Summarization by Latent Dirichlet Allocation: Superior Sentence Extraction through Topic Modeling

Kenton W. Murray

April 17, 2009

A Senior Thesis submitted to the Department of Computer Science in partial fulfillment of the requirements for the degree of a Bachelors of Science in Engineering

Princeton University
Princeton, NJ

Pledge

This Thesis represents my own work in accordance with University Regulations.

Kenton W. Murray

Acknowledgements

I would like to thank my family for all the support and encouragement that they have given me over the years along with continual focus on my education through countless countries, states, and school districts. It looks like it finally paid off.

I would also like to thank my advisor David Blei for his help throughout the course of the last year in developing this thesis. His ideas permeate this entire document. Additionally, I would like to thank Andrea LaPaugh for agreeing to take the time to read and grade this document.

To all of my friends from the past four years: it has been so much fun and I am already looking forward to Reunions.

Contents

1 Introduction **1**
- 1.1 Natural Language Processing . . . 2
- 1.2 What is a Summary? . . . 4
- 1.3 About this Paper . . . 6

2 Related Works **7**
- 2.1 Historically Influential Works . . . 7
- 2.2 Modern Day Approaches to Summarization . . . 9

3 Latent Dirichlet Allocation **12**
- 3.1 Overview of LDA . . . 13
- 3.2 Applications of LDA . . . 14
- 3.3 Derivation of LDA . . . 15
- 3.4 Motivation for Summarization . . . 17

4 Summarization by Latent Dirichlet Allocation **19**
- 4.1 Overview . . . 19
- 4.2 Obtaining a Wikipedia Corpus . . . 20
- 4.3 Calculating Topics . . . 21
- 4.4 Zipfian Distribution of Words . . . 22

5 Rival Methods **25**
- 5.1 Overview . . . 26
- 5.2 NNMF . . . 27
  - 5.2.1 Overview of NMF . . . 28
  - 5.2.2 NMF Algorithm . . . 28
  - 5.2.3 Deficiencies . . . 30
  - 5.2.4 Use in My Work . . . 30
- 5.3 TextRank . . . 31
  - 5.3.1 Influences on TextRank . . . 31
  - 5.3.2 TextRank Algorithm . . . 32
  - 5.3.3 Graph Representations of Documents in TextRank . . . 33
  - 5.3.4 Proof of Single Iteration Convergence . . . 34

6 Summary Evaluation **37**
6.1 Document Understanding Conference 38
6.2 ROUGE 39
6.3 Mechanical Turk 41

7 Results **44**
7.1 Summary Examples 44
7.2 Statistics 47

8 Conclusion **51**
8.1 Future Work 51
8.2 Accomplishments 53

Bibliography **58**

A List of Stop Words **59**

B Mechanical Turk Question Template **60**

C Z-DNA Source Text **62**

D List of Summarized Wikipedia Articles **65**

List of Tables

7.1 Overall Responses to Mechanical Turk Evaluation 47
7.2 Responses to Mechanical Turk Evaluation taking more than 10 seconds . . 48
7.3 χ^2 values of Individual Comparisons 49

A.1 List of Stop Words 59

D.1 List of Summarized Articles 65

List of Figures

B.1 **Mechanical Turk Question Template** 61

Abstract

Latent Dirichlet allocation, or LDA, is a successful, generative, probabilistic model of text corpora that has performed well in many tasks in many areas of Natural Language Processing. Despite being perfectly suited for Automatic Summarization tasks, it has never been applied to them. In this paper, I introduce Summarization by LDA, or SLDA, which better models the subtopics of a document leading to more pertinent, relevant, and concise summaries than other summarization methods. This new approach is competitive with the leading methods in the field and even outperforms them in many aspects. In addition to SLDA, I introduce a novel, paradigm-shifting, evaluation technique of summarization that does not rely on gold-standards. It overcomes many of the challenges imposed by inherent disagreements amongst people of what a good summary is by evaluating over large numbers of people using the commercial service, Mechanical Turk. Overall, this paper lays the ground work for transforming the conventions of the Automatic Summarization field by challenging many definitions.

