MULTI-DOCUMENT SUMMARIZATION THROUGH SENTENCE FUSION

Rémi BOIS
Supervised by Florian BOUDIN

Université de Nantes - LINA

December 8, 2014

Abstract

The plethora of information available on the web have made the need for efficient automatic summarization tools more urgent over the years.

While extracting sentences to compile a summary appears to be insufficient, abstractive approaches are gradually gaining ground. However the existing abstractive techniques rely on heavy linguistic resources, making them domain-dependent and language dependent.

In this document, we introduce a new abstractive summarization process that does not rely on heavy resources and that is competitive with state-of-the-art systems. By the use of sentence fusion and Integer Linear Programming, it can create new sentences conveying information spread among documents and unite them into comprehensive summaries.

Keywords. Multi-document Summarization, Sentence Fusion, Integer Linear Programming

Contents

1 Introduction **1**

2 Research Statement **3**

3 Previous Work **5**

3.1 Extractive Summarization 5

3.2 Abstractive Summarization 6

3.3 Sentence selection : Integer Linear Programming 7

4 System Description **9**

4.1 System Framework 9

4.2 Sentence Clustering 10

4.3 Sentence Fusion 13

4.4 Sentence Selection 15

5 Experiments **19**

5.1 Data and Evaluation measures 19

5.2 Automatic Evaluation 22

5.3 Manual Grammaticality Evaluation 22

6 Discussion **25**

6.1 Strengths and Weaknesses of Sentence Fusion 25

6.2 Minor Parameter Modification Implies Great Changes 26

6.3 Of the Need for Resources 27

7 Conclusion and Perspectives **29**

Bibliography **31**

A System's Details **35**

A.1 Usage Example 35

A.2 System Parameters 36

Acknowledgments

This work took place during a six-month internship at LINA[1], within the TALN[2] team. Supervised by Florian Boudin and tutored by Colin de la Higuera, it benefited from the experience of the TALN team on natural language processing.

Florian Boudin's expertise in automatic summarization techniques (Favre et al., 2006; Boudin, 2008; Boudin et al., 2011) allowed for this study to take place and brought great pieces of advice all along this work.

The TALN team has been very supportive during the internship, bringing ideas and making me feel part of the team. Many thanks to all of them.

[1] Laboratoire Informatique de Nantes Atlantique
[2] Traitement Automatique du Language Naturel

Chapter 1

Introduction

Summarization is widely used in everyday life. We look for back covers when we consider buying a book and read synopses to choose which movie to watch. We read newspapers' headlines to decide if the full articles are worth reading and we select which websites we visit based on the excerpts given by search engines. We read sport results summaries to avoid watching every match and we listen to friends summarizing their trip abroad.

These abstracts are almost always produced by humans and the only instances of day-to-day automatic summaries are incarnated by search engines' results that extract sentences from webpages to create an artificial summary. While these excerpts do offer an insight of the documents contents, they lack the ability to compress the information into a coherent, comprehensive text.

Summaries can be categorized as informative or indicative (Borko and Bernier, 1975). Indicative summaries aim at giving an insight at the content of a document without trying to summarize the information it contains. Article headlines and search engine results' excerpts belong to this category. Informative summaries aim at compressing the information found in a document so that the user doesn't need to read it to know the main information it contains. Sports results are one example of informative summaries. All these examples aim at summarizing the content of a single document.

The advent of the Internet lead to an ever increasing number of people discussing similar topics and put into light another challenge : summarizing multiple documents that convey similar information. This task, while sharing the objective and some of the means of single-document summarization, also holds unique characteristics. Important information may be repeated across documents and thus be easier to find. However, similar information can be conveyed while using completely different terms, often leading to repetition in automatically generated summaries.

Finding similar documents in order to regroup similar information is already efficiently used in many websites (e.g. Google News) but none of them offers comprehensive summaries of the

information they regroup.

To overcome this problem, we propose a multi-document summarization approach that takes advantage of repetition across documents to fuse similar sentences and assemble them in a coherent short summary. Our solution is competitive with other state-of-the-art approaches and alleviates the need for heavy language resources, making it usable in many languages and in many domains.

This document is organized as follows. Chapter 2 exposes the main challenge our work aims to solve. Chapter 3 describes previous work on summarization, from early works on extractive summarization to recent advances in abstractive summarization as well as the commonly used ILP framework for sentence selection. Chapter 4 elaborates on our system framework and the different steps involved during our automatic summarization process. Chapter 5 reports the results we obtained and compares the performance of our approach against other similar state-of-the-art systems. Finally, Chapter 6 discusses the benefits of our approach as well as its limits.

Chapter 2

Research Statement

Today's web is multilingual, mutlimodal and oversized. When looking for an information, chances are that many documents discuss the precise topic you want. While this overload of information allows to find different views on the same topic, the available information is underused. To answer your information need, you have to visit websites based on small excerpts that give you a hint about the chances they discuss the right topic. Then, you have to navigate in the documents that seem relevant and hope the information is there.

When you're information need is light and generic (e.g. a definition, a short description of a concept), you're better off with trying a single source you believe (be it a dictionary or an encyclopedia). However, using this method makes you consider a single source of information that could be biased.

Our interest lies in taking advantage of the overload of information and use many documents in order to gather and regroup information in a well-constructed summary. But doesn't this already exist you may ask?

While summarization is a well studied topic, there exist no instance of automatic summarization on the web. Why is there such a difference between the impressive results that are obtained in the main conferences and the field presence? One of the main answers to this question is the lack of corpora, almost only made available via conferences that focus on almost a single domain (namely newswire) and on a single language (English). As a direct consequence, most systems that are submitted to these conferences or that evaluate themselves on these corpora have a huge bias : they focus on learning how to summarize English newswire, and become better and better at this task by using more and more resources. These resources are highly specialized, and mostly not available in alternative languages.

This resource need is incompatible with the Internet multilinguality. To overcome this problem, we introduce a new framework for summarization that alleviates the need for these spe-

cialized resources. Our approach aims at being multilingual and domain-agnostic and uses no resources beyond a POS-tagger.

We compare our approach with a variant that uses resources to strengthen information gathering and show that the absence of resources is paradoxically beneficial to our method. We also compare our system to a similar state-of-the-art system that relies on heavy resources and show that our system is at least as efficient as the resource-heavy one.

Chapter 3

Previous Work

In this chapter, we describe the existing works that are relevant to our approach and introduce some of the concepts that are used in our work.

Automatic summarization has been studied since the 50's (Luhn, 1958). Existing approaches can be separated into two categories (Mani, 2001): extractive summarization, which aims at extracting the best subset of sentences from documents in order to cover all major information (Rath et al., 1961) and abstractive summarization that advocates for a human-like generation of summaries, generating sentences that don't necessary appear in the documents.

First, we discuss how summarization has been mostly explored under the scope of sentence extraction. We discuss the intrinsic limits of this approach and describe another line of work that holds great expectations: abstractive summarization. Finally, we show how the problem of selecting the best set of candidates sentences to include in a summary can be solved through Integer Linear Programming and the prominent place this approach took in recent research.

3.1 Extractive Summarization

Extractive summarization has been applied to both single-document and multi-document summarization (Barzilay et al., 1997; Goldstein et al., 2000), and sentence relevance ranking has mainly been considered as a classification task (Teufel, 1997). By using a set of features that describe a sentence, classifiers can decide if a sentence should, or should not be included in a summary.

