All peers*, excluding peer 17. Within parentheses the p-value of the corresponding test.

Briefly, Overall Responsiveness represents an overall quality measure (including grammaticality and other textual qualities) for a given system, while Content Responsiveness only refers to whether the required information were contained in summaries from a given system, without taking into account the well-formedness of output summary. The results concern application of the *character n-gram Graph – Value representation with a symmetric window*. P-values reported zero (0.00) indicate actual p-values that are rounded to zero when two digits are considered significant. We should note that we have also used an ordinary non-parametric bootstrapping approach[3] with 10000 replications to better determine the results for our method. The corresponding results appear either in the AutoSummENG entries as separate entries (see Table 7.1) or as second value – p-value pairs (see Table 7.2). [4]

In Table 7.1, there is an obvious difference between the performance of AutoSummENG and existing approaches. Table 7.2 indicates the Pearson correlation performance of evaluation methods when not including system 17 of DUC 2006, for which BE-HM breaks (due to some characters in the input) and performs abnormally.

As an overview of the major evaluation systems' performance over the data of DUC 2005 to 2007, the Table 7.6 has been provided, based partly on [CD08]. It should be noted that the AutoSummENG performance does not correspond necessarily to its *optimal* value, but rather to the performance achieved using pre-estimated parameters. Another important note is that, even though there is a difference between the performance of systems, statistical analysis indicates through confidence intervals that the difference in performance may be due to

[3]For an introduction to bootstrapping see [ET93].

[4]Given the fact that the results using bootstrapping were only marginally modified over many experiments we did not further perform bootstrapping, considering the given original values to be good and indicative estimations of the process, not wanting to diversify the method of calculation of our results from other corresponding works.

Evaluated Group	*Spearman*	*Pearson*	*Kendall*
Automatic peers	0.906 (0.00)	0.908 (0.00)	0.755 (0.00)
Human peers	0.857 (0.00)	0.830 (0.00)	0.764 (0.00)
All peers	0.957 (0.00)	0.985 (0.00)	0.847 (0.00)

Table 7.3: Correlation of AutoSummENG to the Responsiveness Metric of DUC 2005 for *Automatic peers, Human peers and All peers*. Within parentheses the p-value of the corresponding test.

Evaluated Group	*Spearman*	*Pearson*	*Kendall*
Automatic peers	0.870 (0.00)	0.904 (0.00)	0.712 (0.00)
Human peers	0.648 (0.04)	0.684 (0.03)	0.471 (*0.07*)
All peers	0.935 (0.00)	0.966 (0.00)	0.804 (0.00)

Table 7.4: Correlation of AutoSummENG to the Content Responsiveness Metric of DUC 2006 for *Automatic peers, Human peers and All peers*. Within parentheses the p-value of the corresponding test. Statistical importance lower than the 95% threshold are noted by *emphatic text* in the parentheses.

Evaluated Group	*Spearman*	*Pearson*	*Kendall*
Automatic peers	0.864 (0.00)	0.88 (0.00)	0.707 (0.00)
Human peers	0.615 (*0.06*)	0.649 (0.04)	0.396 (*0.12*)
All peers	0.935 (0.00)	0.964 (0.00)	0.801 (0.00)

Table 7.5: Correlation of AutoSummENG to the Content Responsiveness Metric of DUC 2007 for *Automatic peers, Human peers and All peers*. Within parentheses the p-value of the corresponding test. Statistical importance lower than the 95% threshold is noted by *emphatic text* in the parentheses.

Year	*BE(-HM)*	*Rouge-2*	*Rouge-SU4*	*AutoSummENG*
2005	0.87	0.94	0.93	0.91
2006	0.85	0.84	0.85	0.90
2007	0.89	0.88	0.83	0.88

Table 7.6: Pearson Correlation of Measures to the Content Responsiveness Metric of DUC 2005-2007 for Automatic Systems

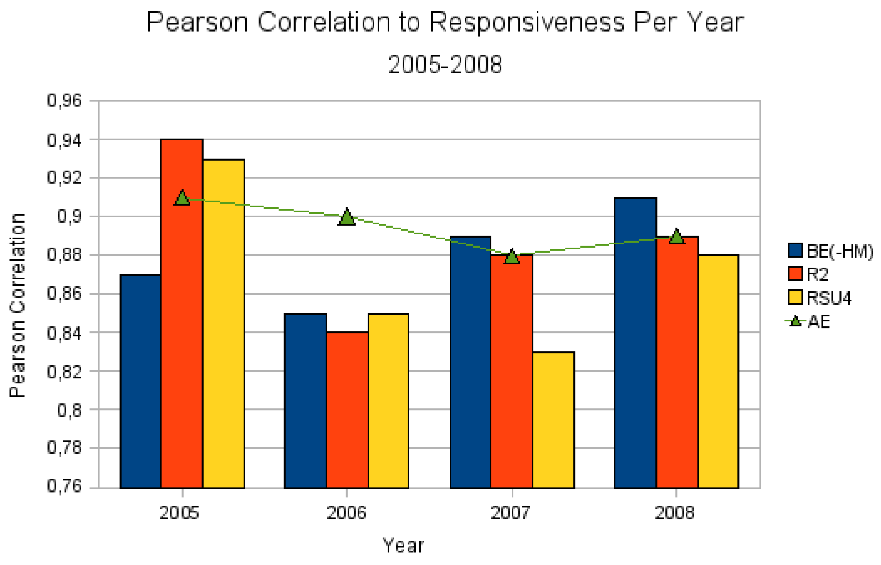


Figure 7.6: Pearson Correlation of Measures to the (Content) Responsiveness Metric of DUC 2005-2008 for Automatic Systems

randomness (see also [Dan05, Dan06, HLZ05]).

Even though the evaluation process itself contains the a-priori estimation step for its parameters, we wanted to check whether the model parameters determined for the corpus of DUC 2006 would function effectively when applied to DUC 2005 and DUC 2007 corpora. In tables 7.3, 7.4, 7.5 we can see the results for all corpora (DUC 2005, DUC 2006, DUC 2007). In the tables the results have been separated by groups (automatic peers and human peers) and there is also the overall ranking correlation, including all peers. The results indicate that the DUC 2006 parameters perform well in other corpora as well, showing that the parameters did not simply overfit the DUC 2006 corpus.

To verify this fact, we also determined model parameters for DUC 2005 and applied them to all corpora: DUC 2005, DUC 2006, DUC 2007. The results were once more satisfying as can be seen in Table 7.7. This hints on the fact that the model parameters are more language-dependent than corpus dependent, but this will have to be verified against another language. In Figure 7.6 one can see the correlation of various measures to (content) responsiveness over the years in DUC and TAC. AutoSummENG appears to be the most robust method, even though it is not consistently the best method.

The fact that our method does not require parsing of some kind, nor syntactic

Year - Evaluated Group	*Spearman*	*Pearson*	*Kendall*
2005 - Automatic peers	0.840 (0.0)	0.885 (0.0)	0.669 (0.0)
2005 - Human peers	0.936 (0.0)	0.878 (0.00)	0.854 (0.00)
2005 - All peers	0.929 (0.00)	0.977 (0.00)	0.803 (0.0)
2006 - Automatic peers	0.871 (0.0)	0.891 (0.0)	0.709 (0.0)
2006 - Human peers	0.759 (0.01)	0.715 (0.02)	0.566 (0.03)
2006 - All peers	0.937 (0.00)	0.967 (0.00)	0.806 (0.0)
2007 - Automatic peers	0.842 (0.0)	0.871 (0.0)	0.687 (0.0)
2007 - Human peers	0.659 (0.04)	0.673 (0.03)	0.442 (*0.08*)
2007 - All peers	0.925 (0.00)	0.966 (0.00)	0.792 (0.0)

Table 7.7: Correlation of AutoSummENG to the Responsiveness Metric of DUC 2005 and Content Responsiveness Metric of DUC 2006, 2007 for *Automatic peers, Human peers and All peers* using estimated parameters based on DUC 2005. Within parentheses the p-value of the corresponding test. Statistical importance lower than the 95% threshold are noted by *emphatic text* in the parentheses.

Representation	*Spearman*	*Pearson*
Graph – Co-occurrence	0.748	0.860
Graph – Value	0.786	0.893
Histogram – Co-occurrence	0.811	0.920
Histogram – Value	0.537	0.858

Table 7.8: Correlation of AutoSummENG to the Overall Responsiveness Metric

or grammatical analysis like other existing methods, offers an advantage, both in terms of complexity, as well as in terms of inherited noise from erroneous preprocessing (which was indicated as a problem in the case of BE [Dan06]).

In the course of our experiments, we used the optimal values found for content responsiveness correlation to check the correlation of the proposed method to the *Overall Responsiveness* of systems in DUC 2006. The results are illustrated in Table 7.8. Once more the method seems to do adequately well, with the histogram-co-occurrence version reaching the highest performance (for the given parameter setting of $L_{\text{min}} = 4, L_{\text{MAX}} = 4, D_{\text{win}} = 4$). This indicates that our method can have more applications than meets the eye and this seems worth investigating.

The fact that we need to estimate parameters, on the critic side of this analysis, can be time consuming and even error-prone, which will affect the overall performance of the system. This problem is only partially addressed by the robustness of the results for non-marginal parameter values. Another lesser drawback of our method is that the graph representation can be memory consuming, even though in our implementation[5] we have optimized the code and the problem has been tackled. Finally, there have been no experiments in different languages, which means that we have not answered the question of whether the language-neutral approach will have *similar performance* in other languages. On the other hand, this does not contradict the fact that the approach remains

[5]The full project, including source code, of AutoSummENG can be found at http://www.ontosum.org/static/AutomaticSummarization.

strictly language-neutral in its methodology.

Furthermore, there is a drawback of our method tied to the complexity of extracting and comparing n-gram graphs. This drawback has already been handled in terms of implementation, but the algorithm itself holds a complexity much higher that that of constructing a histogram, per se. Therefore, it would be interesting to hold only ‘useful’ subgraphs based on a statistical measure of usefulness or find an algorithmic alternative to our own.

Chapter 8

Notes on Summary Evaluation and Combinatorial Evaluation

8.1 Individual Summary Evaluation vs. Summary System Evaluation

Within the research conducted on automatic summary evaluation we found out that by optimizing evaluation methodologies based on their correlation to the responsiveness measure a number of problems have appeared:

- The automatic evaluation process performs well only when tackling the problem of overall evaluation. Different aspects and textual qualities of a summary have not been evaluated separately.
- The evaluation methodologies correlate well, only when viewed at the *system* level. In other words, we cannot judge the quality of *individual summaries* well.

To further support these conclusions we have performed two evaluations using the TAC 2008 dataset:

- the correlation (Spearman's rho, Kendall's tau and Pearson correlation) of the *system* evaluation scores to the human system judgements (average overall responsiveness and average grammaticality). The system evaluation scores are calculated by the average scores of the summaries provided by a single system.
- the correlation of the *summary* evaluation scores to human judgement (overall responsiveness and linguistic quality). The summary evaluation score is the AutoSummENG score of a single summary given a set of model summaries.

We note that the correlation between overall responsiveness and linguistic quality is 0.3788 (Kendall's tau, p-value < 0.01). This means that they are correlated, but not strongly. We also deduce from Table 8.1 that there are aspects

AutoSummENG to ...	*Spearman*	*Kendall*	*Pearson*
Overall Responsiveness	0.8953 (< 0.01)	0.7208 (< 0.01)	0.8945 (< 0.01)
Linguistic quality	0.5390 (< 0.01)	0.3819 (< 0.01)	0.5307 (< 0.01)

Table 8.1: Correlation of the *system* AutoSummENG score to human judgement for peers only (p-value in parentheses)

AutoSummENG to ...	*Spearman*	*Kendall*	*Pearson*
Overall Responsiveness	0.3788 (< 0.01)	0.2896 (< 0.01)	0.3762 (< 0.01)
Linguistic quality	0.1982 (< 0.01)	0.1492 (< 0.01)	0.1933 (< 0.01)

Table 8.2: Correlation of the *summary* AutoSummENG score to human judgement for peers only (p-value in parentheses)

of textual quality that cannot be well estimated at this point in time, like the linguistic quality. As this quality is important and not strongly correlated to the overall responsiveness measure, it seems that the reason for not being able to surpass the current level of performance in evaluating summaries and summarization systems is that we lack *statistically independent* judgements concerning orthogonal aspects of textual quality. If these judgements were performed, we would be able to judge quality better as a composition of the independent judgements.

The tables 8.1 and 8.2 indicate two important aspects of the summarization evaluation. The first has to do with the fact that the AutoSummENG method is good enough to judge system performance rankings. The second indicates that we should conduct research towards a measure that could indicate summary performance, in contrast to system performance. The latter problem is much harder and would also solve the system ranking problem, as the system performance is calculated as the average of the system summaries' performance.

8.2 Combinatorial System Evaluation Using Machine Learning

We attempted to create a meta-estimator of summary system quality using n-grams of various ranks both at the word and character level. The performance of each system was described as a vector, the dimensions of which were the AutoSummENG performance of the system for different configurations (various n-gram sizes, word or character n-grams, various window sizes) as well as ROUGE/BE values. The meta-estimator was created using a set of various machine learning techniques (decision trees, linear regression, multi-layer perceptron, SVM-based regression).

Further investigating the use of meta-evaluators, we wanted to see if the AutoSummENG method, when applied with various parameters values, can offer enough features to better estimate summary system quality. A second question we wanted to answer is whether combining the automatic evaluation methods provides more information than the individual methods do.

The experimental setting involved the corpus of TAC 2008 and we used as input features the following.

Method	*Resp.*			*Ling.*		
	All	*AE*	*Others*	*All*	*AE*	*Others*
Linear R.	*0.915*	0.915	0.903	*0.630*	**0.630**	**0.541**
SMO R.	*0.920*	0.914	0.880	*0.540*	0.567	0.471
Mult. Perc.	**0.928**	0.899	**0.905**	***0.704***	0.547	0.488
ϵ-SVR (LibSVM)	*0.924*	***0.923***	0.903	*0.409*	0.445	0.447
Average	0.922	0.913	0.898	0.571	0.547	0.487

Table 8.3: Combinatorial Evaluation using Machine Learning: Pearson Correlation. Max Performances Indicated as **Bold**.

- Rouge-2 performance, Rouge-SU4 performance
- BE performance
- AutoSummENG performance including character n-grams of various lengths and distances [1] and word n-grams of various lengths and distances[2].

We used the WEKA machine learning platform [WF05] in a 10-fold Cross-Validation experiment to determine how the combination of the aforementioned features performs when estimating Responsiveness or Linguistic Quality. Table 8.3 indicates the performance achieved using three different regression methods, linear regression, SMO regression and multilayer perceptron.

In order to determine whether AutoSummENG outputs are good inputs to the combinatorial estimator, we examined three alternatives. The first (labeled *All*) included all evaluation results as input, the second included only AutoSummENG results (labeled *AE*) and the third all other evaluation results (labeled *Others*). The performance of the meta-estimation indicates that:

- Different versions of AutoSummENG appear to be useful as input to the meta-evaluation process. Together the AutoSummENG evaluation outputs are better features than all the other measures together, as a paired t-test at 90% confidence level has indicated, using the results for Responsiveness in Table 8.3. The results are even more encouraging (95% confidence level) for the difference in performance when estimating Linguistic Quality. However, in our experiments the AutoSummENG features were more numerous than all the other together, which may have biased the results.
- The best results are achieved if one combines all the methods' outputs. However, paired t-tests offered no support, at the 90% level, that using all features will render better results than using only AutoSummENG features.

It would be really interesting to determine what each method contributes to the overall result. Furthermore, one should use other evaluation methodologies, as grammaticality grading [MDWD07, Kel00], to add new (vector) features to the evaluation process, that would ideally be orthogonal to each other.

We have used Principal Component Analysis (see [TK03] for more) to determine more important features, but the analysis showed that all features are

[1]Parameters for character n-grams $(L_{\text{min}}, L_{\text{MAX}}, D_{\text{win}}) \in \{(3,3,3),(5,5,5),(7,7,7)\}$.

[2]Parameters for word n-grams $(L_{\text{min}}, L_{\text{MAX}}, D_{\text{win}}) \in \{(1,1,8),(2,2,8),(3,3,3)\}$.

of equal importance. This have us a hint that using existing evaluation techniques combined does not offer the expected results. The reason for this can be explained by correlation: the results of existing evaluation methods are highly correlated statistically (also see [Dan05, Dan06]). This is normal, because they all aim to give an overall judgement of responsiveness.

At this point, we considered how we can detect and measure other textual quality aspects. To research these aspects we devised and tested whether there can be a statistical measure that can classify summaries as human and non-human, hoping to uncover the idiosyncrasy of human writing. We present this study in the following chapter.

Chapter 9

String Sequence Normality

Identifying the set of qualities that render a text understandable and fluent is a problem that has been apparent in machine translation (MT), natural language generation (NLG) and automatic summarization. In close relation to this, this study focuses on the notion of normality of a textual document with respect to another, aiming to relate this concept to grammaticality and fluency of summaries.