Chapter 1

Introduction

With the explosion in size of the Internet and the associated increased amount of information accessible with ease, the average person has both the ability to access and add to the knowledge available. One of the most prominent examples of this is the collaborative website Wikipedia [2]. Wikipedia is a free online encyclopedia where all of the content is submitted by the people reading and accessing the website. Anyone with an internet connection has the ability to do so, and consequently, the amount of information contained in the articles is staggering. As it is not a verified or peer reviewed source, the information is not guaranteed to be valid and many users often look at it for a brief overview of a topic or as a starting point for learning about a new concept. Often, the amount of information about a certain topic on a page is much more than is needed by the user so the ability to quickly synthesize and summarize what is contained within is frequently desired.

The desire and need for textual document summaries extends beyond simple articles on Wikipedia. There are many different applications for automatically generated summaries and in this thesis, I present a new approach to tackling this problem. I present a method for generating extractive summaries that is focused on modeling topics within a document that generates field leading results along with introducing a new way to judge the field.

This thesis will take a novel approach to summarization that lays the ground work for transforming the field of Automatic Summarization. Demonstrating the effectiveness of my

methods on a corpus comprised of Wikipedia articles, I will show how these methods are extensible to any text corpus available. My method, Summarization by Latent Dirichlet allocation or SLDA, generates superior summaries by modeling the topics within a document and selecting information that has less repetition. Furthermore, it obfuscates the basic distinction between single-document and multi-document summarization that classifies summarization — leading to a new way of thinking about the field.

In addition, I present a revolutionary new way to evaluate automatic summarization that is unlike any other technique currently used. My methods are evaluated using a commercial service, Mechanical Turk, that distributes basic tasks to large numbers of people. This helps solve some of the deficiencies currently a part of summarization evaluation. By distributing the evaluation to multiple people, I am lessening the bias that a single method or person can have on the overall evaluation and creating a fairer assessment of a subjective task.

Though my overall results are significant and as good as leading methods within the field of summarization, the biggest impact that this research will have is by defining new ways that summarization can be viewed. Classical distinctions such as single-document vs. multi-document summarization are more difficult to discern in this system and I also outline ways to blur the line between abstractive and extractive summarization in simple extensions of my research. Furthermore, by introducing a paradigm-shifting method of evaluation, old definitions of what constituted a good summary will also be called into question. This research brings up many new questions that have not been asked in the field, and will not likely find answers any time soon.

1.1 Natural Language Processing

Natural Language Processing, or NLP, is a sub-field of Artificial Intelligence that attempts to model and understand human language in a machine context. It is an incredibly challenging field with many subfields, unanswered questions, and problems with no definitive "right" solutions. It is a growing field that overlaps with many other areas of Computer Science,

Robotics, as well as many other academic disciplines. For most of the problems, getting the correct answer is tantamount to passing the Turing Test.

Whether or not people realize it, almost everyone is influenced by NLP research in his or her day-to-day life. One of the most developed sub-fields of NLP, Information Retrieval, is most recognizable through commercial search engines such as Google, Yahoo, and Live Search. Other subfields may not currently be as prominent, but the potential to impact our lives is just as possible as progress is made in researching those areas. Other notable subfields of NLP include, Machine Translation, Text Categorization, Clustering, and Summarization. It has already been demonstrated that applying successful methods and algorithms in one area of NLP to another has much potential improve and further work in it, such as TextRank described in more detail later [26].

Natural Language Processing is a difficult task because of the inherent ambiguities of human language that lead to multiple answers for any problem. Even very developed areas of NLP, such as Information Retrieval, will never perfectly answer every case. When querying for a term such as "jaguar" there are multiple possible things a user could be looking for: the cat, a car company, or even a school mascot. Ambiguity in words also results in anaphoric relations where one expression refers to another. This is quite difficult for a computer program to pick up on. Additionally, collocations also pose a significant challenge to NLP work. A collocation is a singular idea expressed in multiple words such as "Nassau Hall" that introduces significant complexity into modeling natural language. There are many other examples and challenges facing NLP work that entire books could be written about.