Most common features are sentence location (sentences appearing at the start of a document or paragraph are statistically more important), number of Named Entities (including locations and protagonists), sentence length, term importance (via term frequency or $tf.idf$) (Lin, 1999).

While it is rare to extract multiple sentences that convey the same information when dealing with single-document summarization, it becomes a liability when trying to apply the same technique to multi-document summarization. The objective of multi-document summarization can then be described as maximizing informativity and minimizing redundancy while staying under a limited summary size.

For a long time, the main approach to overcome this liability have been to cluster the events that are to be summarized and to select only one sentence from each cluster, thus limiting redundancy (Radev et al., 2000).

Recently, a new framework has been introduced that links the sentence selection to a well-known optimisation problem: the knapsack problem (McDonald, 2007). This framework called ILP allows to select optimally the best subset of sentences according to their length, informativity, and redundancy. We describe this approach in more depth in section 3.3.

These extractive approaches generally obtain higher results than abstractive systems in the DUC[1] and TAC[2] conferences. However, Genest et al. (2009) showed that existing extractive strategies are really close to what humans can achieve by the means of sentence extraction and are yet largely inferior to human-written abstracts. This lead the research community to investigate abstractive methods in more depth.

3.2 Abstractive Summarization

Abstractive methods regroup a wide range of approaches, from domain-specific template-based methods relying on information extraction (White et al., 2001) to fully abstractive text-to-text generation, an approach that holds great expectations but that is still at an early stage (Genest and Lapalme, 2012).

Somewhere between those two extremes lies another approach which consists in modifying the source text sentences in order to create alternative sentences that are either shorter, or that combine information found in different sentences.

Shortening sentences is known as sentence compression, and has been successfully used to improve extractive systems (Gillick et al., 2009). By compressing sentences, via temporal clauses removal or deletion of unnecessary phrases, more sentences, and hopefully more information, can be added to the summary (Knight and Marcu, 2002; Galley and McKeown, 2007).

Combining different sentences to create a more informative sentence is known as sentence fusion. Fusing sentences allows to create a new sentence that regroups information spread

[1] Document Understanding Conference
[2] Text Analysis Conference

across different source sentences, and can improve in many ways a summary (e.g. reducing redundancy while improving coherence and information coverage). One way to fuse sentences is to use their dependency parse trees and align their branches before generating a new sentence from the fused parse tree, a process known as linearization. However, creating a sentence from the fusion of the parse trees is difficult and often leads to ungrammatical sentences (Filippova and Strube, 2009).

Barzilay and McKeown (2005) were among the first to introduce a competitive multi-document summarization system based on sentence fusion. After clustering related sentences into themes, they fuse the dependency parse trees of sentences in each cluster and generate sentences, ultimately selecting the best fusion via scoring against a language model. They use machine learning techniques for clustering, and select the relevant themes depending on the number of sentences they hold, the similarity between their sentences and a significance score computed from the lexical chains of the documents.

Another method for sentence fusion that does not rely on external resources has been introduced by Filippova (2010). Her approach consists in using a word graph of the sentences to be fused, and choosing a path in the graph that keeps the common information while providing a new sentence. This work was later extended by Boudin and Morin (2013) to generate more informative sentences by reranking fusion candidates according to the keyphrases they contain.

Sentence fusion is a difficult task in itself, and its feasibility has been questioned (Daume III and Marcu, 2004), however, its promising results make it an interesting domain despite the difficulties to evaluate it intrinsically (Thadani and McKeown, 2013).

3.3 Sentence selection : Integer Linear Programming

While sentence compression and sentence fusion offer richer variations of sentences to include in a summary, one still needs to choose which sentences should be added. Optimally choosing the sentences to include by having the maximum of informativeness while keeping a low redundancy and limiting the size can be linked to the knapsack problem, a global optimization problem.

The knapsack problem is formulated as:

> Given a set of items, each with a mass and a value, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible.

In automatic summarization, the items are sentences, their weight is the number of words they contain, and their value is their informativity. The knapsack size is the number of words

we limit ourselves for the summary. This number can be set arbitrarily to a fixed number of words or to a percentage of the size of the original documents.

The knapsack problem is NP-hard, meaning it can't be solved in polynomial time. However, as it is a very common problem, many solvers can solve a knapsack instance quickly if the number of parameters (that is the variables and constraints) are not too numerous via the use of Integer Linear Programming.

The use of Integer Linear Programming to solve the sentence selection problem has been introduced by McDonald (2007). His approach, while it produces optimized summaries, can not be used for large documents, because of its time complexity. An optimization that relies on weighting the concepts inside sentences instead of the full sentences has later been introduced (Gillick and Favre, 2009). While it necessitates to define concepts as well as a way to score them instead of scoring sentences, this approach can be scaled to process documents in a matter of seconds. The common representation of concepts in ILP-based approaches is bigrams, and one of the best-performing scoring method is the bigram's document frequency (that is the number of documents in which a bigram appears).

Several works have since used this framework as a starting point for new improvements on summarization. Some works extended the ILP objective by combining content and surface realization scores (Woodsend and Lapata, 2012; Li et al., 2011). Other works improved the sentence compression model used in (Gillick and Favre, 2009), by performing sentence selection and compression in a single step (Martins and Smith, 2009). These systems usually rely on models trained on preceding TAC datasets to perform sentence compression (Qian and Liu, 2013; Li et al., 2013).

Some works focus on learning compression models based on manually compressed sentences. However, the limited amount of data available makes it hard to learn efficiently (Daume, 2006) and limits the number of features that can be used (Berg-Kirkpatrick et al., 2011).

The ILP framework approach is currently one of the most popular ones among the scientific community.

Chapter 4

System Description

In this chapter, we present how our system has been built as well as the different steps it follows to perform automatic summarization.

First, we introduce the global framework of our system by describing the libraries and resources it relies on as well as the major previous works that we based our system on.

Then, we successively describe in depth each of the three steps involved in summarization : sentence clustering, sentence fusion and sentence selection.

4.1 System Framework

Our system performs multi-document abstractive summarization via sentence fusion and Integer Linear Programming (ILP) sentence selection. It is implemented in the Python Language[1] and uses already existing modules for sentence fusion and ILP solving. Hereafter is described the framework of our system, and Figure 4.1 illustrates our framework.

First, our system takes a set of related texts as input and preprocesses them. The preprocessing step includes tokenization, Part-Of-Speech (POS) tagging, removal of stopwords and stemming. For all these steps, we use the Python nltk toolkit (Bird, 2006). More specifically, we use the punkt sentence tokenizer, the Penn Treebank word tokenizer, the Stanford POS-tagger[2] and the Snowball Stemmer. We use the stopword list included in nltk to filter irrelevant words. This preprocessing step allows us to obtain a more accurate representation of the information included in each sentence, and makes similarity measurement more efficient.

The second step consists in clustering similar sentences. This clustering step allows for the third step, namely sentence fusion, to take place. By regrouping sentences that convey the

[1] `https://www.python.org/`
[2] `http://nlp.stanford.edu/software/tagger.shtml`

same information, we can fuse them and obtain unique new sentences that are either shorter than the original sentences, or that convey more information. Finally, we use ILP to select the best subset of sentences, based on the number of concepts each sentence holds, and finding the optimal combination of sentences that maximize informativity while minimizing redundancy.

All these different steps are issued from previous research works that proved efficient separately. Filippova (2010) introduced sentence fusion, later improved by Boudin and Morin (2013), and tested their approach on clusters composed of dozens of sentences, while our use case of summarization leads to small clusters. Gillick and Favre (2009) introduced the ILP framework we rely on, and showed that it was highly competitive with other state-of-the-art heavier systems. We aim at assembling these methods in order to propose an efficient system for abstractive automatic summarization.