Having the objective to detect qualities related to the 'goodness' of summaries, in relation to grammaticality and fluency (see section 2.3), this chapter studies statistically-extracted features of humanly-created summaries, in contrast to automatically extracted summaries. The aim is to detect invariant features of humanly created summaries so as to devise summarization systems that shall produce summaries having these features. We conjecture that such summaries shall be rated high with respect to grammaticality and fluency: However, this is something to be shown. Towards this aim, this work proposes and studies a statistical measure, named String Sequence Normality (SSN) of a document with respect to another. This is a measure applied to a specific representation of documents based on sequences of strings appearing in them. We study the application of this measure using these analysis of various granularity (character and word n-grams of various ranks) over a set of humanly and automatically created multi-document summaries from two different corpora.

Using the String Sequence Normality (SSN) of a summary t_1 with respect to a 'reference' document t_2, there is no need for a given grammar so as to assess the grammaticality of t_1, as our grammar is the one used in the 'reference' text t_2: The one with respect to which the summary is evaluated concerning its normality. Furthermore, given that the String Sequence Normality (SSN), is closely related to the frequence of appearance of strings, we can state that the 'reference' grammar is a by-product of computing the measure.

The state of the art contains various kinds of evaluators concerning grammaticality and fluency, which are both indicators of acceptability and normality of text. Our method is related to these evaluators, according to the folowing:

- It uses a model corpus that is being used as a 'reference' for measuring the normality of a document.
- It derives patterns of symbol sequences from the model corpus, providing the 'constraints' for 'measuring' grammaticality.

- It uses machine learning methods to discriminate between human and machine-generated texts, by exploiting the normality of texts.

The factors that differentiate the presented method over related ones, are as follows:

- The method does not require extracting the grammar from a text; The aim is to determine *normality* of a text given a reference — *i.e.* model — corpus. Both types of texts (evaluated and reference) are represented as sequences of symbols.
- The method does not require preprocessing of the input texts in any way. Only word splitting is used in cases we use word n-grams as symbols.
- The proposed method requires no background knowledge of the language used. Doing so, it may function independently of language.
- The method may exploit word as well as sub-word structure information, by the use of word or character n-grams of various ranks.
- The method supports variable granularity, allowing to detect different types of normality, from word spelling to syntax.

As already stated, we have been motivated towards measuring normality of summaries with respect to a given corpus so as to detect those invariants of humanly created summaries, which could be applied to automatically created summaries to have them rank high with respect to grammaticality and fluency. As such, normality provides a measure for evaluating summaries and, thus, summarization systems.

The chapter is structured as follows. We present the Statistical String Sequence Representation of textual documents and the definition of the String Sequence Normality, in section 9.1. Experiments over well-known corpora follow in section 9.2, proving the validity of the proposed measure and providing results concerning the qualities of humanly created summaries in contrast to automatically created ones. We close the chapter with the conclusions and the lessons learned from the study, in section 9.3.

9.1 String Sequence Normality

To exploit String Sequence Normality so as to study the invariant features of humanly-created summaries related to grammaticality and fluency, we need a representation of texts in terms of sequences of symbols. This representation must be parametric in terms of symbols' granularity, allowing comparisons.

The proposed representation, which we call Statistical String Sequence Representation (*SSS-R*), is *a set of triples* of the form $<(F,S),D>$. This includes a *pair* (F,S) and *a corresponding distribution for each pair* D. The first part F of each pair is a sequence of strings. The second part S is a single string. The granularity of strings, being a single letter, a word or a whole sentence, is a parameter of the representation. The distribution D for a given pair describes the number of co-occurrences of F and S in the text as a function of the distance between them, up to a maximum distance d_{max}. This distance is measured as the number of strings from F to S in the text. Therefore, this distance depends on the granularity of the strings used as well. From now on we denote such a representation, as a set of triplets of the form:

$F \rightarrow S(D)$, where $D \equiv$ (distance1 $\Rightarrow$ numberOfOccurences1
distance2 $\Rightarrow$ numberOfOccurences2...)

Given a distance x between F and S, $D(x)$ specifies the number of occurrences of F and S in the text with a distance x among them.

To determine the *SSS-R* of any text, we need to specify the following set of parameters: The n-gram rank r of F, the maximum distance d_{max} of F and S co-occurrence, as well as the type of strings identified in the text (*e.g.* character, word, sentence). Thus, we will use the form *SSS-R*$(r, d_{max}$, *stringType*) to fully describe an *SSS-R*. Such a representation describes all the possible sets of triples, for any text: However, given a specific text t, then its representation given the parameters for *SSS-R* is denoted *SSS-R*$(r, d_{max}, \text{StringType})(t)$: This is a set of triplets which is *an instance of* *SSS-R*$(r, d_{max}, \text{SymbolType})$.

Example 9.1.1 *The sentence:*
S: 'A big big test.'
is represented as SSS-R$(1, 1, character)(S)$ *by the following set of triplets:*
$t \rightarrow e(1 \Rightarrow 1.0)$
$\_ \rightarrow b(1 \Rightarrow 2.0)$[1]
$A \rightarrow \_(1 \Rightarrow 1.0)$
$\_ \rightarrow t(1 \Rightarrow 1.0)$
$b \rightarrow i(1 \Rightarrow 2.0)$
$t \rightarrow .(1 \Rightarrow 1.0)$
$e \rightarrow s(1 \Rightarrow 1.0)$
$g \rightarrow \_(1 \Rightarrow 2.0)$
$s \rightarrow t(1 \Rightarrow 1.0)$
$i \rightarrow g(1 \Rightarrow 2.0)$
S may also be represented by the following instance of SSS-R*(2,2,word):*
$big, big \rightarrow test(1 \Rightarrow 1.0)$
$a, big \rightarrow test(2 \Rightarrow 1.0)$
$a, big \rightarrow big(1 \Rightarrow 1.0)$

Therefore, the first set of triplets is an instance of SSS-R*(1,1,character), while the second is an* instance of SSS-R*(2,2,word).*

Given a text t, let T be the corresponding instance of *SSS-R*$(r, d_{max}$, *symbolType*) representing this text. This is denoted as follows:

$$T \equiv SSSR(r, dMax, symbolType)(t) \iff$$
$$T \dashv \textit{SSS-R}(r, d_{max}, \text{symbolType}) \iff$$
$$T \text{ is an instance of } \textit{SSS-R}(r, d_{max}, \text{symbolType})$$

We define the similarity *sim* between two distributions D_1, D_2 to be the sum of the absolute differences of the corresponding *non-zero elements of each distribution.* If X is the set of values $\{x | x : D_1(x) > 0 \text{ or } D_2(x) > 0\}$, then: $sim(D_1, D_2) = \sum_{i \in X}(abs(D_1(i) - D_2(i)))$, where abs is the absolute value function.

On the same basis, the similarity of two triplets A, A', $\text{simT}(A, A')$ equals to the similarity of their distributions D, D', $\text{sim}(D, D')$, given that the two first elements of the triples are identical. Else, we define $\text{simT}(A, A') = 0$.

[1]The underscore at the beginning of the line is actually the blank character (space).

Given that T_1, T_2 are the corresponding instances of *SSS-R*$(r, d_{max},$ *string-Type*$)$ for two texts t_1 and t_2, then we define the *SSN* normality of t_1 with respect to the reference text t_2, as follows:

Definition 9.1.2
normality$(T1|T2) \equiv T_1 \sim T_2 = \frac{\sum_{A \in T_1, A' \in T_2} simT(A,A')}{|T_1|}$*, where* $|T_1|$ *is the number of triplets in* T_1 *and* $T_1, T_2 \dashv$ SSS-R$(r, d_{max}, stringType)$*.*

Given a corpus $C = T_1, T_2, ..., T_n, n \in \mathbb{N}^*$, In order to define the *SSS-R* of this corpus we concatenate the texts in the corpus to produce a single text. Doing so, we can use the corpus as a reference text, measuring the normality of any document representation in *SSS-R*s with respect to this corpus. This normality is what we call *SSN*.

9.2 Experiments

The data on which our experiments were conducted are the summaries and evaluation corpora of DUC 2006 and DUC 2007. DUC 2006 consists of summaries for 50 different topics, as well as the corresponding 25 input documents per topic. Summaries were generated from these texts. Each topic has a number of automatically extracted summaries, one for each participating system, and 4 humanly created summaries. The humanly created summaries are differentiated by means of an identifier. All summaries were truncated to 250 words before being evaluated. DUC 2007, similarly to the one of DUC 2006, consists of summaries for 45 different topics. Each topic has 4 humanly created summaries, as well as 28 machine generated summaries. In the corpora the humanly created summaries appeared both as 'models' and 'peers'. However, in this study we label both occurrences of human summaries as 'human' and those created by summarization systems as 'peer'.

In order to have a set of baseline-quality summaries, we have created a single summary for each topic in the DUC2006 corpus by randomly gathering words from the 25 input documents. The words have been selected so that their frequencies in the summary would tend to be the same as in the input documents. The length of each summary is about 250 words (length chosen from a Poisson distribution averaging to 250). In DUC 2007 we did not insert random summaries, so that we could check the effects of having a baseline on our methodology.

To detect those features that are vital to distinguishing human from automatic summaries, we have conducted the following process over two different corpora:

- For a given topic, the input documents were concatenated to a single document which was represented as an instance of *SSS-R*$(i, j, character)$, where $1 \leq i \leq 8, j = i$, and as an instance of *SSS-R*$(k, l, word)$, where $1 \leq k \leq 3, l = k$. We chose $j = i$, *i.e.* single-rank analysis, to determine the individual effect of different ranks in the process.

- Each summary document, either humanly or automatically-generated was represented as an instance of *SSS-R*$(i, j, character)$, where $1 \leq i \leq 8, j = i$, and as an instance of *SSS-R*$(k, l, word)$, where $1 \leq k \leq 3, l = k$.

Table 9.1: Detailed Accuracy By Class - Simple Naive Bayes DUC 2006

Class	TP	FP	P	R	F
Peer	0.967	0.033	0.993	0.967	0.979
Human	0.968	0.033	0.866	0.968	0.914

Table 9.2: Confusion Matrix - Simple Naive Bayes DUC 2006

Classified As		
Peer	Human	Actual Class
1740	60	Peer
13	387	Human

- We compared each summary text representation $T_{SSS\text{-}R}$ to the corresponding topic representation $C_{SSS\text{-}R}$, creating a feature vector. The values of each vector were the results of $T_{SSS\text{-}R} \sim C_{SSS\text{-}R}$ for each of the *SSS-R* configurations. Each 11-dimensional vector was labeled by a label $L \in \{human, peer\}$. The 'peer' label was assigned to automatic summarizer documents, including our baseline documents.
- We used a simple Naive Bayes classifier to classify human and peer texts. Also, we used a kernel-estimating Naive Bayes classifier, as an alternative. In both cases 10-fold stratified cross-validation was performed to determine the effectiveness of the method (see the WEKA toolkit [WFT+99]). Furthermore, we have used an SVM classifier as well. The classifiers were selected based on their common use and good performance in a variety of classification tasks.
- We calculated the feature's Information Gain, and performed Principal Component Analysis to determine the important features used for the classification decision.

We performed the same process on a second corpus (DUC 2007) to verify our results.

9.2.1 Classification Between Human and Machine-Generated Summaries

In Table 9.1 we see the results of the naive Bayes classification. The columns' labels are as follows: Class is the document class (label), TP is the rate of true positives, FP is the rate of false positives, P is the precision, R is the recall and F is the F-measure. Given the fact that the F-measure for both classes exceeds 90%, it appears that the method has been highly effective in classifying summaries.

It is impressive that we need not apply more complex classifiers; this provides evidence for the features being appropriate. In Table 9.2 we see the numbers of correctly classified and misclassified instances. Once more, the features seem to be appropriate.

We attempted, however, to use a more advanced classifier to see if it is easy to reach the maximum F-measure. Using Bayes that considers multinomial distribution for features, as well as an SVM classifier (C-SVM) with a high cost

Table 9.3: Confusion Matrix - Multinomial Naive Bayes DUC 2006

Classified As		
Peer	Human	Actual Class
1780	20	Peer
4	396	Human

Table 9.4: Confusion Matrix - C-SVM DUC 2006

Classified As		
Peer	Human	Actual Class
1797	3	Peer
1	399	Human

parameter (as found in the LibSVM implementation [CL01]) we got the results shown in tables 9.3, 9.4.

At this point we checked whether the SVM model produced overfits the DUC 2006 corpus. Thus, we evaluated the model on the DUC 2007 corpus data. Impressively, the results were quite similar, as can be seen in Table 9.5, amounting to an F-measure of over 97% for both classes. Then, we evaluated the model of the DUC 2007 corpus on the DUC 2006 data. The results of this experiment, which are described in Table 9.6 show that we had an increased number of false negatives for the human class. However, this is most probably the effect of not including baseline texts in the experiments conducted on the corpus of DUC 2007. In any case, the application of the learning process yields comparable results for both corpora (see also 9.7 for the Bayes performance on DUC 2007).

The experimental results, therefore, proved that the use of *SSS-R* as the means to represent the corpus and the summaries, along with the use of *SSN* for the computation of summaries' normality with respect to the 'reference' text (model corpus), provide good enough features to tell human and automatically generated summaries apart.

9.2.2 Feature Importance - PCA Application

Given the success of the classification process, i.e. the success of detecting the humanly-created summaries (correspondingly the automatically-created ones), we proceeded to detect the key features of the classification process: These form the features that can be conjectured to be the invariants for the humanly-created summaries, differentiating them from the automatically-created summaries. Towards this, we have used two methods to decide upon the answer:

- We ranked the features according to their Information Gain (see [MS99,

Table 9.5: Confusion Matrix - C-SVM model of DUC2006 applied on DUC2007

Classified As		
Peer	Human	Actual Class
1439	1	Peer
18	342	Human

Table 9.6: Confusion Matrix - C-SVM model of DUC2007 applied on DUC2006

Classified As		
Peer	Human	Actual Class
1797	3	Peer
335	65	Human

Table 9.7: Confusion Matrix - Multinomial Naive Bayes DUC 2007

Classified As		
Peer	Human	Actual Class
1418	22	Peer
8	352	Human

p. 583] for linguistic uses of the measure).

- We performed Principal Component Analysis [Wol87].

The information gain calculation gave the ranking of Table 9.8. In the table, the *SSS-R* used was named using 'char' or 'word', indicating the kind of symbol that was used for the representation and the $r = d_{max}$ parameter value. For instance, Char2 indicates single-character strings with $r = d_{max} = 2$. The table presents ranking for both DUC 2006 and 2007 corpora, on the left and right part, correspondingly.

The application of PCA on both corpora, DUC 2006 and DUC 2007, brought a pleasant surprise: the most important Principal Components extracted from both corpora were very similar. Both the absolute values of weights of the original features in the PCA-extracted features, as well as the eigenvalues of the major principal components themselves were similar (see Table 9.9). This indicates emergent important features, only partially dependent on the corpus.

In both rankings, it seems that the low-ranked character n-grams simply reproduce the spelling constraints of a language and offer no useful information. The most important features appear to be the high-rank character n-grams: these span over more than one word. These features are the ones detecting word collocations and other similar phenomena. Using *only Char7 and Char8*

Table 9.8: Ranking of Features using Information Gain

Rank	*IG 2006*	SSS-R	*IG 2007*	SSS-R
1	0.6528	Char8	0.6769	Char7
2	0.6527	Char7	0.67525	Char8
3	0.6463	Char6	0.67394	Char6
4	0.6161	Char5	0.61962	Char5
5	0.3703	Char4	0.35862	Char4
6	0.0545	Char3	0.06614	Char3
7	0.0256	Word3	0.01098	Char1
8	0.0196	Char1	0.0078	Char2
9	0.0133	Word1	0	Word2
10	0	Word2	0	Word3
11	0	Char2	0	Word1

Table 9.9: Major Features by PCA

Corpus	*Eigenvalue*	*Feature Formula*
DUC 2006	5.62218	0.414Char4+0.409Char5 +0.389Char6
DUC 2007	5.70926	-0.411Char4-0.397Char5 -0.372Char6

Table 9.10: Confusion Matrix - 2 Features - Multinomial Naive Bayes DUC 2006

Classified As		
Peer	Human	Actual Class
1784	16	Peer
8	392	Human

features we reached a very high performance, displayed in Table 9.10.

Furthermore, figures 9.1, 9.2, 9.3 show the effect of using different features for the discrimination between human and peer summaries: The light colored (yellow) areas indicate instances of SSSR representations of human summaries and the dark colored (blue) indicate instances of SSSR representations of peer summaries. It is obvious that high-rank character n-grams discriminate between classes, because humans have *lower SSN* in high ranks than automatic summaries, but *higher SSN* than random texts.