This thesis will be focused upon summarization, though many of the methods are either influenced from other areas of Natural Language Processing, or could successfully be applied to them. Regardless, many of the problems facing other areas of NLP are just as prevalent in summarization and will never be completely resolved.

1.2 What is a Summary?

In order to motivate our discussion of automatic summarization, it is first important to discuss what exactly automatic summarization is. Simply put, summarization condenses the information of an input source into a smaller format. Summarization is actually a very broad idea. Newspaper headlines are a summary of the following article. An abstract of a paper is a summary of the research. A movie trailer is a summary of the entire film. For the purposes of this thesis, I will focus on text summarization. Perhaps the best definition of text summarization is given by Karen Sparck Jones in the first book written to capture the trends and current state of the field, "Advances in Automatic Text Summarization" [24]. She 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" [13].

Even with limiting summarization to text summarization, it is still a very general concept. The length of the summary along with its intended purpose will greatly influence how the summary is created. A five word newspaper headline is going to be sufficiently different than a book report on a piece of English literature. In this paper, I will focus on short summaries that intend to capture the essence of a document — something like an abstract to a paper rather than a title. In other words, summaries around one hundred to two hundred words or approximately ten sentences. The intended purpose will be to give a brief good understanding of an input text for a reader who, either does not want to read the entire text, or wants to know if the text is pertinent to what he or she is looking for before spending the time to read it.

There are three main ways to classify a text summarization method, but with the improvements in computing and increasing access to information, I think that these classifications will become more difficult to discern in the next few years. Already, the method used in this thesis blurs the line between single-document summarization and multi-document summarization.

The three ways to classify text summarization are: extraction vs. abstraction, generic vs. query-focused, and multi-document vs. single-document. Single-document summarization and multi-document summarization appear relatively straight forward. Single-document summarization is simply relying on one input text to form a summary whereas, multi-document uses a group of related documents to generate one summary about all the input texts. Though seemingly simple to discern, the method in this paper does not fit into one category or the other and this will likely be the continuing trend of research in this field.

Generic summarization simply takes an input text or texts and generates a summary. Most recent work in the past four years has focused on query-focused summarization. The goal of query-focused summarization is to return a summary that contains information answering the query. As Zhao et al. pointed out earlier this year, "Most query-focused summarizers are extended from generic summarizers by incorporating query-related features, such as the similarity between a sentence and the query."[41]. The method I put forth in this paper is generic, but it is easily modifiable into a query-focused method. Already, I have implemented a simple $tf * idf$ weighting scheme that returns the most relevant page to a query and then summarizes it, but this is merely to demonstrate the feasibility of extending it.

The actual writing of the summary is generated through either extraction or abstraction. Extraction is the process of selecting entire portions of the source text (usually sentences) and arranging them in such a way as to form a summary. Abstraction is taking at least some information not directly written in the source text and including that in the summary. This is often introducing another idea or concept that says something in fewer words, or combining multiple sentences in such a way that are no longer cut-and-pasted. Abstraction is generally more difficult for a computer to do than extraction because it requires semantic understanding of the text. In another paper by K. Sparck Jones discussing the state of summarization, she notes "There has been relatively little non-extractive summarising in the last decade, so it is harder to draw any conclusions ..."[14].

With the continuing improvements in computer hardware, processors and ease of access

to information, I expect that this will change in the next decade. In my opinion, I expect that this will be accomplished not through a greater semantic understanding of input texts, but rather a blurring of the lines between all three of these classifications of summarizations. This paper takes the first few steps towards this and outlines the many, easily achievable ways to continue this that were outside the scope of this project. Summarization by Latent Dirichlet allocation creates single-document summaries but relies on multiple documents to figure out the topics contained within that single document. An easy extension for future research would be to make use of the graph structure representing Wikipedia articles and their inter-connectivity. Links between articles in Wikipedia are an easy way to obscure the delineation between abstractive and extractive summaries because it will become more difficult to decide what constitutes outside knowledge when a reference is given within the document.