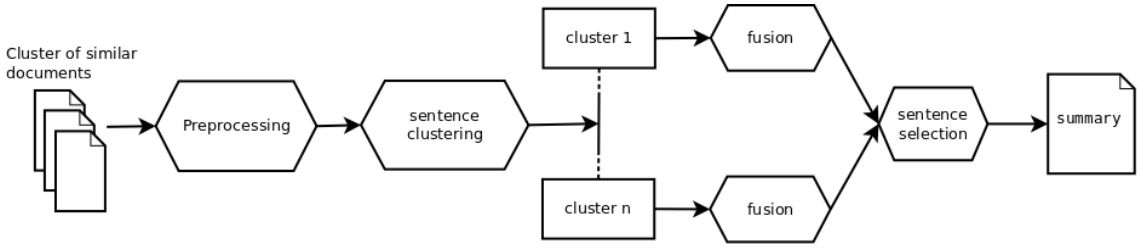


Figure 4.1: System Workflow

This system does not use any linguistic resources except a list of stopwords (that can be easily found in almost any language) and a POS-tagger. We developed an open-source version of our system that is available on github[3]. It is fully documented and largely tested.

4.2 Sentence Clustering

The sentence clustering step allows us to regroup similar sentences in order to generate alternative sentences obtained by fusing sentences that belong to the same cluster. This is a crucial step as if we can't find enough clusters, we won't be able to generate any fused sentences, and if we are too broad during clustering we may try to fuse dissimilar sentences, thus resulting in incoherent fused sentences.

To circumvent the risk of clustering together too many sentences, we use Hierarchical Agglomerative Clustering with a complete-linkage strategy (see Figure 4.2 for an illustration). This method proceeds incrementally, starting with each sentence considered as a cluster, and merging the two most similar clusters after each step. The complete-linkage strategy defines the similarity between two clusters as the lowest similarity score between two items of the clusters. Clusters may be small, but are highly coherent as each sentence they contain must be similar to every other sentence in the same cluster.

[3] `https://github.com/sildar/potara`

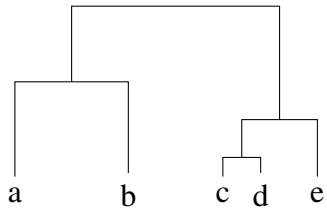


Figure 4.2: Hierachical Agglomerative Clustering

We set a similarity threshold τ to stop the clustering process. If we cannot find any cluster pair with a similarity above the threshold, the process stops, and the clusters are frozen. We use two different measures, both comparing sentences after stopword removal, POS-tagging and stemming.

The first measure we experiment with is cosine similarity over bag-of-stems. This measure captures content similarity using lexical overlap and thus can not identify semantically-related words such as synonyms or hyponyms. To address this issue, we try a second similarity measure which uses Word2Vec (Mikolov et al., 2013)[4].

Word2Vec is a method that represents words as vectors by learning the context in which they appear. By training a Neural Network on enough data, Word2Vec naturally exhibits semantic relations between words. As words are represented as vectors, we use a cosine similarity for computing word-word similarity. This second measure allows us to compare our resource-free approach based on lexical overlap with a resource-heavy approach based on semantic similarity in order to better evaluate the need for external resources. This is discussed in more depth in Chapter 6.3

We train Word2Vec on the first 10^9 bytes (approximately 160,000 articles) of the English Wikipedia[5]. The similarity between two sentences is computed as the sum of the maximum Word2Vec score of each word pair normalized by the length of the shortest sentence. To avoid multiple word matching, we remove the pair of words with the maximum score at each iteration. A similarity score of 0 is given when the two sentences do not have any words in common, as our sentence fusion module requires at least one word in common to operate. More formally, the similarity between the two sentences $S_1 = \{a_1, a_2, ..., a_i\}$ and $S_2 = \{b_1, b_2, ..., b_j\}$ with $|S_1| \leq |S_2|$ is defined as:

[4]We use the Gensim Python implementation of Word2Vec (Řehůřek and Sojka, 2010).
[5]`http://mattmahoney.net/dc/textdata.html`

$$Sim(S_1, S_2) = \frac{1}{|S_1|} \sum_i \max_{b \in S^i} \Big(word2vec(a_i, b) \Big)$$
$$\text{with} \quad S^0 = S_2$$
$$S^{i+1} = S^i \setminus \arg\max_{b \in S^i} (word2vec(a_i, b))$$

To find the optimal clustering threshold τ for each similarity measure, we use the SICK dataset of SemEval-2014[6]. This dataset is made of 4,500 sentence pairs, each annotated for relatedness in meaning. Scores range from 1 (completely unrelated) to 5 (very related). As we are interested in identifying related sentences, we discard the subset of sentences that are only partially related (i.e. scores ranging from 2.5 to 3.5). Sentence pairs with a relatedness score lower than 2.5 are considered dissimilar, while those with a score above 3.5 are considered similar. The remaining dataset is composed of 2,458 similar sentence pairs and 638 dissimilar sentence pairs.

Results in terms of precision, recall and f-measure on the SICK dataset for both similarity measures at their optimal threshold τ are presented in Table 4.1. Here, sentence pairs with a similarity above τ are being considered similar. As this dataset is not balanced, we also report the specificity score (true negatives rate). We observe that the performance of the Word2Vec-based similarity measure (SemSim) is slightly better than that of the bag-of-stems similarity measure (LexSim), both achieving an f-measure above 90%. The optimal values for the τ parameters, namely 0.35 for LexSim and 0.5 for SemSim, are used in our system.

	τ	**P**	**R**	**F**	**Spe.**
LexSim	0.35	0.87	0.93	0.90	0.49
SemSim	0.50	**0.88**	**0.94**	**0.91**	**0.53**

Table 4.1: Results in terms of precision (P), recall (R), f-measure (F), and specificity (Spe.) for sentence similarity detection along with the optimal thresholds τ.

Error analysis shows that most of the errors can be attributed to the fact that sentences in the SICK dataset are very short (less than 10 words on average) and thus contain very few words on which the similarity measures can be computed. To illustrate this, an example of sentence pair is given below:

(1) The man is playing with a skull

(2) The man is playing the guitar

While these two sentences have most of their words in common (2/3 if we exclude stop-

[6] http://alt.qcri.org/semeval2014/task1/

words), they are annotated as dissimilar (1.9/5) as they do not describe the same event. None of the similarity measures, when using the optimal values for τ, are able to detect that these sentences are dissimilar. Nevertheless, we expect the similarity measures to perform better on the dataset that we use to test our whole system, since it contains newswire articles describing the same events and longer sentences (22 words on average).