Summarizing the above, what we must notice is the importance of sub-word (character) features of high rank. However, it is not the spelling that makes the difference, but the joining of words. Also, studying the results we can conjecture that normality indicates whether a text is the result of an abstraction process: This is true given that people (who follow an abstractive summarization process) have lower *SSN* performance than automatic summarization systems (that follow mostly an extractive summarization process), but higher than random texts (that in no way follow the patterns in the corpus).

Figure 9.1: Character 1-grams *SSN* distribution for DUC 2006

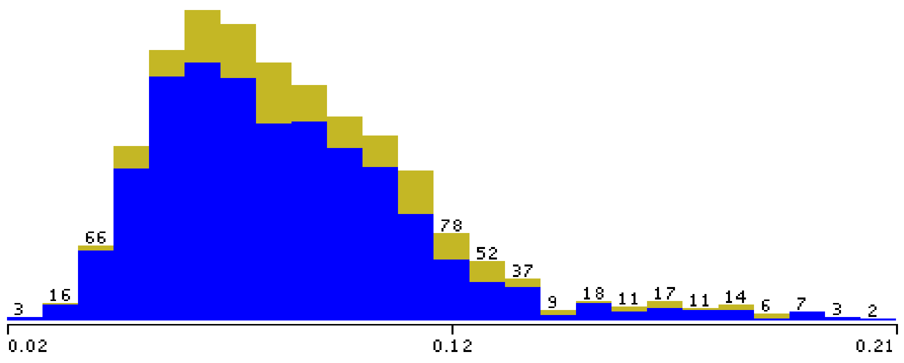


Figure 9.2: Character 4-grams *SSN* distribution for DUC 2006

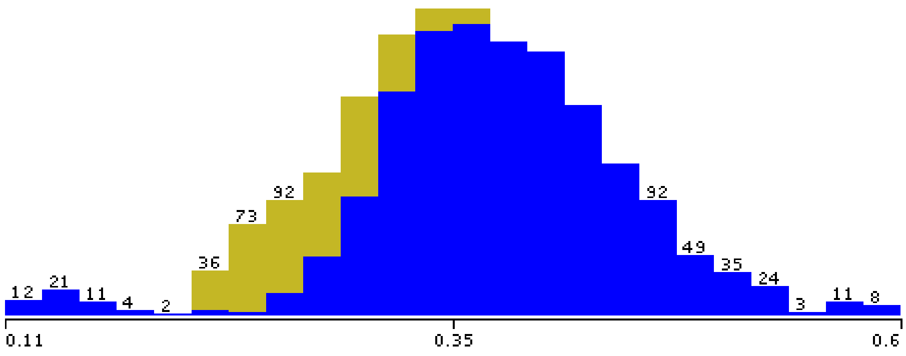
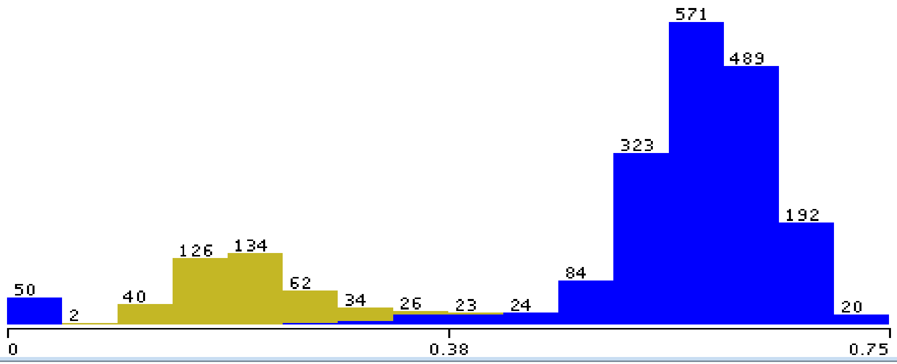


Figure 9.3: Character 8-grams *SSN* distribution for DUC 2006 The 50 automatic texts with low grammaticality are the random instances



9.3 Conclusions on Evaluation

From the study presented we have inferred a number of facts:

- The AutoSummENG method for summarization evaluation is a promising method based on language-neutral analysis of texts and comparison to gold-standard summaries. The method is based on n-gram graphs, even though it provides support for other, histogram-based, approaches. We have found that the character n-gram graph representation, including information of neighbourhood frequency, can render the best results for the given task. The presented method appears to outperform current approaches in the corpus of DUC 2006, providing a good evaluation alternative for future attempts.

- Answering the questions posed in chapter 5, statistical information related to co-occurrence of character n-grams seem to provide important information concerning the evaluation process of summary systems. The actual representation used for capturing this information can be an n-gram graph, as this has been described within out method, with parameters optimized a priori. The distance metric to be preferred would be the Value Similarity between the graph representation of peer summaries and model summaries. Our method, complemented by the parameter optimization step, has proved to be a language-neutral, fully automated, context-sensitive method with competitive performance.

- The combination of individual evaluation measures can prove fruitful in improving the evaluation of summaries. However, research efforts should focus on evaluating different, orthogonal qualities of text in order to achieve higher overall evaluation performance.

- Many existing automatic summarization systems, which are based mostly on extractive techniques, appear to share statistical features. There is such a feature that can tell human summaries apart from automatically generated ones. This is the proposed String Sequence Normality, *SSN*.

- Human summaries tend to have *lower SSN* values than automatically generated summaries. This may be directly connected to the abstractive nature of multi-document summarization process of humans. On the other hand, human summaries tend to have *higher SSN* values than summaries randomly generated.

- The principal components, based on *SSN*, that discriminate humanly-created from automatically-generated summaries for a given language seem to follow a specific pattern of weights. This indicates that humans do follow statistically traceable patterns of text generation if we get to the sub-word level.

In an effort to evaluate automatic texts (summaries) in accordance to the human perception of fluency and grammaticality, the presented *SSN* measure adds one more language-neutral and objective tool. It would be very important to determine other, perhaps similar measures that will be able to detect other invariants of human texts. Doing so, step by step, our intuition of the complex process of summarization will be augmented and, hopefully, we shall be able to design and implement better automatic summarization systems. In our research, we have begun to utilize *SSN* in the process of sentence reformulation for summarization.

Having researched the way a summary should be evaluated, we decided to create a summarization system that would hold a set of desired attributes, as illustrated through the gaps in current literature:

- The system should be as language-independent as possible.
- The system should aim to use background knowledge, in the form of an ontology or a thesaurus.

This system we devised and implemented will be presented in part III covering such summarization subtasks as Subject Analysis, Data Analysis, Feature Generation and Representation, Summary Representation and Summary Generation.

Part III

Using n-gram graphs in extractive summarization

Chapter 10

Salience Detection and Redundancy Removal Using N-gram Graphs

Within this part we tackle the problems of salience detection and redundancy control using a unified framework based on n-gram graphs. The contributed methodologies offer language-neutrality in the subject analysis, data analysis and summary representation steps of the summarization process, under an easily adaptable set of tools. The underlying theory, even though it is based on a rigorous formalization, is based upon the elementary intuition that neighbourhood and relative position between characters, words and sentences offers more information than the 'bag-of-words' approach. In the following chapters we present our proposed methodology (section 10.1) and perform a set of experiments to support its applicability and evaluate its performance (chapter 10.5).

In the presented methodology, the algorithms we use are fully language-independent. Furthermore, our method uses the n-gram graph logic [GKVS08] for sentence and chunk similarity, which overcomes the need for preprocessing. Even the chunking process, used to separate a sentence into subsentential strings, is based upon statistics. Furthermore, we define a methodology for the mapping of a sentence to a set of concepts provided by an external source (section 10.1.1), that we have used for query expansion.

10.1 Proposed Methodologies for Extractive Summarization Subtasks

In the following paragraphs we present both the theory and the devised tools used throughout the salience and redundancy detection. It is important that a single theoretical and practical framework allows for different applications on the Natural Language Processing (NLP) domain.

In order to understand the n-gram graph use, one should take into account the fact that neighbourhood between different linguistic units is a very important factor for determining the meaning of these units. Contextual information has been very widely used and several methodologies have been built upon its value (e.g. [BLL98, Yar95]).

In psychology there is the 'magic number seven' [Mil56], indicative of the upper limit of human immediate memory capacity and communication ability. Most operations concerning the perception, processing and storage of information is limited by this number of input stimuli. Clustering of these stimuli into groups allows for more complex perception, processing and storage. As Miller puts it 'By organizing the stimulus input simultaneously into several dimensions and successively into a sequence or chunks, we manage to break (or at least stretch)' the 'informational bottleneck' of being able to process only 7 ± 2 items. Within this work we consider that neighbourhood between characters, words or sentences offers the ability to represent textual information in a way similar to human cognition as small groups of stimuli. Thus, we have adapted a set of mathematical tools that implement NLP operations using information concerning the neighbourhood of characters or strings, without the need for background language information.

The summarization subtasks we address in the following sections are:

- Query expansion, which aims to improve the subject analysis step of the summarization process.
- Salience detection, as part of the summary representation step.
- Redundancy removal of the summary representation step especially for multi-document summaries.

The methodologies we describe also indicate a new type of feature for the *feature generation and representation* step of the summarization specification: the n-gram graph.

In order to perform the aforementioned subtasks, we describe two tools of statistical analysis we have devised and used in our methodology:

- *Next-character entropy text chunking* to be able to analyse a sentence into its constituent parts, regardless of the underlying language.
- The *semantic index*, which provides a mapping function between a sentence and a set of concepts.

In the following paragraphs, the following basic assumptions have been made.

- The *content* $C_{\mathbb{U}}$ of a *text set (corpus)* $\mathbb{U}$ is considered to be the intersection of all the graph representations of the texts in the set: $C_{\mathbb{U}} = \bigcap^{t \in \mathbb{U}} t$. This assumption indicates that we consider content of a document set the common substrings between all documents.
- A sentence S is considered more similar to the content $C_{\mathbb{U}}$ of a text set, as more of the sentence's *chunks* (sub-strings of a sentence) have an n-gram graph representation similar to the corresponding content representation. Every chunk's similarity to the content is added for the overall similarity of a sentence to the content.

  This assumption is based on the fact that sentences similar in meaning will contain similar substrings. The more common substrings one finds between two sentences, the higher the similarity in meaning (even though this is obviously *not* always the case).

10.1.1 Next-character Entropy Chunking

In order to perform chunking we use a corpus to determine the probability $P(c|S_n)$ that a single given character c will follow a given character n-gram S_n, for every character c apparent in the corpus. The probabilities can then be used to calculate the entropy of the next character for a given character n-gram S_n.

The entropy measure indicates uncertainty. We have supposed that substrings of a character sequence where the entropy of $P(c|S_n)$ surpassed a statistically computed threshold represent candidate delimiters. Within the text to be chunked we seek the delimiters, after the end of which a new chunk begins. In our application we have only checked for unigrams, *i.e.* simple letters, as delimiters even though delimiters of higher rank can be determined. For example, in bi-gram chunking the sequence ',␣' (comma and space) would be considered to be delimiter, while in unigrams the space character only would be considered delimiter. Given a character sequence S_n and a set of delimiters $\mathbb{D}$, our chunking algorithm splits the string after every occurrence of a delimiter $d \in \mathbb{D}$.

10.1.2 Mapping a Sentence to a Set of Concepts Using External Knowledge

Within the scope of our work we tried to extract concepts from a sentence. Usually this happens by looking up, possibly preprocessed, words in thesauri like WordNet. In our approach we have used a decomposition module based on the notion of the *symbolic graph.*

A *symbolic graph* is a graph where each vertex contains a string and edges are connecting vertices in a way indicative of a *substring* relation. As an example, if we have two strings *abc, ab* labeling two corresponding vertices, then since *ab* is a substring of *abc* there will be a directed edge connecting *ab* to *abc.* In general, the symbolic graph of a given text T contains every string in T and for every string it illustrates the set of substrings that compose it. This graph's size is exponential to the size of the input text, therefore we choose an upper limit to the size of substrings apparent within the graph. The symbolic graph should ideally use the *symbols* defined in part II, however in terms of implementation simplicity we chose to use every string as a possible symbol within this part of the research.

When a symbolic graph has been constructed, then one can run through all the vertices of the graph and look up each vertex in a thesaurus to determine if there is a match. If the thesaurus contains a looked up vertex string then the vertex is assigned the corresponding looked up *meaning.* This annotated graph, together with a facility that supports comparing meanings is what we call the *semantic index.*

The semantic index, therefore, represents links between n-grams and their semantic counterparts, implemented as *e.g.* WordNet definitions which are textual descriptions of the sense. Such definitions are used within example 10.1.1. If D_1, D_2 are the sets of definitions of two terms t_1, t_2, then to compare the semantics (meaning) of m_1, m_2 of t_1, t_2 using the semantic index, we actually compare the n-gram graph representation $G_{1i}, G_{2j}, 1 \leq i \leq |D_1|, 1 \leq j \leq |D_2|$ of each pair of definitions of the given terms. Within this section we consider the meaning of a term to map directly to the set of possible senses the term has. The similarity of meaning $\text{sim}_{\text{Meaning}}$ is considered to be the *averaged sum* *of the similarities* over *all pairs of definitions of the compared terms*:

$$\mathrm{sim}_{\mathrm{Meaning}}(t_1, t_2) = \frac{\sum_{G_{1i}, G_{2j}} \mathrm{sim}(G_{1i}, G_{2j})}{|D_1| \times |D_2|} \tag{10.1}$$

This use of similarity implies that uncertainty is handled within the measure itself, because many alternative senses, *i.e.* high $|D_1|, |D_2|$, will cause a lower result of similarity. An alternative version of the similarity measure, that only requires a single pair to be similar to determine high similarity of the meanings is the following.

$$\mathrm{sim}_{\mathrm{Meaning}}{}'(t_1, t_2) = \max_{G_{1i}, G_{2j}} \mathrm{sim}(G_{1i}, G_{2j}) \tag{10.2}$$

Within our examples in this section we have used equation 10.1.

Example 10.1.1 *Compare: smart, clever*
WordNet sense definitions for 'clever':

- *cagey, cagy, canny, clever – (showing self-interest and shrewdness in dealing with others; 'a cagey lawyer'; 'too clever to be sound')*
- *apt, clever – (mentally quick and resourceful; 'an apt pupil'; 'you are a clever man...you reason well and your wit is bold'-Bram Stoker)*
- *clever, cunning, ingenious – (showing inventiveness and skill; 'a clever gadget'; 'the cunning maneuvers leading to his success'; 'an ingenious solution to the problem')*

WordNet sense definitions for 'smart':

- *smart – (showing mental alertness and calculation and resourcefulness)*
- *chic, smart, voguish – (elegant and stylish;'chic elegance';'a smart new dress';'a suit of voguish cut')*
- *bright, smart – (characterized by quickness and ease in learning;'some children are brighter in one subject than another';'smart children talk earlier than the average')*
- *fresh, impertinent, impudent, overbold, smart, saucy, sassy, wise – (improperly forward or bold;'don't be fresh with me';'impertinent of a child to lecture a grownup';'an impudent boy given to insulting strangers';'Don't get wise with me!')*
- *smart – (painfully severe;'he gave the dog a smart blow')*
- *smart – (quick and brisk;'I gave him a smart salute';'we walked at a smart pace')*
- *smart – (capable of independent and apparently intelligent action;'smart weapons')*

Similarity: 0.0794

t_1	t_2	$\text{sim}_{Meaning}$
smart	stupid	0.0388
smart	pretty	0.0339
run	operate	0.0768
run	walk	0.0436
run	jump	0.0416
run	die	0.0557
hollow	empty	0.0893
hollow	holler	0.0768

Table 10.1: Other examples of comparisons: using sense overview

t_1	t_2	$\text{sim}_{Meaning}$
smart	clever	0.0000
smart	stupid	0.0020
smart	pretty	0.0036
run	operate	0.2162
run	walk	0.0020
run	jump	0.0017
run	die	0.0152
hollow	empty	0.1576
hollow	holler	0.3105

Table 10.2: Other examples of comparisons: using only synonyms from overview

In Table 10.1, we offer some more pairs of terms and their corresponding similarity values. These primary results indicate that, even though the measure appears to have higher values for terms with similar meaning, it may be biased when two words have similar spelling. This happens because the words themselves appear in their definition, which causes a partial match between otherwise different definitions.

The results further depend heavily on the textual description — *i.e.* definition — mapped to any term's individual sense (synset in WordNet). The results for the same examples when using synonyms only as descriptors of individual senses can be seen in Table 10.2. We notice that the words 'smart' and 'clever' are found to have no relation whatsoever, because no common synonyms are found within the WordNet results[1]. Furthermore, since the given word always appears in its synonym list, word substring similarity still plays an important role, *e.g.* 'hollow' and 'holler'.