1.3 About this Paper

This thesis provides a novel approach to automatic summarization through Latent Dirichlet allocation. By focusing on the topics contained within a document, my method generates better summaries with less repetition. Furthermore, I introduce a new evaluation method that addresses some of the flaws in conventional summarization evaluation by distributing the decision making process so as not to emphasize one idea or gold-standard.

I will begin with a brief introduction to related works in chapter 2, followed by a progression from LDA in chapter 3 to SLDA in chapter 4. I then discuss two other leading algorithms in chapter 5 and how they will compare to SLDA. I discuss the problems of evaluation in chapter 6 and the finally discuss my results in chapter 7 where I compare SLDA to the two other methods.

Chapter 2

Related Works

2.1 Historically Influential Works

The first paper on automatic summarization, by H.P. Luhn, is on automatic extraction of technical papers and was intended to automatically generate abstracts[21]. He uses a relatively simple, yet effective approach of using the frequency of words to determine how important they are. His basic hypothesis is that relevant topics in a paper will be mentioned more often and therefore should be mentioned in a summary. Furthermore, in very technical papers, he hypothesizes that there is a low probability of the same word having multiple meanings. The method outlined in the paper uses a stop list of common words, as is now standard practice in most information and textual retrieval processes. Another interesting part of the algorithm is that he stems words so that different endings of the same root word are considered the same. Two words are judged to be the same word if one of them has six or fewer letters appended onto the end of the same root word. In the paper he gives the example of, "Thus the variants *differ, differentiate, different, differently, difference* and *differential* could ordinarily be considered identical notions and regarded as the same word."[21] Being the first paper on the subject, it is incredibly successful and is still cited and reviewed over half a century later. That being said, there are some drawbacks. The method, though somewhat successful, is very simplistic. It also only focuses on extraction

as opposed to summarization. The goal of his research was to generate abstracts for the large number of technical papers that have been written, and therefore focused on getting the relevant aspects of the paper, not necessarily presenting it in a way that causes it to flow. This paper was also very successful for the technology that was available. He was only able to use papers that had 4,000 words or less, a fraction of the size of this document, because that was the storage capacity of the machine available.

The next major paper in this field was by Edmundson, written about a decade later, and was also a paper on extraction and the generation of abstracts[8]. It refines Luhn's methods a little and adds some complexity to them. There are four major areas that he outlines as being potentially important to summarization. The most important contribution to the literature of these four is the location method. He hypothesizes that those sentences at the very beginning and end of papers, such as the introduction and conclusion, will be more important in summarizing the entire paper and therefore should be weighted more heavily. The method is somewhat beneficial, but the major contribution of this paper is the fact that it is adding new parameters to Luhns method and adding complexity – a trend that continues as methods progress.

One of the seminal papers in Information Retrieval, "A Statistical Interpretation of Term Specificity and its Application in Retrieval", by Karen Sparck Jones, standardizes how the frequency of occurrences of a term will influence documents and corpora[12]. Her method standardizes the frequencies of terms based on how many documents they appear in. Essentially, a term that occurs in every document is less useful, relevant, and specific than a term that occurs a few times and only in one document. This method still forms the basis for nearly every information retrieval system out there, though generally slightly modified and made more complex. This concept was incredibly useful and drove the field of Information Retrieval to what it is today. Her method has been applied just as successfully to many other subfields of NLP — and summarization is no exception.

C. Paice and P. Jones' famous 1993 paper attempts to summarize by abstraction[33]. Using highly structured empirical research papers, it tries to fit portions of the source text

to various slots in a frame. Candidates for extraction are determined using context patterns that define various stylistic constructs. Upon ranking the candidates, output templates are filled with the top candidates. Since the information contained in these templates are not included in the original document, it classifies as abstraction. The downsides to this method are that the same templates are used for all summaries, so it is mimicking a more intelligent abstractor and the input must be very specifically formatted.

2.2 Modern Day Approaches to Summarization

Modern day approaches to summarization try a variety of methods that attempt to model the documents more sophisticatedly. Many of the methods are inspired by other areas of NLP research and are only tailored in order to apply to summarization.