4.3 Sentence Fusion

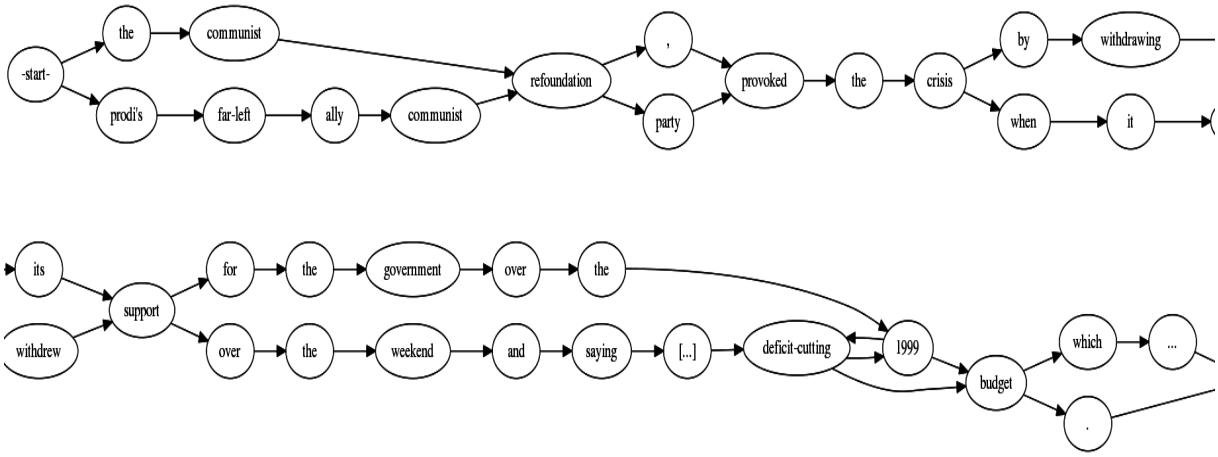
Sentence fusions are generated using Filippova (2010)'s method, implemented in the `takahe` module[7]. Her approach consists in building a directed word graph from a set of similar sentences, in which nodes represent unique words, defined as word and POS tuples, and edges express the original structure of sentences (i.e. word ordering). Sentence fusions are then obtained by finding commonly used paths in the graph. Here, redundancy within the input sentences provides a reliable way of generating both informative and grammatical sentences.

Figure 4.3 illustrates the fusion process, showing the word graph, the two sentences (1) and (2) that are involved in fusion (we report only two sentences out of a cluster of five sentences in order to improve readability) and the fusion that is generated (F). The end node has also been omitted.

An enhancement of Filippova (2010)'s approach was recently proposed by (Boudin and Morin, 2013). They use a N-best re-ranking strategy to produce more informative fusions by re-ranking fusion candidates according to the number and relevance of the keyphrases they contain. Keyphrases are defined as words or phrases that capture the main topics of the set of input sentences, and they are detected using the TextRank method (Mihalcea and Tarau, 2004).

In this work, we experiment with both Filippova (2010)'s and Boudin and Morin (2013)'s word-graph based sentence fusion approaches. Similarly to (Gillick et al., 2009), we generate up to 10 fusions per sentence cluster, each fusion having a minimal length of 6 words, in order to avoid ill-formed sentences. For clusters composed of only one sentence, no fusion is generated.

[7] `https://github.com/boudinfl/takahe`



(a) Word graph

(1) *The Communist Refoundation Party provoked the* **crisis** by withdrawing its support over the weekend and saying it would not vote for his deficit-cutting **1999** *budget.*

(2) Prodi's far-left ally, the Communist Refoundation Party, provoked the **crisis** *when it withdrew support for the government over the* **1999** deficit-cutting budget, which it said did not do enough to stimulate job creation.

(F) *The Communist Refoundation Party provoked the* **crisis** *when it withdrew support for the government over the* **1999** *budget.*

(b) Text sentences and their fusion

Figure 4.3: Word graph of two sentences that lead to a more efficient fused sentence

Sentence fusions that are generated by this method only consist of words that appear in the set of input sentences. However, phrases are added, removed or modified, so that fused sentences are often more informative while being shorter. An example of sentence fusion obtained from a set of three related sentences (topic D30055) is given below :

(1) *Prime Minister Rafik* **Hariri** has declined an informal invitation from Lebanon's new president to form the next government, sparking a political crisis in this country as it rebuilds from its devastating civil war.

(2) **Hariri**, *Lebanon's top businessman,* **has** almost single-handedly created a multibillion dollar program to rebuild a country destroyed by the civil war.

(3) Hariri, 53, the architect of Lebanon's multibillion dollar postwar reconstruction program, **has** *been in power since 1992*.

(F) *Prime Minister Rafik* **Hariri**, *Lebanon's top businessman,* **has** *been in power since 1992.*

We observe that the generated fusion contains information nuggets originating from each input sentence. Redundant words or phrases act as pivots to pass from a sentence to another.

Depending on the level of redundancy within the set of input sentences, fusing sentences using Filippova (2010)'s method can give a fusion that is similar to what we would obtain using sentence compression. To illustrate this, another example of sentence fusion (topic D30011) is given below:

(1) *Anwar* **has been** accused of engaging in homosexual acts illegal under Malaysia law, but the charges are generally seen as a pretext for his political persecution.

(2) Anwar since **has been** *charged with corruption and illegal homosexual acts, and is to go on trial Nov. 2.*

(F) *Anwar* **has been** *charged with corruption and illegal homosexual acts, and is to go on trial Nov. 2.*

The fused sentence is only one word shorter and contains information from only one input sentence. Here, the advantage of using Filippova (2010)'s approach over a sentence compression method is that it does not require any manually crafted rules, syntactic parser or training data.

4.4 Sentence Selection

As exposed by McDonald (2007) and Gillick and Favre (2009), we consider the sentence selection problem as being a variation of the knapsack problem (see Section 3.3 for an introduction to the knapsack problem). The problem is then to find the set of sentences that is both relevant and non-redundant.

In this work, we use the concept-based ILP framework introduced in (Gillick and Favre, 2009). This approach aims to extract sentences that cover as many important concepts as possible, while ensuring the summary length is within a given constraint. We follow Gillick and Favre (2009) and use bigrams, after stemming and stopwords removal, as concepts, and assign a weight to each bigram using its document frequency. Bigrams consisting of two stopwords or one punctuation mark are pruned, as are those appearing in fewer than four documents (that is, less than one third of the documents).

ILP allows to represent a problem as a function to maximize while respecting some constraints. The mathematical objects it deals with must be integers, and while solving ILP problems is NP-hard, a solution is usually found quickly by solvers through the use of heuristics.

Let w_i be the weight of concept i and c_i a binary variable that indicates the presence of concept i in the summary. Let l_j be the number of words in sentence j, s_j a binary variable that indicates the presence of sentence j in the summary and L the length limit for the whole summary. Let Occ_{ij} indicate the occurrence of concept i in sentence j, the ILP formulation we

use is:

$$\text{Maximize :} \quad \sum_i w_i c_i \tag{1}$$

$$\text{Subject to :} \quad \sum_j l_j s_j \leq L \tag{2}$$

$$s_j Occ_{ij} \leq c_i, \quad \forall i, j \tag{3}$$

$$\sum_j s_j Occ_{ij} \geq c_i, \quad \forall i \tag{4}$$

$$c_i \in \{0, 1\} \quad \forall i$$

$$s_j \in \{0, 1\} \quad \forall j$$

We try to maximize the weight of the concepts (1) in the selected sentences, while avoiding repetition of those concepts (3, 4) and staying under the maximum number of words allowed for the summary (2).

This formulation is the one that Gillick and Favre (2009) used in their original approach. We add some more constraint, to take advantage of our previous clustering step and to limit the risks we take during sentence fusion.

First, we adapt the ILP formulation so that the optimization procedure decides which fused alternatives to pick. More formally, let g_k be a cluster of sentences that corresponds to the set of similar sentences plus their fused alternatives. We add the following constraint to our ILP formulation:

$$\sum_{j \in g_k} s_j \leq 1, \quad \forall g_k \tag{5}$$

This constraint encodes that only one sentence per cluster, fused or not, can appear in the summary. This is similar to the constraint introduced in (Gillick et al., 2009; Li et al., 2013) for limiting multiple sentence compression variants to appear in the summary.