The use of a semantic index is that of a meaning look-up engine. The semantic index is actually an annotated symbolic graph. If there is no matching vertex in the graph to provide a meaning for a given input string then the string is considered to have the meaning of its closest, in terms of graph path length, substrings that have been given a meaning. This 'inheritance' of meaning from short to longer strings is actually based on the intuition that a text chunk contains the meaning of its individual parts. Furthermore, a word may be broken down to elementary constituents that offer meaning. If one uses an ontology or

[1]We have used the overview option in this set of experiments and only kept the synonyms through regular expressions on the WordNet result.

even a thesaurus including prefixes, suffixes or elementary morphemes and their meanings to annotate the symbolic graph, then the resulting index becomes quite a powerful semantic annotation engine.

Within this work, we have combined a symbol graph and WordNet into a semantic index to annotate queries with meanings and perform query expansion.

10.2 Query Expansion

Query expansion is based on the assumption that a set of words related to an original query can be used as part of the query itself to improve the recall and usefulness of the returned results. In the literature much work has indicated that query expansion should be carefully applied in order to improve results [Voo94, QF93].

In our approach, we have used query expansion in a very simplistic way, looking up all query words in WordNet [MBF+90] and appending the resulting *WordNet's 'overview of senses'*-contained words to the query. An example overview of senses for the word 'ambiguous' can be seen in example 10.2.1.

Example 10.2.1 `Overview of verb test`

```
The verb test has 7 senses (first 3 from tagged texts)

1. (32) test, prove, try, try out, examine, essay --
(put to the test, as for its quality,
or give experimental use to;
"This approach has been tried with good results"; "Test this recipe")
2. (9) screen, test --
(test or examine for the presence of disease or infection; "screen
the blood for the HIV virus")
3. (4) quiz, test --
(examine someone's knowledge of something; "The teacher tests us every week";
"We got quizzed on French irregular verbs")
4. test -- (show a certain characteristic when tested;
"He tested positive for HIV")
5. test -- (achieve a certain score or rating on a test;
"She tested high on the LSAT and was admitted to
all the good law schools")
6. test -- (determine the presence or properties of (a substance))
7. test -- (undergo a test; "She doesn't test well")

Overview of adj ambiguous

The adj ambiguous has 3 senses (first 3 from tagged texts)

1. (9) equivocal, ambiguous -- (open to two or more interpretations;
or of uncertain nature or significance;
or (often) intended to mislead; "an equivocal statement";
"the polling had a complex and equivocal (or ambiguous) message for
potential female candidates";
"the officer's equivocal behavior increased the victim's uneasiness";
"popularity is an equivocal crown";
"an equivocal response to an embarrassing question")
2. (4) ambiguous -- (having more than one possible meaning;
"ambiguous words"; "frustrated by ambiguous instructions,
the parents were unable to assemble the toy")
3. (1) ambiguous -- (having no intrinsic or objective meaning;
not organized in conventional patterns; "an ambiguous situation
with no frame of reference"; "ambiguous inkblots")
```

This approach does not function effectively, because much noise is inserted within the query; on the other hand, this experimentation with query expansion offers some insight concerning the usefulness of query expansion for our approach on the query-based summarization task.

For a given word w, a set of senses' overviews is returned by the semantic index; from these senses $s_i, i > 0$ we only utilize senses s_j with graph representations G_{s_j} that have more in common with the content $C_{\mathbb{U}}$ than a given threshold (see section 4.1 for the definitions of the used functions): $G_{s_j} \cap C_{\mathbb{U}} \neq \emptyset$ and $\mathrm{VS}(G_{s_j}, C_{\mathbb{U}}) > t, t \in \mathbb{R}^+$. Finally, the query is integrated in the content definition by merging the representation of the original query G_q and the representations G_{s_j} of all the j additional extracted senses to the original content, giving a new query-based content definition $C_{\mathbb{U}}'$. Having calculated $C_{\mathbb{U}}'$, we can judge important sentences simply by comparing the graph representation of each sentence to the $C_{\mathbb{U}}'$. The refer to the removal of noisy definitions from the overview of senses as our *sense filter*.

Even though the query expansion process was finally rather successful, in the original query expansion process noise was added, due to chunks like 'an', 'in' and 'o' which were directly assigned the meanings of 'angstrom', 'inch' and 'oxygen' correspondingly (also see experiments in section 10.5). This lowered the evaluation scores of our submitted runs. Using the sense filter as shown here, the deficiency has been avoided.

10.3 The Content Selection Process

Given the content definition and the chunking process, each sentence is assigned a score, which is actually the sum of the similarities of its chunks to the content. This process, we call *chunk scoring*, offers an *ordered list of sentences* $\mathbb{L}$. Another alternative, we call *sentence scoring*, would be to assign to each sentence its similarity to the content, without chunking. Both of these alternatives have been examined experimentally in section 10.5.

Given the sentences' ordered list, a naive selection algorithm would select the highest-scoring sentences from the list, until the summary word count limit is reached. However, this would not take redundancy into account and, thus, this is where redundancy removal comes in.

10.4 The Tackling of Redundancy – Redundancy and Novelty

10.4.1 Novelty

The novelty detection process has two aspects, the *intra-summary* novelty and the *inter-summary* or *user-modeled* novelty. The intra-summary novelty refers to the novelty of a sentence in a summary, given the rest of the content of the summary. The inter-summary or user-modeled novelty refers to the novelty of information apparent when the summarization process takes into account information already available to the reader (as per the TAC 2008 update task).

In order to ensure intra-summary novelty, one has to make sure that every sentence added only minimally repeats already existing information. To achieve this goal, we use the following process:

- Extract the n-gram graph representation of the summary so far, indicated as G_{sum}.
- Keep the part of the summary representation that does not contain the content of the corresponding document set $\mathbb{U}$, $G'_{\text{sum}} = G_{\text{sum}} \triangle C_{\mathbb{U}}$.
- For every candidate sentence in $\mathbb{L}$ that has not been already used
  - extract its n-gram graph representation, G_{cs}.
  - keep only $G'_{cs} = G_{cs} \triangle C_{\mathbb{U}}$, because we expect to judge redundancy for the part of the n-gram graph that is not contained in the common content $C_{\mathbb{U}}$.
  - assign the similarity between $G'_{cs}, G_{sum'}$ as the sentence redundancy score.
- For all candidate sentences in $\mathbb{L}$
  - Set the score of the sentence to be its rank based on the similarity to $C_{\mathbb{U}}$ minus the rank based on the redundancy score.
- Select the sentence with the highest score as the best option and add it to the summary.
- Repeat the process until the word limit has been reached or no other sentences remain.

In the TAC 2008 corpus, systems are supposed to take into account the first of two sets per topic, set A, as prior user knowledge for the summary of set B of the same topic. In fact, set A contains documents concerning a news item (*e.g.* Antarctic ice melting) that have been published before the documents in set B. We have used the content of the given set A, $C_{\mathbb{U}A}$, in the redundancy removal process by further merging the content of set B, $C_{\mathbb{U}B}$, to G_{sum} after the first step of the process. In other words, the content of set A appears to always be included in the current version of the summary and, thus, new sentences avoid redundancy.

10.4.2 Redundancy

Within this work we have also implemented a method of redundancy removal, as opposed to novelty detection, where redundancy is pinpointed within the original set of candidate sentences: we consider a sentence to be redundant, if it surpasses an empirically computed threshold[2] of overlap to any other candidate sentence. In each iteration within the redundancy removal process each sentence is compared only to sentences not already marked as redundant. As a result of this process, only the sentences never marked as redundant are used in the output summary.

10.5 Experiments

The experiments conducted upon the TAC 2008 corpus were numerous, to research aspects of the summarization process. We consider each variation of our system based on a different parameter set to be a different system, with a different System ID. We have used the AutoSummENG as our system evaluation method, since it correlates well to the DUC and TAC responsiveness measure [GKVS08].

In TAC 2008[3] there were two tasks. The main task was to produce a 100 B•word summary from a set of 10 documents (Summary A). The update task was to produce a 100-word summary from a set of subsequent 10 documents, with the assumption that the information in the first set is already known to the reader (Summary B). There were 48 topics with 20 documents per topic in chronological order. Each summary was to be extracted based on a topic description defined by a title and a narrative query. The summaries were expected to have a maximum length of 100 words. For every topic 4 model summaries were provided to allow for evaluation.

At this point we indicate that various drawbacks exist in using an overall measure like AutoSummENG, ROUGE [Lin04] or Basic Elements [HLZF05] (also see [Bel09] for a related discussion):

- Small variations in system performance are not indicative of real performance change, due to statistical error.
- The measure can say little about *individual summaries*, because it correlates really well when judging a *system*.
- The measure cannot judge performance of intermediate steps, because it judges the output summary only.
- The measure can only judge the summary with respect to the given model summaries.

Given the above restrictions, we have performed experiments to judge the change in performance when using:

[2]The threshold should be computed via experiments or machine learning to relate with human estimated redundancy of information, but this calculation has not been performed in the scope of this work.

[3]See `http://www.nist.gov/tac/publications/2008/presentations/TAC2008_UPDATE_overview.pdf` for an overview of the Text Analysis Conference, Summarization Update Task of 2008.

System ID	CS	SS	RR	ND	QE	NE	Score
1		✓		✓		✓	0.120211
2		✓	✓			✓	0.130345
3	✓		✓		✓		0.121837
4		✓		✓	✓		0.119849
5		✓	✓		✓		0.129929
6	✓					✓	0.125457

Table 10.3: AutoSummENG summarization performance for different settings concerning scoring, redundancy and query expansion. **Legend** CS: Chunk Scoring, SS: Sentence Scoring, RR: Redundancy Removal, ND: Novelty Detection, QE: Query Expansion, NE: No Expansion

- chunk scoring for sentence selection versus sentence scoring.
- redundancy removal versus novelty detection.
- query expansion versus no query expansion.

Before judging the results in table 10.3, we performed an ANOVA (analysis of variance) test to determine whether the System ID — *i.e.* system configuration — is an important factor for the AutoSummENG similarity of the peer text to the model texts. It was shown with a p-value below 10^{-15} that both the topic for which the summary was performed and the System ID is indeed important for the summary performance. This indicates that:

- There are topics of various difficulty and the topic is an important factor for system performance.
- Selection of different components for the summarizer, from the range of our proposed components, can affect the summaries' quality.

The systems using chunk scoring have no statistically significant difference in performance from the ones that use sentence scoring, as the t-test gave a p-value of 0.64. However, the systems using chunk scoring, namely systems 3 and 6, had a slightly lower average performance than the others. The systems using redundancy removal appear to have statistically significant difference in performance from the ones that use novelty detection, nearly at the 0.05 confidence level (one-sided t-test). System 6 was chosen to not use any redundancy removal method and performs near the average of all other systems, thus no conclusion can be drawn. Concerning query expansion, it was not proved whether query expansion indeed offers improvement, as the t-test gave a p-value of 0.74.

In table 10.4 information on the average performance of TAC 2008 participants over all topics is illustrated. More on the performance of TAC 2008 systems can be found in [DO08]. Our system performs below average but quite better than the least successful participant.

To further examine the performance of our system in other corpora, we performed summarization using the configuration that performed optimally in the TAC 2008 corpus on the corpora of DUC year 2006. Systems in DUC 2006 were to synthesize from a set of 25 documents a brief, well-organized, fluent answer to a non-trivially expressed declaration of a need for information. This means that the query could not be answered by just stating a name, date, quantity, or

System (TAC 2008 SysID)	*AutoSummENG Score*
Top Peer (43)	0.199136
Last Peer (18)	0.102862
Peer Average (All Peers)	0.1647544 (Std. Dev. 0.0215723)
Proposed System (-)	**0.130345**

Table 10.4: AutoSummENG performance data for TAC 2008. NOTE: The top and last peers are based on the AutoSummENG measure performance of the systems.

System (DUC 2006 SysID)	*AutoSummENG Score*
Baseline (1)	0.143679
Top Peer (23)	0.204977
Last Peer (11)	0.12597
Peer Average (All Peers)	0.1841712 (Std. Dev. 0.0170088)
Proposed System (-)	**0.1783284**

Table 10.5: AutoSummENG performance data for DUC 2006. NOTE: The top and last peers are based on the AutoSummENG measure performance of the systems.

similar singleton. The organizers of DUC 2006, NIST, also developed a simple baseline system that returned all the leading sentences of the 'TEXT' field of the most recent document for each topic, up to 250 words [Dan06].

In Table 10.5 we illustrate the performance of our proposed system on the DUC 2006 corpus. It is shown that the system strongly outperforms the baseline system, and is less that a standard deviation (0.01700883) below the AutoSummENG mean performance (0.1841712) of all the 35 participating systems.

From the comparison between the results on the DUC 2006 and the TAC 2008 task we can conclude that our proposed system performed better in the generic summarization task of DUC 2006 than in the update task of TAC 2008. However, this is judged only by the responsiveness-related AutoSummENG measure which makes identifying the exact defects of the TAC summaries non-trivial and requires further investigation in future work.

Nevertheless, it is very important that the proposed summarization components offered competitive results *without* using machine learning techniques with a rich set of sentence features like sentence position or existence of title words. This indicates the usefulness of n-gram graphs as well as the generality of application of the n-gram graph operators and functions. However, other components need to be added to reach state-of-the-art performance, given the existing means of evaluation. These components should aim to improve the overall coherence of the text and tackle such problems as anaphora resolution.

10.6 Conclusions on Summary Extraction Using N-gram Graphs

We have offered a set of methodologies, based on the language-neutral representation and algorithms of n-gram graphs, aiming to tackle a number of automatic summarization problems:

Salience detection , where we have indicated ways to determine the content of a cluster of documents and judge salience for a given sentence.

Redundancy removal , where two different approaches have been presented following the Cross-Sentence Informational Subsumption (CSIS) [RJB00] and Marginal Relevance [CG98] paradigms.

Query expansion , where a scheme to broaden a given query has been proposed, with a slightly improving effect.

The experimental results presented judged only one commonly used aspect of our systems; namely its responsiveness ability. Based on these results we have seen that combining different methods for the components of the overall summarization method, one can achieve significantly different results. The best results were achieved when using sentence scoring with redundancy removal, where query expansion made no real difference in performance. It is very important that the proposed summarization components offered competitive results through simple application of n-gram graph theory and methodologies.

Part IV

Other Contributions and Discussion

Chapter 11

FABLE: An Architecture Proposal for the Support of the AESOP TAC Task (Automatically Evaluating Summaries Of Peers) and the Evaluation Community

This proposal aims to set the foundations of a unified software architecture and framework for the integration of evaluation methodologies of peer summaries. The framework is based on open protocols, supports easy integration of new evaluation methods and allows for remote execution of methods. It aims to be scalable, secure and system independent. It offers support for the use of an optional set of gold standard summaries, support measurements of correlation between different summarization evaluation results and allows for combination of evaluation techniques into complex new techniques.

11.1 Motivation

It has been shown through a variety of works[CD08, Jon07, Nen06, Sjö07, GKV08] that the domain of evaluation of summaries is a difficult task. Existing methods of automatic evaluation like ROUGE [Lin04], BE (and BEwT-E) [HLZF05, TH08] and AutoSummENG [GKVS08] manage to assign such grades to summary systems that system ranking correlates highly to the ranking offered by the human assigned responsiveness measure used in the Document Understanding Conferences (DUC) and Text Analysis Conference (TAC). However, a number of problems have been pinpointed by the summarization community, including the following:

- The evaluation process is so far mostly a shallow evaluation between a summary and a set of gold standard human summaries.

- Existing automatic evaluation grades do not indicate such things as grammaticality, coherence and other desired textual qualities.
- It is not trivial to provide evaluation results concerning where each existing method performs well and where it does not.
- No method is able to judge the quality of a single summary with certainty. Only the average performance over a set of summaries from each system correlates to human judgement.
- The expression of information need appears to be vaguely defined in summarization tasks, which further makes the variations of the responsiveness measure used in recent Document Understanding Conferences and the Text Analysis Conference not robust. We cannot say for certain whether systems have improved over the years, or whether they have achieved the aim of the yearly tasks, because of the vagueness of information need.

The aforementioned problems should be tackled through a community effort aiming to provide for:

- A unified set of benchmark data, including corpora, information need specifications and baseline results.
- A unified approach for the evaluation of different aspects of a summary text, including all its syntactic, semantic and pragmatic properties.
- A unified set of meta-evaluation tools, *i.e.* evaluation of the evaluation, through statistical methods and practices.

It would also be important to judge each evaluation measure according to its indended application, as this is defined by its inventor. Also, one would like to have other researchers comment on the drawbacks and advantages of every method, so that improvement may occur.

11.2 The Architecture

The proposed architecture and corresponding framework, we will call Framework for Automated Binding to Linguistic Evaluation *FABLE*, is illustrated in Figure 11.1. The architecture comprised four main types of roles and corresponding Application Programming Interfaces (APIs):

The Evaluation Client This is the application or individual that wants to use the evaluation framework to evaluate a summarization system. The client interacts with the evaluation proxy to request the evaluation of a summary, or with the meta-evaluator to request evaluation of a given evaluation method for a given corpus.

The Dataset Provider The dataset provider is the source of the data for the summarization and evaluation tasks. It may contain text corpora for summarization, including queries and gold standard summaries. It provides such functionality that one can look up corpora, which are broken down into topics. The topics contain documents which can be retrieved through a corresponding method.

At this point, it should be determined whether new corpora can be registered remotely, by assigning *e.g.* Uniform Resource Identifiers (URIs) to document records in the dataset provider service. This way the dataset provider may not need to host the texts itself, but rather retrieve the text via its URI.

The Evaluator The evaluator provides the functionality of evaluating a measurable textual quality, optionally given a set of gold standard documents. The documents can be simple references to dataset provider resources, or actual texts.

Each evaluator publishes a list of meta-data (attribute tags), showing the domain of application and the specific requirements. These meta-data may define whether the evaluator requires a gold standard set of documents for the evaluation, the languages the evaluator supports and so forth. [1]

Complex evaluators can be implemented by simply creating an evaluator that calls other evaluators and combines the latter's results in a unique way.

The Evaluator Proxy The evaluator proxy acts as the main link to a set of evaluators and their functionality. It provides a facility to register new evaluators to support easy integration of new methodologies. Furthermore, it contains information about the attributes of registered evaluators, so that an evaluation client can locate evaluators based on desired attributes. The evaluation proxy provides the actual interface for evaluation clients, because a client will request the proxy to execute a specific type of evaluation for a given summary. The proxy may then forward the call to an evaluator and transmit the evaluation result to the client when available.

The evaluator proxy also contains the authentication logic, so that security rules can be applied on the availability of services per client. Overall, the evaluator proxy is the gateway of the evaluation methods to the clients.

The Meta-evaluator The meta-evaluator acts as an evaluator proxy, but also provides functionality for the testing of an evaluation result against a gold standard evaluation result. The meta-evaluation methodologies the meta-evaluator supports (e.g. Pearson, Spearman, Kendall correlation) are published and the client can select any meta-evaluation method for a given set of evaluation results.[2]

11.3 Implementation Issues

In order to fulfil the need for open standards, the proposed implementation framework for FABLE would be the Web Services framework. Web services are

[1]It might also be important to also let community members propose tags to an evaluator according to its performance in different tasks, but this may be a second step after the implementation of FABLE.

[2]Optionally, the meta-evaluation may be divided into proxies and meta-evaluators, but this may increase the complexity of the architecture without an actual need to support.

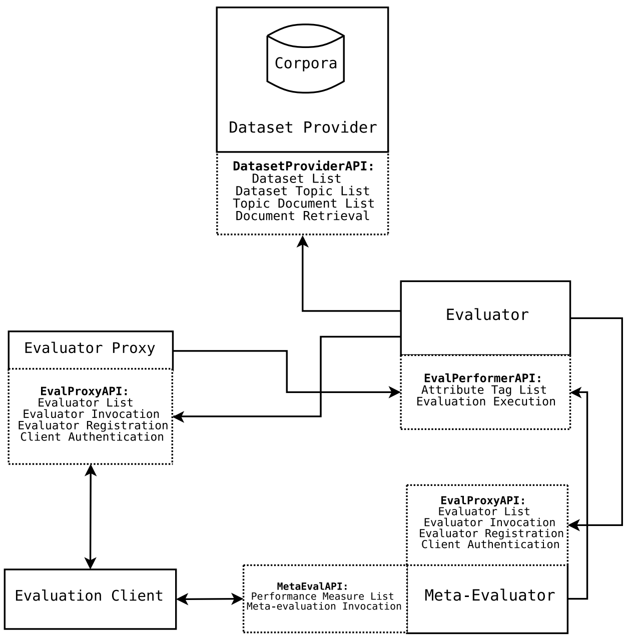


Figure 11.1: The proposed Framework for Automated Binding to Linguistic Evaluation. The arrows point from the caller to the method implementor (API implementor).

system independent, allow for distributed execution and are quite secure when executed via the HTTPS or a similar encrypted protocol. Furthermore, they provide for abstraction and hide implementation details, which can be important when facing non-disclosure agreements or commercial, non-open products.

Web Services' specifications have long been established and can be easily applied to any kind of underlying language. This means that the set of already existing methodologies will only need to provide a wrapper for their execution, in accordance to the specifics of the FABLE APIs. In order to provide for custom options the API should include a 'custom parameters' field. However, evaluation methods should use default values, where applicable, to avoid the necessity of using these custom options.

There is a question of whether the transfer of documents over HTTP will cause too much overhead. However, caching methods can be implemented in evaluators so that a text does not need to be transported from the Dataset Provider every time it is required, especially for highly requested corpora. An alternative would be to use intelligent agents, that can transfer between servers and execute where the corpus resides, but this might pose a set of implementation issues given the fact that we need a whole community to adopt the framework.

11.3.1 Roadmap

The steps required in order to help the community adopt FABLE are the following.

- For each service of the functionality proposed herein, decide on the exact APIs through community discussion.
- Create a primary Dataset Provider to make widely used corpora available.
- Create the Evaluator Proxy service and create wrappers for the basic evaluation methods, providing detailed information on how to register a new evaluator through documentation.
- Provide generic wrappers that can *e.g.* provide results from a command line (shell) execution. This means that any command line-based application will be easily wrapped and supported by FABLE.
- Determine the security policy of the Evaluator Proxy service. How can one register to use the services? Should the registration service be open or not? Should there be an automated mechanism for registration?
- Create a basic Meta-evaluator for the commonly used meta-evaluation methodologies, such as correlation to a given ranking. It is important at this part to determine the format of the gold standard data and whether gold standard evaluation results can reside in the repository of the Dataset Provider service.
- Provide a test case of the framework to the community as a whole to determine problems and drawbacks.
- Finalize system specifications and request its use from community members, supporting them throughout the process, at least through good documentation.

11.4 Overview of FABLE

The proposed architecture, called FABLE, aims to provide a common architecture and set of tools for summary evaluation primarily, and possibly for other purposes. It allows for easy integration of existing evaluation methods, as well as of new methods via a secure, publicly available set of gateways in the form of Evaluator Proxies. It further supports the evaluation of evaluation methods regardless of the underlying methodology. Finally, it provides an open protocol, system-neutral, programming language-neutral testbench for existing evaluation methods, without requiring collocation of application code. Finally, the repository of corpora may prove important for other domains of Natural Language Processing (NLP) as well and alleviate the cost for corpus availability.

Chapter 12

JINSECT Toolkit: A Generic NLP Toolkit using N-Gram Graphs

The JINSECT open source (LGPL) toolkit is a set of applications, programming structures and algorithm implementations that allow for the use of n-gram graphs for a series of NLP tasks. The generic methods and operators concerning the graphs have been optimized and tested over a variety of settings and applications with very promising results. The toolkit can be used both as a library for the development of novel NLP applications or as a minimal suite of applications, including a summarizer and a summary evaluation application. Within this chapter we briefly present its current and potential applications and scope.

Within the past few years, research in the Natural Language Processing (NLP) domain has led to a multitude of novel methods for the analysis of text. Ongoing work in the fields of topic models, machine learning as well as the application of graph representations to represent textual information tend to augment the set of tools available to the NLP community.

In view of the increasing tendency to use statistical methods within NLP tasks, as well as the usefulness and generic utility of such methods regardless of language, we have designed and implemented a toolkit for NLP to help the adoption of n-gram graphs within NLP tasks. The toolkit has a full set of programming structures, algorithms and method implementations to help such tasks as the ones shown in Table 12.1. The described toolkit is more an object oriented library than a fully-fledged NLP program suite. There is much ongoing work and a set of applications that we plan to support using the toolkit.

12.1 Purpose of the Toolkit

The JINSECT acronym stands for Java INteroperable Semantic Extraction Context-based Toolkit. The main targets of the toolkit are:

- Use of contextual information.

Text Classification
Authorship Identification
Text Indexing
Semantic Annotation
Automatic Summarization
Automatic Evaluation of Summary Systems
Opinion Extraction
Text Stemmatology

Table 12.1: Applications of JINSECT

- The extraction of high level information using statistical, language-neutral methods.
- Interoperability with other tools and resources.

The main contribution of the toolkit concerns the implementation of programming structures and algorithms adopting the n-gram graph representation method for analysing texts. Summarization subtasks like query expansion, sentence selection and summary evaluation have already been performed using the JINSECT toolkit [GKV08, GKVS08]. Ongoing work concerning a variety of applications (see Figure 12.1) indicates the generic applicability of the toolkit in different settings.

12.1.1 Requirements

The toolkit is platform-independent due to the use of Java language as the implementation language. It requires Java v1.6 or newer to perform without problems (even though v1.5 may be fine, as well). The licence used in JINSECT is LGPL, allowing for any kind of commercial or non-commercial use[1].

12.2 JINSECT as a library

Within JINSECT an application developer can find structures and algorithms concerning the extraction and representation of texts – and sequences of characters in general – as one of the following alternatives:

- character n-gram graphs
- character n-gram histograms
- word n-gram graphs
- word n-gram histograms

N-gram graphs, the implementation of which has been based the OpenJGraph library[2], capture co-occurrence information retaining neighbour-of-a-neighbour information, unlike feature vectors. This type of representation is

[1]Libraries coming from other sources, used in example applications in the toolkit, may of course hold their own licence. However, almost all dependencies are on open source projects. This holds certainly for all basic structures and algorithms.

[2]See also `http://sourceforge.net/projects/openjgraph/`

the most promising part of the library. However, the useful information is held with a cost in analysis time and memory usage. Therefore, a really important set of time and memory optimizations have been implemented within the JINSECT library to make sure that the algorithms perform well in most circumstances.

The library supports a set of operations upon and between graphs, as these have been defined by Giannakopoulos et al. in [GKV08], namely:

- Similarity *sim*
- Merging or Union $\cup$
- Intersection $\cap$
- Delta Operator (*All-Not-In* operator) $\triangle$
- Inverse Intersection Operator $\triangledown$

Most of these operators' implementations have been optimized for maximum performance, using caching methods and efficient algorithms for speed improvement in everyday applications.

The ability to use n-grams of variable granularity, applying character n-gram and word n-gram analysis of various ranks, offers both maneuverability and simplicity of use. This helps attain both language-neutrality, but also the extraction of higher level features in a purely statistical manner.

As additional help, we have implemented:

for clustering and indexing , a set of clustering and indexing algorithms, using n-gram graph as an alternative (ongoing work).

for statistical calculations a set of structures which can be used as distributions or histograms and corresponding algorithms calculating mean, standard deviation, *argmax* and other similar functions.

for serialization a set of abstraction classes for the serialization of almost all objects that appear in the toolkit, which allows for caching and transferability of intermediate results. The abstraction provides unified access to memory, files, compressed files and any other repository.

for summary evaluation the AutoSummENG method (see section 12.3.2), which allows both single-machine multi-threaded execution and use by agents (through JADE [BPR99]), as a case study of distributed execution.

for interoperability modules in JINSECT allow for the use of WordNet [MBF+90] as well as other dictionaries and thesauri which have been exposed via Web Services[3]. Also, we support different kinds of corpora through abstraction classes for both documents and corpus objects.

other utility functions for generic Java tasks (efficient file reading, multi-threading, etc.).

[3]See `http://services.aonaware.com/DictService/DictService.asmx`.

Emirates is the the biggest customer of the A380 with an order for 43 planes, and has been expecting to take delivery of the aircraft in October 2006. An Airbus spokesman in France said the company has begun to measure turbulence in the wake of the A380 but that studies are not complete. Singapore Airlines will be the first to receive the new model, with the first orders delivered in late 2006, following earlier production delays. Construction problems have delayed the introduction of the double-deck A380, the largest passenger plane in the world.

Figure 12.1: A sample summary from the TAC 2008 corpus – Topic 801A-A

Within the toolkit we have included and used, for example, methods for the annotation of sentences through the combination of statistical methodology and background, thesaurus-based knowledge. In the same domain, we have created a statistical chunker based on the method described in section 10.1.1. These code snippets offer excellent samples of ongoing, not yet published, research concerning the use of n-gram graphs as part of various NLP tasks.

JINSECT has a large part of its classes documented as JavaDoc documentation, even though documentation is a constantly ongoing endeavour. Within the toolkit source code one can find a whole set of usage samples from spam filtering usages to summary evaluation.

12.3 JINSECT as an Application Suite

JINSECT contains two main applications:

- A Text Analysis Conference (TAC) corpus compatible update summarizer that can be called via the command-line.
- A Summary System Evaluation application using n-gram graphs, with a working specialized GUI.

12.3.1 Summarizer

The summarizer (see a sample summary in Figure 12.1) implemented in the toolkit offers both a proof-of-concept for the generic applicability of n-gram graphs in NLP subtasks, as well as an excellent example for the use of the library. The summarizer at this moment is a console-only application, but soon we plan to create a user interface for its use.

The summarizer is under heavy development and has offered really promising results so far, especially after the integration of a third-party sentence splitting tool by the National University of Singapore[4].

12.3.2 Summary System Evaluation

The AutoSummENG methodology, which has been implemented within JINSECT, evaluates a set of summarizing systems, or 'peers', with respect to a given set of model summaries. In order to perform this kind of evaluation, the

[4]see `http://www.comp.nus.edu.sg/~qiul/NLPTools/JavaRAP.html`.

system compares an n-gram graph representation of each peer summary to the n-gram graph representations of the model summaries, grading each system by a set of similarity indications: one similarity indication for each model summary on the same topic.

The AutoSummENG system can function on either n-grams of *words* or *characters*, each giving its own results upon application. It has been shown by means of analysis, that if one uses the toolbox to evaluate how responsive the summaries of a given system are, it is preferable to use character n-grams.

The graph of n-grams (whether words or characters) are created by having each n-gram represented by a node in the graph, and add edges between neighbouring n-grams. N-grams are considered neighbours if they fall within a number of words (or characters correspondingly) of each other. The toolkit also takes into account n-grams of different sizes, which can be set manually. Therefore, the parameters the user can test are:

- Whether character n-grams or word n-grams are to be used.
- The minimum size of n-grams to take into account.
- The maximum size of n-grams to take into account.
- The distance within which two n-grams are to be considered neighbours.
- Whether for the Value grade, the *number of co-occurrences* of two n-grams may be taken into account, or the *average distance* between the neighbouring n-grams in these occurrences.

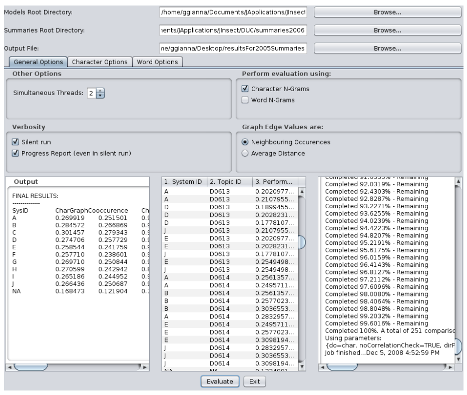
The grades (values) returned by the toolkit are supposed to provide a *ranking* similar to that humans would decide on if they evaluated the same set of peers over the same topics. That ranking, should be expected to be correct, given a sufficient number of topics to evaluate on.

There are different kind of grades returned for a single comparison:

- The first refers to overlap, indicating how much the graph representation of a model summary overlaps a given peer summary. This is called the *Co-occurrence* measure.
- The second also refers to overlap, also taking into account how many times two n-grams are found to be neighbours. This is called the *Value* measure, which expects two similar texts to have n-grams neighbouring about the same number of times.
- The third is the *Size* measure, simply indicating the ratio of n-grams between the smaller and larger summary (whether model or peer)[5].
- The final grade is the *Overall* grade, which is a weighted sum of the previous measures and is in a planning stage.

The JINSECT toolkit also implements and supplies a method for determining optimal n-gram parameters for a given corpus, as indicated in [GKV08]. The implementation of the AutoSummENG method allows for the use of multithreading for higher performance.

[5]This grade should indicate a baseline for correlation to human grades, and should not be taken into account otherwise.