Fused sentences are generally shorter and more informative than the sentences from which they are created. But as they are generated automatically, they are likely to be ungrammatical. To minimize the risk of degrading the linguistic quality of the summary, sentence fusions should be selected only if there is no not-fused sentence of equal length and importance. Let e be an

extracted sentence and f a fusion, we add the following constraint to our ILP formulation:

$$s_f \leq 0, \quad \forall f$$
$$\text{if } \exists\, e \mid \left(l_f = l_e \wedge Occ_{ie} = Occ_{if}, \;\; \forall i\right) \quad (6)$$

Here, we ensure that e and f are of equal length and contain the same concepts. In those cases, the fused sentence is discarded. This constraint mainly reduces the verb form errors introduced by the sentence fusion component. An example of fusion (topic D30047) that is discarded by our model along with its corresponding extracted sentence is given below:

(f) Space officials from 16 nations taking part in the project cheered as the rocket **soaring** into the cloudy sky.

(e) Space officials from 16 nations taking part in the project cheered as the rocket **soared** into the cloudy sky.

The ILP problem is then solved exactly using an off-the-shelf ILP solver[8]. Summaries are generated by assembling the optimally selected sentences.

[8] We use Gurobi, `http://www.gurobi.com`

Chapter 5

Experiments

In this chapter, we present how we evaluated our system.

First, we describe the dataset we used and the two evaluation measures we chose to consider: the automatic ROUGE measure to evaluate the overall quality of the generated summaries and the manual evaluation of grammaticality to assess for the possibility of ill-formed sentences that occur during the sentence fusion step.

Then, we show that our system obtains ROUGE results that are higher than a similar state-of-the-art system based on ILP and sentence compression.

Finally, we show that sentence fusion leads to grammaticality scores similar to previously reported results, while originating from smaller sentence clusters.

5.1 Data and Evaluation measures

We use the DUC 2004 dataset to evaluate our summarization system. DUC is a series of summarization evaluations that have been conducted by the National Institute of Standards and Technology from 2001 to 2007. From 2008 and on, the conference has been replaced by the Text Analysis Conference, which continues to focus on automatic summarization. DUC 2004 was however the last track that focused on "classical" multi-document summarization, next tracks being focused on update-summarization (summarizing what's new from past and current data), query-focused summarization (where user-like queries guide the summary) or other specialized summarization tasks (the last one being biomedical summarization).

The DUC 2004 dataset is made of 50 sets of documents, each composed of 10 newswire articles about a given topic from the Associated Press and The New York Times that were published between 1998 and 2000. The documents describe events with an international focus,

from election results and political crisis to cooperation for space station building and nobel prize awarding.

Given a set of documents, the task consists in generating a summary of maximum 100 words. Four human-authored reference summaries are provided for each set, and are used for automatic evaluation.

Four configurations of our summarization system are examined, depending on the similarity measure used for clustering (LexSim or SemSim, c.f. Chapter 4.2) and the sentence fusion method (Fusion for Filippova (2010)'s method or Reranking for Boudin and Morin (2013)'s method, c.f. Chapter 4.3). As our main objective is to propose an approach that does not need heavy-resources, we are particularly interested in evaluating the actual benefits of using a semantic similarity measure compared to a lexical similarity measure.

We compare our system against the ICSISumm system (Gillick et al., 2009) which is the best-performing summarization system available through Sumrepo, a repository of summaries generated by state-of-the-art systems on the DUC 2004 dataset (Hong et al., 2014). The ICSISumm system is based on the concept-based ILP framework to summarization and uses sentence compression to generate higher quality summaries. Compressed alternatives for each sentence are created by manipulating its parse tree, extracted with the Berkeley parser (Petrov and Klein, 2007).

To gain a better insight into the benefits of sentence fusion, we also report a baseline system consisting of our method without the sentence clustering and fusion steps. We used the same parameters for the ILP formulation, as well as the same preprocessing treatments. This baseline is thus an implementation of the concept-based ILP framework to summarization (Gillick and Favre, 2009), without sentence compression.

As our system is abstractive, it faces the risk to generate incorrect sentences. To account for this liability, we conduct two different evaluation protocols, one that evaluates the informativity of the summary via the automatic ROUGE measure (Lin, 2004) and another one that manually assesses for the grammaticality of the generated sentences.

The ROUGE measure is based on n-gram recall between the generated summary and the human-written gold abstracts. ROUGE-2 for instance, corresponds to the following formula:

$$\text{ROUGE-2} = \frac{\sum_R \sum_{b_i \in R} Count_{match}(b_i)}{\sum_R \sum_{b_i \in R} Count(b_i)}$$

Where R is the set of reference summaries, $b_i \in R$ are the bigrams in the current reference summary, $Count_{match}(b_i)$ is the number of bigrams that are both in the candidate summary and

the current reference summary and $Count(b_i)$ is the number of bigrams in the current reference summary.

We use ROUGE-1 (unigrams recall), ROUGE-2 (bigrams recall), and ROUGE-SU4 (skip-bigrams with a maximum range of 4 plus unigrams). Skip-bigrams calculated in ROUGE-SU4 considers bigrams that can be separated by up to 4 tokens.

In our study, we follow the standard ROUGE parameters suggested by Owczarzak et al. (2012)[1], where stemming and stopwords not removed provides the best agreement with manual evaluations.

As for manual evaluation, we follow (Barzilay and McKeown, 2005; Filippova, 2010; Boudin and Morin, 2013) and evaluate the grammaticality of the fused sentences on a 3-points scale: perfect (2 pts), if the fusion is a complete grammatical sentence; almost (1 pt) if it requires minor editing, e.g. one mistake in articles, agreement or punctuation; ungrammatical (0 pts), if it is none of the above. Two human raters were asked to assess the grammaticality of the fused-sentences that appear in the summaries. An example of almost grammatical sentence (the possessive *'s* is to be removed, topic D30028) is given below.

(F) Syria**'s** plans to build dams on the Euphrates river.

The class of ungrammatical sentences is composed of sentences that do not make sense, that miss important components (like a verbal clause or a subject), or that are purely ungrammatical. Two examples (topics D31033 and D30047), illustrating the wide variety in quality of the sentences belonging to this class, are given below.

(F) Microsoft's Windows, the operating system that controls about 90 percent of personal computers sold today.

(F) Russia's Zarya and attach the two units.

The first of the two sentences lacks a verbal clause, but is completely understandable and exposes a fact that can be of interest in a summary. The second one is totally ungrammatical and does not convey any understandable information. These two examples demonstrate that averaging grammaticality scores does not give much information on the overall quality of the fused sentences. However, it allows us to compare ourselves to previous works that use this scoring method.

[1] We use the following command: ./ROUGE-1.5.5.pl -n 4 -2 -4 -u -m -a -l 100 -x -c 95 -r 1000 -f A -p 0.5 -t 0

5.2 Automatic Evaluation

ROUGE scores are reported in Table 5.1. Overall, we observe that our system consistently outperforms the baseline and ISCISumm on the three ROUGE measures we experimented with. When using the LexSim / Reranking configuration, our system significantly outperforms the baseline, indicating that fusing input sentences does lead to better summaries.

It may be surprising to see that a purely lexical similarity measure performs just as well as a semantic similarity measure. This is mainly due to the fact that the fusion approach we use relies on lexical repetition to mix phrases from source sentences. While a semantic similarity measure allows to cluster sentences that do share a similar meaning, the synonyms they hold won't be considered as mergeable terms during fusion, and won't lead to any new fused sentence. This aspect is discussed more thoroughly in Chapter 6.3.