Models Root Directory: /home/ggianna/Documents/JApplications/JInsect
Summaries Root Directory: ıents/JApplications/JInsect/DUC/summaries2006
Output File: ne/ggianna/Desktop/resultsFor2005Summaries

Other Options
Simultaneous Threads: 2

Perform evaluation using:
[✓] Character N-Grams
[ ] Word N-Grams

Verbosity
[✓] Silent run
[✓] Progress Report (even in silent run)

Graph Edge Values are:
(●) Neighbouring Occurences
( ) Average Distance

Output

```
FINAL RESULTS:
--------------
SysID     CharGraphCooccurence     Ch
A         0.269919     0.251501    0.9
B         0.284572     0.266869    0.9
C         0.301457     0.279343    0.9
D         0.274706     0.257729    0.9
E         0.258544     0.241759    0.9
F         0.257710     0.238601    0.9
G         0.269710     0.250844    0.9
H         0.270599     0.242942    0.8
I         0.265186     0.244952    0.9
J         0.266436     0.250687    0.9
NA        0.168473     0.121904    0.7
```

1. System ID	2. Topic ID	3. Perform...
A	D0613	0.2020977...
A	D0613	0.2107955...
D	D0613	0.1899455...
D	D0613	0.2028231...
D	D0613	0.1778107...
J	D0613	0.2107955...
E	D0613	0.2020977...
E	D0613	0.2028231...
J	D0613	0.1778107...
E	D0613	0.2549498...
J	D0613	0.2549498...
A	D0614	0.2561357...
A	D0614	0.2495711...
B	D0614	0.2561357...
B	D0614	0.2577023...
B	D0614	0.3036553...
A	D0614	0.2832957...
E	D0614	0.2495711...
E	D0614	0.2577023...
E	D0614	0.3098194...
J	D0614	0.2832957...
J	D0614	0.3036553...
J	D0614	0.3098194...

```
Completed 92.0319% - Remaining
Completed 92.4303% - Remaining
Completed 92.8287% - Remaining
Completed 93.2271% - Remaining
Completed 93.6255% - Remaining
Completed 94.0239% - Remaining
Completed 94.4223% - Remaining
Completed 94.8207% - Remaining
Completed 95.2191% - Remaining
Completed 95.6175% - Remaining
Completed 96.0159% - Remaining
Completed 96.4143% - Remaining
Completed 96.8127% - Remaining
Completed 97.2112% - Remaining
Completed 97.6096% - Remaining
Completed 98.0080% - Remaining
Completed 98.4064% - Remaining
Completed 98.8048% - Remaining
Completed 99.2032% - Remaining
Completed 99.6016% - Remaining
Completed 100%. A total of 251 compariso
Using parameters:
{do=char, noCorrelationCheck=TRUE, dirF
Job finished...Dec 5, 2008 4:52:59 PM
```

Figure 12.2: A snapshot from the main AutoSummENG window

12.3.3 N-gram Graphs in Spam Filtering

As proof of our intuition that graphs can be used in a classification task we applied our n-gram graph methodology on a spam filtering task using JInsect. The task is the one described in CEAS 2008[6]. The task was to classify about 140000 e-mails as spam or ham, but in the process of classification feedback was given to the classifier, much like the case where a user provides feedback on what is considered to be spam.

Given this scenario we created a spam filter server[7] and tried a variety of approaches that used n-gram graphs as the means to represent the e-mails and their classes, to compare new e-mails to the classes and to update the class representations. In our preliminary experiments the most promising approach, which used a maximum of 20K characters from the e-mails, with *no preprocessing* at all, performed really well as can be seen in Table 12.2. The evaluation measures for the filter combine the percentage of spam blocked and the filter's false positive rate. They are the *Logistic Average Misclassification (LAM)* and the *Area under the ROC curve (ROCA)* measures. According to the CEAS organizers the LAM calculation is 'smoothed':

$$lam = invlogit((logit(FPrate) + logit(FNrate))/2) \tag{12.1}$$

where $logit(p) = log(\frac{p}{1-p})$, $invlogit(x) = \frac{e^x}{1+e^x}$ and $FPrate = (\#\text{ham-errors} + 0.5)/(\#\text{ham} + 0.5)$, $FNrate = (\#\text{spam-errors} + 0.5)/(\#\text{spam} + 0.5)$.

The diagrams of the performance of the system over the number of classified e-mails, as well as the performance for various thresholds concerning the strictness of the filter are correspondingly Figure 12.3 and Figure 12.4. The strictness of the filter indicates the trade-off between ham misclassification and spam misclassification.

The aforementioned experiments, together with a first place in the primary ranking concerning a stemmatology challenge (Computer Assisted Stemmatology Challenge[8]), where the n-gram graphs were used as a clustering methodology to determine the derivation of different versions of a single text, provided

[6]See `http://www.ceas.cc/2008/challenge/` for more on the challenge.
[7]The full source can be found in the JINSECT toolkit (see chapter 12).
[8]See `http://www.cs.helsinki.fi/u/ttonteri/casc/results.html` for more information.

```
              Gold Standard
        +--------------------+
        |       ham    spam  |
        |                    |
Filter  | ham  24900   1777  |
Result  |spam   2229 108799  |
        +--------------------+
Total          27129 110576

ham%:     8.22 (7.89-8.55)
spam%:    1.61 (1.53-1.68)
lam%      3.68 (3.58 - 3.79)
```

Table 12.2: Results of the application of n-gram graph in the CEAS 2008 task.

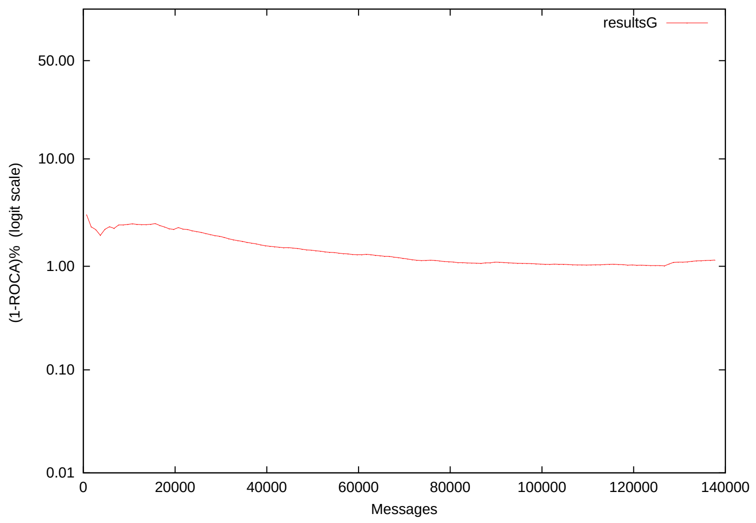


Figure 12.3: The performance of the n-gram graph-based spam filter over the number of classified e-mails.

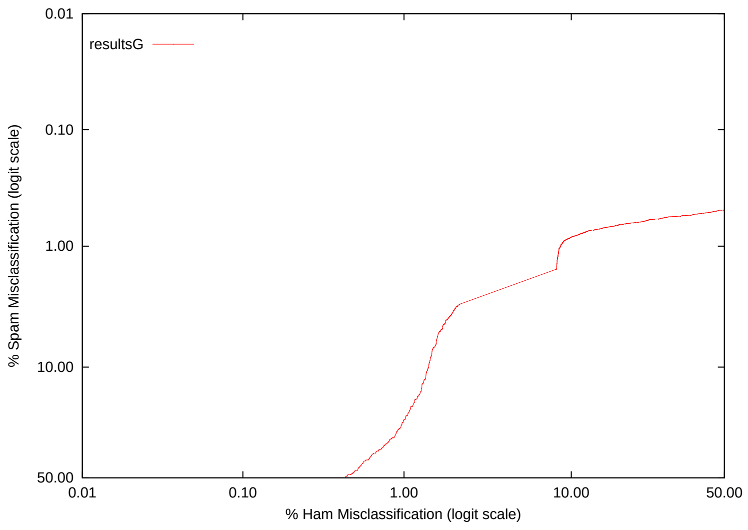


Figure 12.4: The performance of the n-gram graph-based spam filter with respect to strictness.

the support we needed for the use of n-gram graphs in a variety of applications. Since the full report of all side-experiments and results would be out of scope here, we plan to publish the multitude of applications where n-gram graphs can be found to be useful in the future.

12.4 Overview of JINSECT and Future Work

The JINSECT toolkit is a set of applications, programming structures and algorithm implementations that allow for the use of n-gram graphs for a series of NLP tasks. The generic methods and operators concerning the graphs have been optimized and tested over a variety of settings and applications. The project has been partially documented, even though there is much more to be done. The licence of the toolkit allows the latter's use as a royalty-free library.

The toolkit, which can be found at `http://www.ontosum.org/?q=static/AutomaticSummarization`, is under constant development and soon we plan to have an overall estimation of the next development steps. Specifically, much more work has to be done concerning the documentation and its update. Furthermore, the breaking up of the library in self-sufficient sub-parts may offer further modularity and make adoption even easier for prospective application developers. We also plan to have a larger set of sample, full applications using the toolkit to test the n-gram graphs applicability in real-life settings.

The presented toolkit offers both intuition through examples as well as verification potential for the use and usefulness of n-gram graphs and their related methods. Additionally, it offers a framework for the application of these graphs which is extensible, open and evolving. Future work should aim at further taking into account development requirements by the developping community itself, to provide a fully usable toolset for the creation of modern NLP applications.

Chapter 13

Overall Discussion and Future Work

The domain of multi-document summarization offers a multitude of challenges. As a complex human process, summarization combines individual lower-level functions into a highly powerful, but intensely subjective cognitive tool. Even though much work has been done to model the summarizing process and its evaluation according to human summarizers' methods [EN00, VHT03, SOC^{+}02], more must be done in order to achieve the following targets:

- Create an intermediate representation for the source information efficiently, without inserting much noise in the extraction process.
- Combine the intermediate representation with common knowledge, *e.g.* through logic and reasoning, to deduce further useful information for the given use and purpose of the summary.
- Generate fluent, coherent text achieving the satisfaction of communication, taking into account user preferences and knowledge, in terms of a user model.
- Quantify and measure automatically different aspects of textual quality.
- Determine methodologies that will function regardless of the underlying language.

However, the research on summarization in its more than 50-year-old path has led to a variety of very useful applications. Furthermore, both positive and negative results, strengths and weaknesses have helped increase the community intuition on what summarization is about. Valuable works, ranging from formalisms on discourse structure and content salience, to overviews of what summarization systems have achieved, have shown that the domains of Natural Language Processing and Knowledge Representation are intertwined in a way that offers incentives on interdisciplinary collaboration.

The use of Machine Learning in the process of summarization together with the integration of background knowledge, in the form of thesauri (*e.g.* WordNet) or semi-structured text (*e.g.* Wikipedia), also indicate the fact that the summarization process is and should be viewed as a number of individual subprocesses. Such subprocesses need to be determined and judged individually.

Furthermore, knowledge-rich summarization approaches need to take into account user preferences and human subjectivity when determining their output, probably by using a user model and, ideally, pragmatic knowledge.

13.1 Expression of Information Needs and Human Interface

Ontologies, as a means to describe information needs (see information need model in section 2.2), will probably allow to perform user-driven (actually information-need driven) summarization based on reasoning processes, where the desired pieces of information will be required to belong to the classes of information expressed in the information need model.

To put it in simpler words, if someone is interested in the cause of an event, he may construct a model that describes the *restrictions* for the required information, *i.e.* the set of restrictions over data extracted from the original text. The aim is to support filtering out anything but causes. We will call the set of information compatible with these restrictions a *facet* of the original information, and the actual restrictions a *facet restriction set.* It should be noted that the facet and its restriction set are defined by the *point-of-view* (POV) of the consumer, *i.e.* the position the consumer holds relative to a subject.

Thus, one may be able to construct a facet restriction set using ontologies, according to one's information need model, and use reasoning processes to extract the subset of information required to feed the summarizing process. This could be by itself a salience selection method.

Currently, systems do not use advanced user-interfaces to gather user needs to facilitate the summarization process: this limits the expressivity of information needs to a boolean query or similar representation. Existing approaches do not exploit the individual aspects of a summarization process, where a number of qualities define the overall quality of the result. It would be interesting to see how one can lead the user to define his information needs in a complete and well-defined fashion using qualitative criteria as well as informational ones. Thus, a user should be able to ask for a *strongly cohesive, brief summary, well-based on existing research* concerning *summarization*, or a *low-intertextuality, low-redundancy but highly informative summary* concerning *comments on a movie.*

Expressing information needs is mostly an interface problem. We would like to be able to express queries such as the following:'What happened with Iran today?', or even worse 'What's new about Iran?' (see also section 2 of [EN00] for an indication of problematic interfaces). The pragmatic context (*i.e.* world knowledge) of a fellow speaker would probably link the question to the temporal context of today (*e.g.* 29/6/2006) or the last few days. Then the same kind of knowledge, together with a salience selection process about facts for Iran, would lead to international politics concerning my query. Finally, the recall of our last talk, one day before, would lead to a redundancy removal process, leaving only new information at our disposal.

Current interfaces do not facilitate expressing user needs and preferences in a detailed way. In the human-to-human interface, the underlying pragmatics aid the communication process immensely, even in a very brief query. Of course the requester of information knows what to expect of his fellow speaker, *i.e.* knows the common underlying pragmatics. Should we, then, try to approach the summarization process in a more dialectic pattern? Should we let the user define the common knowledge between him and the system or try to deduce it? These matters need to be studied to allow for better human-machine interaction.

13.2 Textual Qualities

Intentionality, *i.e.* the ability of a text to convey the intention of the writer, can play a vital role in summarization. It is rather self-evident that if one omits the intention of the writer in the sentence 'I expect you believe I do not have better things to do' by reducing the sentence to 'I do not have better things to do', one has lost a vital part of the semantics. A second example would be the sentences: 'The troops believe they are holding up democracy. Is that serious thinking?'. The main point of the sentence is that troops do not follow the correct train of thought, and not that they are holding democracy. Thus, a reductive transformation would have to take the intention of communication into account, which would give a very different result. The questions arising are:

- How can one model communication intention?
- How can one locate and capture these communicative semantics?
- How can one use it in the summarization process?

For the communication intention and communicative semantics perhaps the summarization community should consult, among other, the agent community, as well as psycholinguistics.

Anaphora resolution, on the other hand, seems to play a vital role in the semantic representation of a text. If a system fails to connect an attribute to its rightful owner, it will also fail to represent the provided information correctly. Many systems appear to have reduced effectiveness due to this exact problem, so more effort should be put towards this. Summarization research seems to have already focused on the alleviation of anaphora resolution problems (*e.g.* [CIK$^+$06, WKB06]) and the problem has been indicated in various papers (see [LOSG06]).

The vision of a human-oriented, pragmatics-rich summarization system, achieving high quality on output evaluation by humans, as well as featuring a set of flexible interface tools seems quite difficult to complete at this point in time. But breakthroughs can often occur on the basis of a different perspective. That change of perspective, as well as the combination of inter-disciplinary knowledge may offer the momentum required to achieve excellence. This is what we think that the vivid summarization community should strive for.

13.3 Language and Application Neutrality

The notion of language-neutrality is somewhat neglected in the basis of Natural Language Processing tasks, in that we often make basic assumptions concerning the underlying grammar, the syntax and even the direction of writing. The n-gram graph representation can help remove some of these assumptions and determine how semantic features that are common over all languages can be extracted and represented in a common fashion.

We consider that the implications originating from the use of n-gram graphs will have a serious impact to how the language will be treated in future research. The sense of proximity, either surface or conceptual, appears to be quite a powerful feature for a variety of tasks. Higher order relations can be extracted by mining on the n-gram graph structure, using such tools as cliques or even generic models on the evolution of text graphs over space or time. The models that function upon bag-of-words may offer richer information when applied to proximity-based representations, that offer information on sequence.

To determine whether language neutrality can be maintained within various tasks, we are going to present in the immediate future a variety of tests involving the use of n-gram graphs and their corresponding algorithms in such tasks as text classification, authorship identification, text stemmatology and sentiment analysis.

Concerning the neutrality of the n-gram graphs in the application level, we will try to prove via a set of experiments on image and video applications, such as behaviour recognition and optical character recognition. If these experiments offer the results we expect, then a whole range of new tools based on the n-gram graph notion, will be available to the research community, providing the existing scientific arsenal with more than a new set of tools: it will offer a new perspective of data and information, based on neighbourhood and proximity.

13.4 Improvements over Presented Research

13.4.1 AutoSummENG

As far as AutoSummENG is concerned, it would be interesting to use only symbols in our analysis of the summary texts, to see if there is hidden information in what we have called 'non-symbols' or not. On the same basis, a parameter-free version of the method would determine n-grams of various length and synthesize a single n-gram graph, not requiring any input but the corpus of summaries. It would also be interesting to investigate different types of neighbourhood and different functions of importance for neighbourhood, as well as different weighting functions for the importance of matching n-grams of specific rank (e.g. longer vs. shorter n-grams) or nature (e.g. rare vs. common n-grams). We are planning to investigate the evaluation of other characteristics, like grammaticality, cohesion and so forth, using the same scheme with different parameters. In this investigation, the word n-gram dimension should be re-examined, because it may provide more noise-free information, considering the fact that whole words usually follow our definition of *symbols* by being meaningful.

13.4.2 Query Expansion

Within the redundancy removal method presented in section 10.4.2 we only use morphology, through the n-gram graphs' representation of sentences, to determine overlap. However, it would be interesting to determine whether redundancy based on the concepts apparent in a sentence would prove more fruitful. An alternative we plan to apply in the future is that of using the semantic index (see section 10.1.2) to annotate a sentence with its ‘meaning’ and then perform the comparison between ‘meanings’, as described in 10.1.2.

13.4.3 Summarization System

It is our aim for the future to determine the effect of each individual component of our presented summarization systems better, by implementing methodologies and experiments that may judge individual components. Furthermore, we plan to research new evaluation methods that will be able to measure other textual qualities (also see part I)

Chapter 14

Summary of Presented Research

Within this work, we have provided a set of tools and methodologies usable both in the domain of document summarization, as well as other NLP tasks. We have illustrated how the proposed methodologies can be adapted to tackle such problems as sentence salience detection, redundancy removal and novelty detection in summaries, summary system evaluation and the quantification of textual qualities. We believe that this research offers the basis for the evolution of current NLP algorithms through the use of n-gram graphs. We further hope to have provided a new view for the modeling of textual representation while retaining language-neutrality and usefulness, aiming at generic NLP usability algorithms, operators and functions.

Recapitulating, this work has made the following basic contributions.

- A statistically extracted, language neutral, generic usability representation — namely n-gram graphs — that offers richer information than the feature vector. The representation is accompanied by a set of theoretical and practical tools for the application of the n-gram graph representation and algorithms in NLP tasks. (Part I)
- An automatic evaluation system, aiming to capture the textual quality of given summaries in a language-neutral way, by using the n-gram graph representation. (Part II) The evaluation system we call AutoSummENG has achieved state-of-the-art performance while maintaining language neutrality and simplicity.
- The Symbol Sequence Statistical Normality measure, as a quality indicative feature of text, based on the statistics of character sequences within a given text.
- An automatic summarization system based on the use of n-gram graphs, focusing on the tackling of content selection and redundancy removal in a language-neutral manner. (Part III) The proposed variations of our summarization system offered competitive results on the TAC 2008 corpus, without using complex features and machine learning techniques to optimize the performance.

Within the presented research we dedicated some time to help promote the collaboration between summarization community researchers. This time gave birth to:

- The FABLE framework, aiming to support the AESOP (Automatically Evaluating Summaries Of Peers) task of the Text Analysis Conference upcoming in 2009, by providing a common framework for the integration and evaluation of summary evaluation techniques.
- The JINSECT toolkit, which is a Java-based toolkit and library that supports and demonstrates the use of n-gram graphs within a whole range of Natural Language Processing applications, ranging from summarization and summary evaluation to text classification and indexing. The toolkit is a contribution to the NLP community, under the LGPL licence that allows free use in both commercial and non-commercial environments.

Bibliography

[ACD$^+$98] J. Allan, J. Carbonell, G. Doddington, J. Yamron, and Y. Yang. Topic detection and tracking pilot study: Final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, volume 1998, 1998.

[ADKK04] S. D. Afantenos, I. Doura, E. Kapellou, and V. Karkaletsis. Exploiting cross-document relations for multi-document evolving summarization. *Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science) 3025*, pages 410–419, 2004.

[AGK01] J. Allan, R. Gupta, and V. Khandelwal. Temporal summaries of new topics. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 10–18. ACM Press New York, NY, USA, 2001.

[AGPV05] E. Amigo, J. Gonzalo, A. Penas, and F. Verdejo. Qarla: a framework for the evaluation of automatic sumarization. *Proceedings of the 43th Annual Meeting of the Association for Computational Linguistics*, 2005.

[AKS05a] S. D. Afantenos, V. Karkaletsis, and P. Stamatopoulos. Summarization from medical documents: A survey. *Artificial Intelligence in Medicine*, 33:157–177, 2 2005. Issue 2.

[AKS05b] S. D. Afantenos, V. Karkaletsis, and P. Stamatopoulos. Summarizing reports on evolving events; part i: Linear evolution. 2005.

[All87] J. Allen. *Natural Language Understanding.* Benjamin/Cummings Pub. Co Menlo Park, Calif, 1987.

[Ami01] E. Amitay. *What Lays in the Layout.* PhD thesis, 2001.

[AWB03] J. Allan, C. Wade, and A. Bolivar. Retrieval and novelty detection at the sentence level. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 314–321. ACM New York, NY, USA, 2003.

[BDH$^+$00] B. Baldwin, R. Donaway, E. Hovy, E. Liddy, I. Mani, D. Marcu, K. McKeown, V. Mittal, M. Moens, D. Radev, and Others. An evaluation roadmap for summarization research. Technical report, 2000.

[Bel09] Anja Belz. That's nice... what can you do with it? *Comput. Linguist.*, 35(1):111–118, 2009.

[BEM02] R. Barzilay, N. Elhadad, and K. McKeown. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 17:35–55, 2002.

[BHR06] P. Blache, B. Hemforth, and S. Rauzy. Acceptability prediction by means of grammaticality quantification. *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the ACL*, pages 57–64, 2006.

[BKM90] J. Barnett, K. Knight, and I. Mani. Knowledge and natural language processing. *Communications of the ACM*, 33(8):50–71, 1990.

[BL97] T. Berners-Lee. Metadata architecture. Technical report, 1997.

[BL04] R. Barzilay and L. Lee. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *Proceedings of HLT-NAACL 2004*, pages 113–120, 2004.

[BLL98] C. Burgess, K. Livesay, and K. Lund. Explorations in Context Space: Words, Sentences, Discourse. *Discourse Processes*, 1998.

[BM05] R. Barzilay and K. R. McKeown. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–327, 2005.

[BME99] R. Barzilay, K.R. McKeown, and M. Elhadad. Information fusion in the context of multi-document summarization. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 550–557. Association for Computational Linguistics Morristown, NJ, USA, 1999.

[BNJ03] D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022, 2003.

[BP98] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998.

[BPR99] F. Bellifemine, A. Poggi, and G. Rimassa. JADE–A FIPA-compliant agent framework. In *Proceedings of PAAM*, volume 99, pages 97–108, 1999.

[Bun98] H. Bunke. Error-tolerant graph matching: a formal framework and algorithms. *Advances in Pattern Recognition, LNCS*, 1451:1–14, 1998.

[BV04] Michele Banko and Lucy Vanderwende. Using n-grams to understand the nature of summaries. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Short Papers*, pages 1–4, Boston, Massachusetts, USA, May 2004. Association for Computational Linguistics.

[CD08] John M. Conroy and Hoa Trang Dang. Mind the gap: Dangers of divorcing evaluations of summary content from linguistic quality. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 145–152, Manchester, UK, August 2008. Coling 2008 Organizing Committee.

[CG98] J. Carbonell and J. Goldstein. Use of mmr, diversity-based reranking for reordering documents and producing summaries, the. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 335–336. ACM Press New York, NY, USA, 1998.

[CH01] N. Chenowith and J. R. Hayes. Fluency in writing: Generating text in l1 and l2. *Written Communication*, 18(1):80, 2001.

[Cho55] N. Chomsky. Grammaticality in the logical structure of linguistic theory, 1955.

[Cho05] N. Chomsky. *Rules And Representations*. Columbia University Press, 2005.

[CIK$^+$06] T. Copeck, D. Inkpen, A. Kazantseva, A. Kennedy, K. Darren, V. Nastase, and S. Szpakowicz. Leveraging duc. 2006.

[CL01] C. C. Chang and C. J. Lin. Libsvm: a library for support vector machines. *Software Available at Http://www. Csie. Ntu. Edu. Tw/cjlin/libsvm*, 80:604–611, 2001.

[Cle81] W. S. Cleveland. Lowess: A program for smoothing scatterplots by robust locally weighted regression. *The American Statistician*, 35(1):54–54, 1981.

[CNP06] G. Carenini, R. T. Ng, and A. Pauls. Interactive multimedia summaries of evaluative text. In *Proceedings of the 11th International Conference on Intelligent User Interfaces*, pages 124–131. ACM Press New York, NY, USA, 2006.

[CS04] T. Copeck and S. Szpakowicz. Vocabulary usage in newswire summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 19–26. Association for Computational Linguistics, 2004.

[CSO07] J.M. Conroy, J.D. Schlesinger, and D.P. O'Leary. CLASSY 2007 at DUC 2007. In *Proceedings of Document Understanding Conference (DUC) Workshop 2006*, 2007.

[CSS05] J. M. Conroy, J. D. Schlesinger, and J. G. Stewart. Classy query-based multi-document summarization. In *Proceedings of the Document Understanding Conf. Wksp. 2005 (DUC 2005) at the Human Language Technology Conf./Conf. on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, 2005.

[Dan05] H. T. Dang. Overview of DUC 2005. In *Proceedings of the Document Understanding Conf. Wksp. 2005 (DUC 2005) at the Human Language Technology Conf./Conf. on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, 2005.

[Dan06] H. T. Dang. Overview of DUC 2006. In *Proceedings of HLT-NAACL 2006*, 2006.

[DDF$^+$90] S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[DHS01] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley New York, 2001.

[DIM05] H. Daume III and D. Marcu. Bayesian summarization at duc and a suggestion for extrinsic evaluation. In *Proceedings of the Document Understanding Conf. Wksp. 2005 (DUC 2005) at the Human Language Technology Conf./Conf. on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, 2005.

[DNS$^+$05] W. Doran, E. Newman, N. Stokes, J. Dunnion, and J. Carthy. Iirg-ucd at duc 2005. In *Proceedings of the Document Understanding Conf. Wksp. 2005 (DUC 2005) at the Human Language Technology Conf./Conf. on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, 2005.

[DO08] H. T. Dang and K. Owczarzak. Overview of the TAC 2008 update summarization task. In *TAC 2008 Workshop - Notebook papers and results*, pages 10–23, Maryland MD, USA, November 2008.

[DRA03] N. Daniel, D. Radev, and T. Allison. Sub-event based multi-document summarization. *Proceedings of the HLT-NAACL 03 on Text Summarization Workshop-Volume 5*, pages 9–16, 2003.

[DSDGL99] J. F. Da Silva, G. Dias, S. Guilloré, and J. G. P. Lopes. Using localmaxs algorithm for the extraction of contiguous and non-contiguous multiword lexical units. *Proceedings of the 9th Portuguese Conference on Artificial Intelligence: Progress in Artificial Intelligence*, pages 113–132, 1999.

[Edm69] HP Edmundson. New methods in automatic extracting. *Journal of the ACM (JACM)*, 16(2):264–285, 1969.

[ELH$^+$03] B. Erol, D.S. Lee, J. Hull, R.C.R. Center, and CA Menlo Park. Multimodal summarization of meeting recordings. In *Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on*, volume 3, 2003.

[EN00] Brigitte Endres-Niggemeyer. Human-style www summarization, 2000.

[ENW04] Brigitte Endres-Niggemeyer and E. Wansorra. Making cognitive summarization agents work in a real-world domain. *NLUCS Workshop. Porto, Portugal, April*, pages 86–96, 2004.

[ER04a] G. Erkan and D. R. Radev. Lexpagerank: Prestige in multi-document text summarization. In *Proceedings of EMNLP*, pages 365–371, 2004.

[ER04b] G. Erkan and D. R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479, 2004.

[ER04c] G. Erkan and D. R. Radev. Michigan at duc 2004 – using sentence prestige for document summarization. *Proceedings of the Document Understanding Conferences Boston, MA*, 2004.

[ET93] B. Efron and R. Tibshirani. *An Introduction to the Bootstrap.* Chapman & Hall/CRC, 1993.

[FGFdC08] Jorge García Flores, Laurent Gillard, Olivier Ferret, and Gaël de Chandelar. Bag of senses versus bag of words: comparing semantic and lexical approaches on sentence extraction. In *TAC 2008 Workshop - Notebook papers and results*, pages 158–167, Maryland MD, USA, November 2008.

[FH03] E. Filatova and V. Hatzivassiloglou. Domain-independent detection, extraction, and labeling of atomic events. In *Proceedings of the RANLP Conference*, pages 145–152, 2003.

[FL03] A. Farzindar and G. Lapalme. Using background information for multi-document summarization and summaries in response to a question. In *Proceedings of DUC03: NAACL 2003 Workshop in Automatic Text Summarization, Edmonton, Alberta, Canada*, May 2003.

[FRTF] M. Fuentes, H. Rodriguez, J. Turmo, and D. Ferres. Femsum at duc 2006: Semantic-based approach integrated in a flexible eclectic multitask summarizer architecture.

[GKV06] G. Giannakopoulos, V. Karkaletsis, and G. Vouros. Automatic multi-document summarization and prior knowledge: Past, present and vision (demo-2006-2). Technical report, 2006.

[GKV08] G. Giannakopoulos, V. Karkaletsis, and G. Vouros. Testing the use of n-gram graphs in summarization sub-tasks. In *TAC 2008 Workshop - Notebook papers and results*, pages 158–167, Maryland MD, USA, November 2008.

[GKVS08] George Giannakopoulos, Vangelis Karkaletsis, George Vouros, and Panagiotis Stamatopoulos. Summarization system evaluation revisited: N-gram graphs. *ACM Trans. Speech Lang. Process.*, 5(3):1–39, 2008.

[GMCC00] J. Goldstein, V. Mittal, J. Carbonell, and J. Callan. Creating and evaluating multi-document sentence extract summaries. In *Proceedings of the Ninth International Conference on Information and Knowledge Management*, pages 165–172. ACM Press New York, NY, USA, 2000.

[HB08] Iris Hendrickx and Wauter Bosma. Using correference links and sentence compression in graph-based summarization. In *TAC 2008 Workshop - Notebook papers and results*, pages 429–435, Maryland MD, USA, November 2008.

[HCGL08] Tingting He, Jinguang Chen, Zhuoming Gui, and Fang Li. Ccnu at tac 2008: proceeding on using semantic method for automated summarization yield. In *TAC 2008 Workshop - Notebook papers and results*, pages 158–167, Maryland MD, USA, November 2008.

[HLY$^+$06] Yanxiang He, Dexi Liu, Hua Yang, Donghong Ji, Chong Teng, and Wenqing Qi. Lncs a hybrid sentence ordering strategy in multi-document summarzation. 2006.

[HLZ05] E. Hovy, C. Y. Lin, and L. Zhou. Evaluating duc 2005 using basic elements. *Proceedings of DUC-2005*, 2005.

[HLZF05] E. Hovy, C. Y. Lin, L. Zhou, and J. Fukumoto. Basic elements, 2005.

[HLZF06] E. Hovy, C. Y. Lin, L. Zhou, and J. Fukumoto. Automated summarization evaluation with basic elements. *Proceedings of the Fifth Conference on Language Resources and Evaluation (LREC)*, 2006.

[HMR05] B. Hachey, G. Murray, and D. Reitter. Embra system at duc 2005: Query-oriented multi-document summarization with a very large latent semantic space, the. In *Proceedings of the Document Understanding Conf. Wksp. 2005 (DUC 2005) at the Human Language Technology Conf./Conf. on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, 2005.

[Hof99] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM Press New York, NY, USA, 1999.

[HR06] O. Hamon and M. Rajman. X-score: Automatic evaluation of machine translation grammaticality. *Proceedings of the 5 ThInternational Conference on Language Resources and Evaluation (LREC)*, 2006.

[HS06a] M. Hassel and J. Sjobergh. Towards holistic summarization–selecting summaries, not sentences. In *Proceedings of LREC 2006*, 2006.

[HS06b] John Houvardas and Efstathios Stamatatos. N-gram feature selection for authorship identification. In *J. Euzenat, and J. Domingue (Eds.) Proc. of the 12th Int. Conf. on Artificial Intelligence: Methodology, Systems, Applications (AIMSA'06)*, pages 77–86, 2006.

[HW73] M. Hollander and D. A. Wolfe. Nonparametric statistical inference. *New York*, 1973.

[Jin02] H. Jing. Using hidden markov modeling to decompose human-written summaries. *Computational Linguistics*, 28(4):527–543, 2002.

[Jon99] S. Jones. *Automatic Summarizing: Factors and Directions*, pages 1–12. 1999.

[Jon07] Karen Sparck Jones. Automatic summarising: The state of the art. *Information Processing & Management*, 43(6):1449 – 1481, 2007. Text Summarization.

[Kel00] F. Keller. *Gradience in Grammar*. PhD thesis, 2000.

[Ken62] M. G. Kendall. *Rank Correlation Methods*. Hafner New York, 1962.

[Kle99] J.M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[KM00] K. Knight and D. Marcu. Statistics-based summarization-step one: Sentence compression. In *Proceedings of the International Conference on Artificial Intelligence (AAAI)*, 2000.

[LAC$^+$03] L.S. Larkey, J. Allan, M.E. Connell, A. Bolivar, and C. Wade. UMass at TREC 2002: Cross Language and Novelty Tracks. pages 721–732. National Institute of Standards & Technology, 2003.

[Lam05] S. Lamkhede. Multidocument summarization using concept chain graphs. Master's thesis, 2005.

[Lap03] M. Lapata. Probabilistic text structuring: Experiments with sentence ordering. In *Proceedings of the 41st Meeting of the Association of Computational Linguistics*, pages 545–552, 2003.

[LB05] M. Lapata and R. Barzilay. Automatic Evaluation of Text Coherence: Models and Representations. In *International Joint Conference on Artificial Intelligence 2005*, volume 19, page 1085. LAWRENCE ERLBAUM ASSOCIATES LTD, 2005.

[Led08] Yulia Ledeneva. Effect of preprocessing on extractive summarization with maximal frequent sequences. In *In proceedings of MICAI 2008*, pages 123–132, 2008.