As noted by Hong et al. (2014), it is hard to obtain significant differences between state-of-the-art systems. Despite having better overall scores, the DUC 2004 dataset (50 sets of documents) is not large enough for our system to achieve significant improvements over ICSISumm. Nevertheless, these results show that word graph-based sentence fusion leads to results which are at least comparable to sentence compression, a method that does rely on heavier resources (e.g. syntactic parser).

System	ROUGE-1	ROUGE-2	ROUGE-SU4
Baseline	38.0	9.57	12.95
ICSISumm	38.4	9.79	12.94
LexSim / Fusion	38.6	9.91	13.15
LexSim / Reranking	**38.8**†	**10.07**†	13.32
SemSim / Fusion	**38.8**†	9.81	**13.35**
Semsim / Reranking	38.7	9.88	13.26

Table 5.1: ROUGE-1, ROUGE-2 and ROUGE-SU4 results for the baseline, ISCISumm and our system using the four different configurations of sentence clustering and sentence fusion († indicates significance at the 0.05 level using Student's t-test for our system versus the Baseline).

5.3 Manual Grammaticality Evaluation

Table 5.2 reports results of the manual evaluation on grammaticality of the fused sentences. Sentences that are not modified are not considered during this evaluation.

The configuration that gives the best ROUGE scores (LexSim / Reranking) also obtains the best manual ratings with 64% of the generated fusions judged as perfectly grammatical. Overall,

System	Grammaticality			Avg.	κ
	0	1	2		
LexSim / Fusion	27%	20%	53%	1.25	0.58
LexSim / Reranking	23%	13%	**64%**	**1.40**	0.62
SemSim / Fusion	23%	18%	59%	1.36	0.48
SemSim / Reranking	27%	12%	61%	1.34	0.53

Table 5.2: Distribution over possible manual ratings for grammaticality, expressed on a scale of 0 to 2. The average ratings over all fusions (Avg.) and the inter-annotator agreement (κ) are also reported.

we obtain an average grammaticality score of 1.34, ranging from 1.25 to 1.40 depending on the annotators and configurations.

These scores are almost as high as those reported in previous works (Filippova, 2010; Boudin and Morin, 2013). The slight decrease in grammaticality is mainly due to the fact that the clusters of similar sentences that we construct are smaller than those used in other works, thus leading to less repetition and allowing to chain infrequent terms, leading to ungrammatical sentences. However, our top configuration manages to achieve grammaticality scores identical to those reported by Filippova (2010).

As for the ratio of perfect sentences, we also obtain similar results with an average 59% of sentences that do not contain any error. Since the sentences belonging to class 1 only contain small mistakes, they are still perfectly intelligible. Overall, 75% of all sentences (that is classes 2 and 1) are comprehensible for the reader. The kappa between the two annotators denotes a moderate agreement, ranging from 0.48 to 0.62.

While we can compare our results with previous work on sentence fusion, it is difficult to compare with compression grammaticality, as results are often not reported, and vary widely between the methods that are used.

As an example, Li et al. (2013) use ILP and sentence compression for summarization and generate up to 200 compression variants per sentence and seem likely to generate incorrect sentences. Other systems, based on manually tailored rules like the first approach explored by Gillick and Favre (2009) are supposedly less error-prone, but have been evaluated[2] on a 5-point scale that takes into account not only grammaticality, but overall readability, including focus, referential clarity and non-redundancy.

Figure 5.1 illustrates the differences between compression and fusion by showing a summary created from the ICSISumm system as well as a summary created by our system. The

[2] they follow the TAC 2008 track guidelines

summarized documents talked about a trial for Yugoslav war crimes that occurred in 1992.

We can see that both methods output a very similar first sentence, that exposes the main reported fact. However, compression fails to extract the name of the main protagonist while fusion has been able to report it through sentence mixing. Fusion also reports more information, focusing on the impact of the event (a likely appeal and the supposed influence of other countries) whereas compression extracted information about details of the trial.

Compression :
The Yugoslav war crimes tribunal Monday acquitted a Muslim military commander of war crimes against Bosnian Serb prisoners in 1992, but convicted three underlings in the first U. N. case dealing with anti-Serb atrocities.
The tribunal, set up by the Security Council in 1993, has convicted only one other person following a trial, Bosnian Serb Dusan Tadic, who was sentenced in May 1997 to 20 years for killing and torturing Muslims in 1992.
During the trial, the tribunal's longest to date involving 122 witnesses, survivors described the campaign of terror unleashed against Serbs in the area.

Fusion :
The Yugoslav war crimes tribunal cleared Zejnil Delalic, a Muslim, of responsibility for war crimes committed against Serb captives at a Bosnian government-run prison camp under his command.
The tribunal, set up by the Security Council in 1993, has convicted only one other person following a trial Bosnian Serb Dusan Tadic, who was sentenced in May 1997 to 20 years for killing and torturing Muslims in 1992.
Prosecutor Grant Niemann said he would likely appeal Delalic's acquittal.
He said the U.N. court was influenced by powerful countries that dominate the international body.
He faces a maximum sentence of life imprisonment.

Figure 5.1: Two generated summaries that illustrate the benefits of fusion over compression

Chapter 6

Discussion

In this chapter, we discuss three major aspects of our system.

First, we expose some of the strengths and weaknesses of our approach, including the benefits of merging the information that is originally spread between sentences and the dangers that also lie in sentence fusion.

Then, we demonstrate how small parameter modifications can lead to completely different summaries, emphasizing the results demonstrated by Hong et al. (2014) on state-of-the-art summarization systems.

Finally, we describe some major inherent flows of state-of-the-art approaches and how they rely more and more on specialized linguistic resources, making them both language dependent and domain-dependent, a characteristic that makes them hard to use in generic applications (i.e. search engines).

6.1 Strengths and Weaknesses of Sentence Fusion

Word graph-based sentence fusion often leads to shorter alternatives of the original sentences. This shortening happens by starting on a sentence and quickly jumping to the end of another one.

While this might occasionally lead to incomplete sentences, it most of the time achieves a compression that would be very difficult to obtain via manually tailored rules, as demonstrated in the example below (topic D30055):

(1) *The Communist Refoundation Party provoked the* **crisis** by withdrawing its support over the weekend and saying it would not vote for his deficit-cutting **1999** *budget.*

(2) Prodi's far-left ally, the Communist Refoundation Party, provoked the **crisis** *when it withdrew support for the government over the* **1999** deficit-cutting budget, which it said did not do enough to stimulate job creation.

(F) *The Communist Refoundation Party provoked the* **crisis** *when it withdrew support for the government over the* **1999** *budget.*

In this example, every piece of information can be extracted from sentence 2. However, one would need to remove three components from the original sentence, pruning 45% of the words, including sentence starting and ending, a risky operation. Instead, the fusion manages to be shorter than any of the two sentences, while containing all major information.

Grammaticality is very often an issue in abstractive summarization, and our system is no exception, as exposed in section 5.3. But sentence fusion can sometimes lead to more subtile errors, such as counter senses. The example below shows one occurrence of such an error in our generated summaries. While the fused sentence is perfectly grammatical, it does not convey the correct information, and can mislead a reader.

(1) **Cardoso** wants to impose tough **measures** *that would slash government* **spending** and impose new taxes to try to halt the slide in Brazil's economy and restore investor confidence.