[LH01] C. Y. Lin and E. Hovy. From single to multi-document summarization: a prototype system and its evaluation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 457–464. Association for Computational Linguistics Morristown, NJ, USA, 2001.

[LH02] C. Y. Lin and E. Hovy. Manual and automatic evaluation of summaries. In *Proceedings of the ACL-02 Workshop on Automatic Summarization-Volume 4*, pages 45–51. Association for Computational Linguistics Morristown, NJ, USA, 2002.

[LH03] Chin-Yew Lin and Eduard Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 71–78, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[Lin04] C. Y. Lin. Rouge: A package for automatic evaluation of summaries. *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*, pages 25–26, 2004.

[LOSG06] S. Li, Y. Ouyang, B. Sun, and Z. Guo. Peking university at duc 2006. 2006.

[LOWS07] S. Li, Y. Ouyang, W. Wang, and B. Sun. Multi-document Summarization Using Support Vector Regression. In *Proceedings of Document Understanding Conference (DUC) Workshop 2006*, 2007.

[LRPN07] Daniel S. Leite, Lucia H. M. Rino, Thiago A. S. Pardo, and Maria das Graças V. Nunes. Extractive automatic summarization: Does more linguistic knowledge make a difference? In *Proceedings of the Second Workshop on TextGraphs: Graph-Based Algorithms for Natural Language Processing*, pages 17–24, Rochester, NY, USA, 2007. Association for Computational Linguistics.

[Luh58] H. P. Luhn. Automatic creation of literature abstracts, the. *IBM Journal of Research and Development*, 2(2):159–165, 1958.

[Man01] I. Mani. Automatic summarization. *Computational Linguistics*, 28(2), 2001.

[Mar00] Daniel Marcu. *Theory and Practice of Discourse Parsing and Summarization, The.* The MIT Press, 2000.

[MB97] Inderjeet Mani and Eric Bloedorn. Multi-document summarization by graph search and matching. In *Proceedings of AAAI-97*, pages 622–628. AAAI, 1997.

[MB99] I. M. Mani and E. M. Bloedorn. Summarizing similarities and differences among related documents. *Information Retrieval*, 1(1):35–67, 1999.

[MBF$^+$90] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller. Introduction to wordnet: An on-line lexical database. *International Journal of Lexicography*, 3(4):235–244, 1990.

[MDWD07] A. Mutton, M. Dras, S. Wan, and R. Dale. Gleu: Automatic evaluation of sentence-level fluency. *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 344–351, 2007.

[MGB99] I. Mani, B. Gates, and E. Bloedorn. Improving summaries by revising them. In *Proceedings of the 37th Conference on Association for Computational Linguistics*, pages 558–565. Association for Computational Linguistics Morristown, NJ, USA, 1999.

[MH91] J. Morris and G. Hirst. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1):21–48, 1991.

[Mih04] R. Mihalcea. Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Lingusitics (ACL 2004)(companion Volume)*. ACL, 2004.

[Mih05] R. Mihalcea. Multi-document summarization with iterative graph-based algorithms. In *Proceedings of the First International Conference on Intelligent Analysis Methods and Tools (IA 2005)*. McLean, 2005.

[Mil56] G.A. Miller. The magic number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63(2):81–97, 1956.

[MJ51] F. J. Massey Jr. Kolmogorov-smirnov test for goodness of fit, the. *Journal of the American Statistical Association*, 46(253):68–78, 1951.

[MK04] E. Miltsakaki and K. Kukich. Evaluation of text coherence for electronic essay scoring systems. *Natural Language Engineering*, 10(01):25–55, 2004.

[MKH$^+$99] K. McKeown, J. Klavans, V. Hatzivassiloglou, R. Barzilay, and E. Eskin. Towards multidocument summarization by reformulation: Progress and prospects. *Proceedings of AAAI*, 99:453–460, 1999.

[MLDBGH04] M. J. Maña-López, M. De Buenaga, and J. M. Gómez-Hidalgo. Multidocument summarization: An added value to clustering in interactive retrieval. *ACM Transactions on Information Systems (TOIS)*, 22(2):215–241, 2004.

[MOI01] Y. Matsuo, Y. Ohsawa, and M. Ishizuka. A document as a small world. In *Proceedings the 5th World Multi-Conference on Systemics, Cybenetics and Infomatics (SCI2001*, volume 8, pages 410–414, 2001.

[MR06] A. A. Mohamed and S. Rajasekaran. Query-based summarization based on document graphs. 2006.

[MS99] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, 1999.

[MT87] W. C. Mann and S. A. Thompson. *Rhetorical Structure Theory: A Theory of Text Organization*. University of Southern California, Information Sciences Institute, 1987.

[Nas08] Vivi Nastase. Topic-driven multi-document summarization with encyclopedic knowledge and spreading activation. In *Proceedings on the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 763–772. Association for Computational Linguistics, October 2008.

[Nav01] G. Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, 2001.

[Nen06] A. Nenkova. *Understanding the Process of Multi-Document Summarization: Content Selection, Rewriting and Evaluation*. PhD thesis, 2006.

[NPDP05] J. Niekrasz, M. Purver, J. Dowding, and S. Peters. Ontology-based discourse understanding for a persistent meeting assistant. In *Proceedings of the AAAI Spring Symposium Persistent Assistants: Living and Working with AI*, 2005.

[OER05] Jahna Otterbacher, Güneş Erkan, and Dragomir R. Radev. Using random walks for question-focused sentence retrieval. In *HLT '05: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 915–922, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

[ORL02] J. C. Otterbacher, D. R. Radev, and A. Luo. Revisions that improve cohesion in multi-document summaries: a preliminary study. In *Proceedings of the ACL-02 Workshop on Automatic Summarization-Volume 4*, pages 27–36. Association for Computational Linguistics Morristown, NJ, USA, 2002.

[PLA$^+$06] S. Park, J. H. Lee, C. M. Ahn, J. S. Hong, and S. J. Chun. Query based summarization using non-negative matrix factorization. *Proceeding of KES*, pages 84–89, 2006.

[PMSG06] R. J. Passonneau, K. McKeown, S. Sigelman, and A. Goodkind. Applying the pyramid method in the 2006 document understanding conference. In *Proceedings of Document Understanding Conference (DUC) Workshop 2006*, 2006.

[PRWZ01] K. Papineni, S. Roukos, T. Ward, and W. J. Zhu. Bleu: a method for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318, 2001.

[PS04] C. I. A. Prince and P. Smolensky. Optimality theory: Constraint interaction in generative grammar. *Optimality Theory in Phonology: A Reader*, 2004.

[QF93] Y. Qiu and H.P. Frei. Concept based query expansion. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 160–169. ACM Press New York, NY, USA, 1993.

[Rad00] D. Radev. A common theory of information fusion from multiple text sources, step one: Cross-document structure. In *Proceedings, 1st ACL SIGDIAL Workshop on Discourse and Dialogue*, 2000.

[RBGZ01] D. Radev, S. Blair-Goldensohn, and Z. Zhang. Experiments in single and multidocument summarization using mead. *First Document Understanding Conference*, 2001.

[RD00] E. Reiter and R. Dale. Building applied natural language generation systems. *Natural Language Engineering*, 3(01):57–87, 2000.

[RGW02] J. W. Raymond, E. J. Gardiner, and P. Willett. Rascal: Calculation of graph similarity using maximum common edge subgraphs. *The Computer Journal*, 45(6):631, 2002.

[RHM02] Dragomir R. Radev, Eduard Hovy, and Kathleen McKeown. Introduction to the special issue on summarization. *Computational Linguistics*, 28(4):399–408, 12 2002.

[RJB00] D. R. Radev, H. Jing, and M. Budzikowska. Centroid-based summarization of multiple documents: Sentence extraction, utility-based evaluation, and user studies. *ANLP/NAACL Workshop on Summarization*, 2000.

[RJST04] Dragomir R. Radev, Hongyan Jing, Malgorzata Stys, and Daniel Tam. Centroid-based summarization of multiple documents. *Information Processing & Management*, 40(6):919 – 938, 2004.

[RJZ89] LF Rau, PS Jacobs, and U. Zernik. Information extraction and text summarization using linguistic knowledge acquisition. *Information Processing and Management: an International Journal*, 25(4):419–428, 1989.

[RKEA00] N. Reithinger, M. Kipp, R. Engel, and J. Alexandersson. Summarizing multilingual spoken negotiation dialogues. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 310–317. Association for Computational Linguistics Morristown, NJ, USA, 2000.

[ROWBG05] D. Radev, J. Otterbacher, A. Winkel, and S. Blair-Goldensohn. Newsinessence: Summarizing online news topics. *Communications of the ACM*, 48(10):95–98, 2005.

[Sag05] H. Saggion. Topic-based summarization at duc 2005. In *Proceedings of the Document Understanding Conf. Wksp. 2005 (DUC 2005) at the Human Language Technology Conf./Conf. on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, 2005.

[Sag06] H. Saggion. Multilingual multidocument summarization tools and evaluation. In *Proceedings, LREC 2006*, 2006.

[SDC04] A. Sengupta, M. Dalkilic, and J. Costello. Semantic thumbnails: a novel method for summarizing document collections. *Proceedings of the 22nd Annual International Conference on Design of Communication: The Engineering of Quality Documentation*, pages 45–51, 2004.

[SEKA05] Y. Seki, K. Eguchi, N. Kando, and M. Aono. Multi-document summarization with subjectivity analysis at duc 2005. In *Proceedings of the Document Understanding Conf. Wksp. 2005 (DUC 2005) at the Human Language Technology Conf./Conf. on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, 2005.

[SHW06] A. K. Seewald, C. Holzbaur, and G. Widmer. Evaluation of term utility functions for very short multi-document summaries. *Applied Artificial Intelligence*, 20:57–77, 2006.

[SJ04] J. Steinberger and K. Jezek. Using latent semantic analysis in text summarization and summary evaluation. In *Proc. ISIM '04*, pages 93–100, 2004.

[Sjö07] Jonas Sjöbergh. Older versions of the ROUGEeval summarization evaluation system were easier to fool. *Information Processing & Management*, 43(6):1500–1505, November 2007.

[SJWS02] A. Spink, B. J. Jansen, D. Wolfram, and T. Saracevic. From e-sex to e-commerce: Web search changes. *IEEE Computer*, 35(3):107–109, 2002.

[SK05] A. Sorace and F. Keller. Gradience in linguistic data. *Lingua*, 115(11):1497–1524, 2005.

[SL02] H. Saggion and G. Lapalme. Generating indicative-informative summaries with sumum. *Computational Linguistics*, 28(4):497–526, 2002.

[SM86] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, Inc. New York, NY, USA, 1986.

[SM02] G. Silber and K. McCoy. Efficiently computed lexical chains as an intermediate representation for automatic text summarization. *Computational Linguistics*, pages 487–496, 12 2002.

[SN03] S. Sekine and C. Nobata. A survey for multi-document summarization. *Proceedings of the HLT-NAACL 03 on Text Summarization Workshop-Volume 5*, pages 65–72, 2003.

[SOC$^+$02] J. D. Schlesinger, M. E. Okurowski, J. M. Conroy, D. P. O'Leary, A. Taylor, J. Hobbs, and W. H. T. Wilson. Understanding machine performance in the context of human performance for multi-document summarization. In *Proceedings of the DUC 2002 Workshop on Multi-Document Summarization Evaluation*, 2002.

[Spe06] C. Spearman. Footrule for measuring correlation. *British Journal of Psychology*, 2:89–108, 1906.

[SS05] M. Soubbotin and S. Soubbotin. Trade-off between factors influencing quality of the summary. 2005.

[Ste74] MA Stephens. Edf statistics for goodness of fit and some comparisons. *Journal of the American Statistical Association*, 69(347):730–737, 1974.

[SYGH05] Ou Shi-Yan, Khoo Christopher S. G., and Goh Dion H. Constructing a taxonomy to support multi-document summarization of dissertation abstracts. *Journal of Zhejiang University Science*, 6A(11):1258–1267, 2005.

[TAGMS05] R. Torralbo, E. Alfonseca, J. M. Guirao, and A. Moreno-Sandoval. Description of the uam system at duc-2005. In *Proceedings of the Document Understanding Conf. Wksp. 2005 (DUC 2005) at the Human Language Technology Conf./Conf. on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, 2005.

[Tai05] J. Tait. Making Better Summary Evaluations. In *Proceedings of the Internatinal Workshop:≪ Crossing Barriers in Text Summarisation Research≫ Horacio Saggion and Jean-Luc Minnel (eds.), Borovets, Bulgaria, Septemeber*, pages 26–31, 2005.

[TH08] Stephen Tratz and Eduard Hovy. Summarization evaluation using transformed basic elements. In *TAC 2008 Workshop - Notebook papers and results*, pages 36–45, Maryland MD, USA, November 2008.

[TK03] S. Theodoridis and K. Koutroumbas. *Pattern Recognition.* Academic Press, 2003.

[TM02] S. Teufel and M. Moens. Summarizing scientific articles: Experiments with relevance and rhetorical status. *Computational Linguistics*, 28(4):409–445, 2002.

[TNI04] Junji Tomita, Hidekazu Nakawatase, and Megumi Ishii. Calculating similarity between texts using graph-based text representation model. pages 248–249, Washington, D.C., USA, 2004. ACM.

[Tsa93] E. Tsang. *Foundations of constraint satisfaction.* Academic Press San Diego, 1993.

[VH06] R. Varadarajan and V. Hristidis. A system for query-specific document summarization. *Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, pages 622–631, 2006.

[VHT03] H. Van Halteren and S. Teufel. Examining the consensus between human summaries: Initial experiments with factoid analysis. In *Proceedings of the HLT-NAACL 03 on Text Summarization Workshop-Volume 5*, pages 57–64. Association for Computational Linguistics Morristown, NJ, USA, 2003.

[VKB$^+$06] I. Vlahavas, P. Kefalas, N. Bassiliades, F. Kokkoras, and I. Sakellariou. *Artificial Intelligence, 3rd Edition (in Greek)*. V.Giurdas Publications, 2006.

[Voo94] E.M. Voorhees. Query expansion using lexical-semantic relations. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 61–69. Springer-Verlag New York, Inc. New York, NY, USA, 1994.

[Voo03] E. M. Voorhees. Overview of the trec 2003 question answering track. In *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003)*, 2003.

[WF05] I.H. Witten and E. Frank. Data Mining: Practical Machine Learning Tools and Techniques. 2005.

[WFT$^+$99] I. H. Witten, E. Frank, L. Trigg, M. Hall, G. Holmes, and S. J. Cunningham. Weka: Practical machine learning tools and techniques with java implementations. *ICONIP/ANZIIS/ANNES*, pages 192–196, 1999.

[Wil45] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, pages 80–83, 1945.

[WKB05] René Witte, R. Krestel, and S. Bergler. Erss 2005: Coreference-based summarization reloaded. In *Proceedings of the Document Understanding Conf. Wksp. 2005 (DUC 2005) at the Human Language Technology Conf./Conf. on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, 2005.

[WKB06] René Witte, Ralf Krestel, and Sabine Bergler. Context-based multi-document summarization using fuzzy coreference cluster graphs. In *Proceedings of Document Understanding Workshop (DUC)*, New York City, NY, USA, June 2006.

[WKC$^+$00] M. White, T. Korelsky, C. Cardie, V. Ng, D. Pierce, and K. Wagstaff. Multidocument summarization via information extraction. In *Proceedings of the First International Conference on Human Language Technology Research*, pages 1–7. Association for Computational Linguistics Morristown, NJ, USA, 2000.

[WL03] C. W. Wu and C. L. Liu. Ontology-based text summarization for business news articles. *Proceedings of the ISCA 18th International Conference on Computers and Their Applications*, pages 389–392, 2003.

[Wol87] S. Wold. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1):37–52, 1987.

[Yar95] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics Morristown, NJ, USA, 1995.

[YP06] RR Yager and FE Petry. A multicriteria approach to data summarization using concept ontologies. *Fuzzy Systems, IEEE Transactions on*, 14(6):767–780, 2006.

[ZBGR02] Z. Zhang, S. Blair-Goldensohn, and D. Radev. Towards cst-enhanced summarization. In *Proceedings of the 18th National Conference on Artificial Intelligence*, 2002.

[ZCM02] Y. Zhang, J. Callan, and T. Minka. Novelty and redundancy detection in adaptive filtering. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 81–88. ACM New York, NY, USA, 2002.

[ZLMH06] L. Zhou, C. Y. Lin, D. S. Munteanu, and E. Hovy. Paraeval: Using paraphrases to evaluate summaries automatically. In *Proceedings of the Human Language Technology Conference - North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT/NAACL-2006)*, 2006.

[ZSN05] Q. Zhou, L. Sun, and J. Y. Nie. Is_sum: A multi-document summarizer based on document index graphic and lexical chains. In *Proceedings of the Document Understanding Conf. Wksp. 2005 (DUC 2005) at the Human Language Technology Conf./Conf. on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, 2005.

Index