(2) To halt Brazil's slide toward recession, **Cardoso** *is preparing austerity* **measures** including **spending** *cuts, tax hikes and lower interest rates.*

(F) **Cardoso** *is preparing austerity* **measures** *that would slash government* **spending** *cuts, tax hikes and lower interest rates.*

These counter senses could be critical in other domains such as medical documents. The summary could give the reader wrong information if the full text is not consulted.

6.2 Minor Parameter Modification Implies Great Changes

Hong et al. (2014) showed that different systems that performed similar results didn't actually output similar summaries (i.e. the set of sentences extracted was different and yet lead to similar scores). The experiments we conduct rely on very similar approaches though, the only two variations being the similarity measure we use for clustering and the sentence fusion method. A natural question arises: are the summaries generated using different configurations similar?

Table 6.1 shows the number of identical sentences obtained by each pair of configurations.

While we obtain a higher overlap than the one exposed by Hong et al. (2014), we see that the summaries generated using the four configurations are still considerably different. Yet the generated summaries are quite dissimilar, with at most 60% of common sentences between the outputs of our different configurations. As a matter of fact, our two most dissimilar configurations, LexSim / Reranking and Semsim / Fusion only share 35% of their sentences and output only one identical summary out of the 50 in the dataset.

System	LexSim / Fusion	LexSim / Reranking	SemSim / Fusion	SemSim / Reranking
LexSim / Fusion	271	160	122	100
LexSim / Reranking	-	273	96	120
SemSim / Fusion	-	-	291	148
SemSim / Reranking	-	-	-	290

Table 6.1: Number of identical sentences in output summaries. Diagonal corresponds to the total number of sentences generated by a system

This wide diversity is due to two factors. First, the smallest modification can lead the ILP framework into choosing a entirely different set of sentences as different combinations may have very close scores while being composed of completely different sentences. Secondly, there exists many possible fusions to choose from, adding to the variety of sentences that can be included.

6.3 Of the Need for Resources

Automatic summarization interest in the scientific community has been largely increased by the advent of the Internet. However, most existing approaches use heavy resources in order to be competitive with other systems. While the use of linguistic resources seems to be mandatory in many Natural Language Processsing domains, our experiment showed that adding semantic similarity did not increase the efficiency of our system.

As automatic summarization evaluation is costly and time consuming, there exist very few datasets with manual summaries. To alleviate this issue, previous research works usually rely on documents that come naturally with a summary, such as research papers that come with a manually-written abstract. However, this evaluation process has many inconvenients, from the size of the reference abstracts that can vary a lot, to the bias of the author and the nonexistence of multiple reference summaries.

The lack of corpora leads most authors to evaluate their systems during conference workshops and tracks, narrowing their possibilities both in terms of types of summarization (the last multi-document summarization track was DUC 2004, the following conferences being focused on update summarization or other specific summarization tasks), and text genre (almost always extracted from English newswire).

Since almost every corpora have the same language and genre, systems have been tweaked to obtain higher scores on these tasks, achieving ever increasing scores while using more and more specialized resources. As an example, many recent works try to improve summarization tools by using machine learning on previous DUC campaigns (Wong et al., 2008; Lin and Bilmes, 2012; Almeida and Martins, 2013), lowering the ability to summarize different genres and languages.

Despite having systems that do offer quite readable and informative summaries during conference tracks, there is almost no automatic summarization on the Internet, the place it is most needed. Search engines extract one or two sentences containing the key terms that are searched, and do not try to offer a comprehensive summary of the entries they provide to their users. Websites like `metacritic.com` give a list of human-written critics and provide only the mean score of the movie/music/game it is dealing with. This absence is characteristic of the overspecialization of state-of-the art systems.

As the web is multilingual and unspecialized, there is a huge need for genre-agnostic and language-agnostic methods. We believe our approach takes a first step in this direction by proving that there exist competitive methods that do not require heavy resources and can thus be easily applied to different genres and languages.

Chapter 7

Conclusion and Perspectives

In this report, we presented our system based on ILP and sentence fusion and showed that it achieves state-of-the-art results on news summarization. This system does not rely on heavy resources or on manually tailored rules and is thought to be domain-independent, as well as language-independent.

This is a first step toward building a robust system that could leverage the overload of information that we witness on the Internet. Abstractive summarization presence on the web would deeply change how information retrieval is applied, going from document retrieval and usage of knowledge databases to document clustering and question answering. By typing a search such as ”Ukraine conflict 2014 -500words” you would have access to a short summary that would contain most useful information while providing links to the documents from which each information have been extracted, allowing you to pursue your information search through the full documents if needed.

As for possible improvements of our system, one of its virtues is that it is composed of independent parts that can be separately enhanced. The bigram weighting could be enhanced by taking into account a finer measure than its document frequency. Approaches that predict word importance could be adapted to fit this purpose (Hong and Nenkova, 2014). However, these methods do rely on machine learning and may be language-dependant.

Using more elaborate measures for sentence clustering could also lead to an improvement. However, as showed in our experiments, sentence fusion needs repetition across the sentences to take place, using a similarity measure that does not rely enough on the surface form of the words may compromise this step. Moreover, the most efficient existing systems for semantic sentence similarity rely on many resources such as training resources (e.g. Wikipedia), WordNet, thesauri, lemmatizers and Named Entities Recognizers, as showed by the recent SEMEVAL tracks (Agirre et al., 2012; Agirre et al., 2013).

As the ILP sentence selection framework is one of the main focuses in the summarization community, many discussions take place to determine how to best represent concepts in a set of documents and how to weight more efficiently bigrams. These works may lead to big improvements and strenghen the position of ILP as one of the main candidates for efficient automatic summarization.

Sentence fusion is supposed to be usable in many languages, and has successfully been used in French and Spanish (Boudin and Morin, 2013; Filippova, 2010). Experiments for the full framework in alternative languages and domains should be relatively easy to consider and would prove the wide usability of this method. As sentence fusion is relatively new, we also expect improvements of the method that may lead to less ungrammatical sentences.

We published the source code on GitHub and are eager to see people participating and improving on this system.

References

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, Montréal, Canada, 7-8 June. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *sem 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

Miguel Almeida and André FT Martins. 2013. Fast and robust compressive summarization with dual decomposition and multi-task learning. In *ACL (1)*, pages 196–206.

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

Regina Barzilay, Michael Elhadad, et al. 1997. Using lexical chains for text summarization. In *Proceedings of the ACL workshop on intelligent scalable text summarization*, volume 17, pages 10–17. Madrid, Spain;.

Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 481–490, Portland, Oregon, USA, June. Association for Computational Linguistics.

Steven Bird. 2006. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 69–72. Association for Computational Linguistics.

Harold Borko and Charles L Bernier. 1975. Abstracting concepts and methods.

Florian Boudin and Emmanuel Morin. 2013. Keyphrase extraction for n-best reranking in multi-sentence compression. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 298–305, Atlanta, Georgia, June. Association for Computational Linguistics.

F. Boudin, S. Huet, and J.M. Torres-Moreno. 2011. A graph-based approach to cross-language multi-document summarization. *Polibits*, 43:113–118.

Florian Boudin. 2008. *Exploration d'approches statistiques pour le résumé automatique de texte*. Ph.D. thesis, Laboratoire Informatique d'Avignon – Université d'Avignon.

Hal Daume III and Daniel Marcu. 2004. Generic sentence fusion is an ill-defined summarization task. In Stan Szpakowicz Marie-Francine Moens, editor, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 96–103, Barcelona, Spain, July. Association for Computational Linguistics.

Harold Charles Daume, III. 2006. *Practical Structured Learning Techniques for Natural Language Processing*. Ph.D. thesis, Los Angeles, CA, USA. AAI3337548.

B. Favre, F. Béchet, P. Bellot, F. Boudin, M. El-Beze, L. Gillard, G. Lapalme, and J.M. Torres-Moreno. 2006. The

lia-thales summarization system at duc-2006. In *Document Understanding Conference (DUC)*.

Katja Filippova and Michael Strube. 2009. Tree linearization in english: Improving language model based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 225–228. Association for Computational Linguistics.

Katja Filippova. 2010. Multi-sentence compression: Finding shortest paths in word graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 322–330. Association for Computational Linguistics.

Michel Galley and Kathleen McKeown. 2007. Lexicalized Markov grammars for sentence compression. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 180–187, Rochester, New York, April. Association for Computational Linguistics.

Pierre-Etienne Genest and Guy Lapalme. 2012. Fully abstractive approach to guided summarization. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 354–358. Association for Computational Linguistics.

Pierre-Etienne Genest, Guy Lapalme, and Mehdi Yousfi-Monod. 2009. Hextac: the creation of a manual extractive run. In *Proceedings of the Second Text Analysis Conference*, Gaithersburg, Maryland, USA, Nov.

Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pages 10–18, Boulder, Colorado, June. Association for Computational Linguistics.

Daniel Gillick, Benoit Favre, Dilek Hakkani-Tur, Berndt Bohnet, Yang Liu, and Shasha Xie. 2009. The ICSI/UTD Summarization System at TAC 2009. In *Proc. of the Text Analysis Conference workshop, Gaithersburg, MD (USA)*.

Jade Goldstein, Vibhu Mittal, Jaime Carbonell, and Mark Kantrowitz. 2000. Multi-document summarization by sentence extraction. In *Proceedings of the 2000 NAACL-ANLPWorkshop on Automatic summarization-Volume 4*, pages 40–48. Association for Computational Linguistics.

Kai Hong and Ani Nenkova. 2014. Improving the estimation of word importance for news multi-document summarization. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 712–721, Gothenburg, Sweden, April. Association for Computational Linguistics.

Kai Hong, John Conroy, Benoit Favre, Alex Kulesza, Hui Lin, and Ani Nenkova. 2014. A repository of state of the art and competitive baseline summaries for generic news summarization. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1608–1616, Reykjavik, Iceland, May. European Language Resources Association (ELRA). ACL Anthology Identifier: L14-1070.

Kevin Knight and Daniel Marcu. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107.

Peng Li, Yinglin Wang, Wei Gao, and Jing Jiang. 2011. Generating aspect-oriented multi-document summarization with event-aspect model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1137–1146. Association for Computational Linguistics.

Chen Li, Fei Liu, Fuliang Weng, and Yang Liu. 2013. Document summarization via guided sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 490–500, Seattle, Washington, USA, October. Association for Computational Linguistics.

Hui Lin and Jeff A Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. *arXiv preprint arXiv:1210.4871*.

Chin-Yew Lin. 1999. Training a selection function for extraction. In *Proceedings of the Eighth International Conference on Information and Knowledge Management*, CIKM '99, pages 55–62, New York, NY, USA. ACM.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81.

Hans P Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165.

Inderjeet Mani. 2001. *Automatic summarization*, volume 3. John Benjamins Publishing.

André FT Martins and Noah A Smith. 2009. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, pages 1–9. Association for Computational Linguistics.

Ryan McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *Advances in Information Retrieval*, pages 557–564. Springer.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into texts. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 404–411, Barcelona, Spain, July. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Karolina Owczarzak, John M. Conroy, Hoa Trang Dang, and Ani Nenkova. 2012. An assessment of the accuracy of automatic evaluation in summarization. In *Proceedings of Workshop on Evaluation Metrics and System Comparison for Automatic Summarization*, pages 1–9, Montréal, Canada, June. Association for Computational Linguistics.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 404–411, Rochester, New York, April. Association for Computational Linguistics.

Xian Qian and Yang Liu. 2013. Fast joint compression and summarization via graph cuts. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1492–1502, Seattle, Washington, USA, October. Association for Computational Linguistics.

Dragomir R Radev, Hongyan Jing, and Malgorzata Budzikowska. 2000. Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *Proceedings of the 2000 NAACL-ANLP Workshop on Automatic Summarization*, pages 21–30. Association for Computational Linguistics.

GJ Rath, A Resnick, and TR Savage. 1961. The formation of abstracts by the selection of sentences. part i. sentence selection by men and machines. *American Documentation*, 12(2):139–141.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May. ELRA. `http://is.muni.cz/publication/884893/en`.

Simone Teufel. 1997. Sentence extraction as a classification task. In *Intelligent Scalable Text Summarization*, pages 58–65.

Kapil Thadani and Kathleen McKeown. 2013. Supervised sentence fusion with single-stage inference. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1410–1418, Nagoya, Japan, October. Asian Federation of Natural Language Processing.

Michael White, Tanya Korelsky, Claire Cardie, Vincent Ng, David Pierce, and Kiri Wagstaff. 2001. Multidocument summarization via information extraction. In *Proceedings of the first international conference on Human language technology research*, pages 1–7. Association for Computational Linguistics.

Kam-Fai Wong, Mingli Wu, and Wenjie Li. 2008. Extractive summarization using supervised and semi-supervised learning. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 985–992. Association for Computational Linguistics.

Kristian Woodsend and Mirella Lapata. 2012. Multiple aspect summarization using integer linear programming. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 233–243. Association for Computational Linguistics.

Appendix A

System's Details

A.1 Usage Example

You can install potara (the name of our opensourced system) by typing those commands:

```
pip install potara
```

Then, to use potara in a Python script, just import it. Here is a small example to summarize 4 documents:

```
from potara import summarizer
from potara import document

doc1 = document.Document('file1.txt')
doc2 = document.Document('file2.txt')
doc3 = document.Document('file3.txt')
doc4 = document.Document('file4.txt')

s = summarizer.Summarizer(minbigramcount=2)
s.setDocuments([doc1, doc2, doc3, doc4])

summary = s.summarize(wordlimit=100)

print(summary)
```

A.2 System Parameters

Our systems relies on parameters that are to be set via its API. The default parameters have been tested for summarizing news articles in English. Our dataset came in sets of 10 articles, and the system may perform poorly on smaller or bigger sets if the default parameters are not modified. Hopefully, it is very easy to modify the details of the method as we exposed as much methods and functions as possible.

As an example, our summarizer relies on similarity measures to cluster sentences. We implemented two different similarity measures and cosine is the default. However, the user can initialize a summarizer with a different customized similarity measure, as the summarizer can take as parameter an arbitrary similarity measure. Table A.1 describes the different parameters of our system.

Parameter name	Usage	Default Value
	Summarizer	
minbigramcount	bigrams occurring less than this value will be discarded	4
similaritymeasure	similarity used for sentence clustering	cosine
wordlimit	limits the size of the summary to this number of words	100
	Document	
stopwords	filter those words when comparing sentences	nltk stopwords list - English
sentTokenizer	Split sentences during document preprocessing	nltk Punkt - English
wordTokenizer	Split a sentence into tokens, putting punctuation out of words	nltk default tokenizer - English
postagger	POStag a sentence during preprocessing	Stanford POStagger - English
stemmer	Stems (shortens) tokens of a sentence during preprocessing	Snowball Stemmer - English

Table A.1: Parameters and default values for our system