One of the most recent works was the application of Non-negative Matrix Factorization to summarization [18]. This method also focuses on the subtopics of a document like SLDA, and claims to be quite successful. Having successfully been applied to many other related tasks to summarization, just like LDA, it was never previously applied to this specific task. I will spend more time discussing this method later on because I will compare it to SLDA in the evaluation portion of this thesis.

Many graph based approaches to automatic summarization have also been developed. Most of the early work with graph based summarization is built upon work done in other aspects of Natural Language Processing, and in particular, Information Retrieval. More specifically, the successful webpage ranking algorithm PageRank developed by Larry Page and Sergey Brin was the basis for the first two major graph based algorithms [32]. The rational behind applying this method to this field is that document summarization is an information retrieval task where extracting the most important sentences to include in the summary is similar to returning the most important documents in search engine query. The first system, named LexRank which is short for lexical PageRank, is a multi-document summarization system that measures similarity between sentences. It is a generic, unsuper-

vised, extractive summarization system. Unlike PageRank, the similarity graph between sentences is undirected, but this does not change the computation or stationary distribution. LexRank operates under the bag-of-words assumption which means that the order of words does not matter only that they are included in the document and can be randomly selected for a document. Calculating the similarity between two sentences is done using a modified cosine similarity equation called $idf - modified - cosine(x, y)$. LexRank performed well in the DUC 2004 evaluation and ranked first in more than one task [9]. LexRank was further expanded to be query-focused in "Using Random Walks for Question-focused Sentence Retrieval" [31]. Here individual sentences were returned from complex news stories based upon a specific question, but the underlying algorithm was still the same.

The other early graph based summarization system developed around the same time is TextRank. As with LexRank, it is based upon PageRank and uses undirected graphs, though unlike LexRank, TextRank is only for summarizing single documents and is a generic, unsupervised, extractive summarization system [26]. Initially, the text is tokenized and then annotated with part of speech tags, but it only uses single words as possible additions to the graph [28]. This method implements the concept of recommendation, which means that a text unit recommends other related text units. The strength of the recommendation is computed recursively [26]. The overlap between two sentences is the number of common tokens between the lexical representations of the sentences. A normalizing factor is used to avoid biasing longer sentences. It does not require any training data, nor does it favor any particular language [27].

The most recent graph based approach is an expansion of the query based LexRank. The method, presented in "Using query expansion in graph-based approach for query-focused multi-document summarization", attempts to build upon LexRank by incorporating both sentence importance and sentence-to-word relationships [41]. It is a multi-document, query based, unsupervised, and extractive summarization system. It uses pairwise similarities between sentences using a $tf * idf$ evaluation. Previous literature took word synonyms as expansion words, but this method takes informative and query relevant words based on the

graph ranking results and continues into an empirical quantity is reached. In an attempt to avoid redundant and repetitive sentences, a redundancy penalty is calculated using a greedy algorithm [41]. The redundancy penalty is based upon the Diversity Penalty presented in "Improving Web Search Results Using Affinity Graph" [40].

Chapter 3

Latent Dirichlet Allocation

Latent Dirichlet allocation, or LDA for short, is a generative probabilistic model of collections of discrete data[5]. It is an algorithm used in Natural Language Processing to model the topics within documents and corpora. It is the basis for my summarization method which attempts to form summaries based upon coverage of document topics. LDA successfully attempts to improve data modeling over other methods by allowing for documents within corpora to be modeled as collections of topics. The unique and revolutionary idea behind this model is that the topic variable in the model is selected repeatedly within each document — allowing for documents to be comprised of multiple topics. This is also the intention of the pLSI method, but LDA uses a hidden random variable that allows for it to adapt to previously unseen documents without overfitting. For the discussion of LDA within this paper, I will focus on documents that are comprised of text corpora. This will lead to its' application in summarization and the creation of my work, SLDA. SLDA directly derives from LDA and in order to fully understand this method, it presupposes knowledge of LDA.

LDA has been applied to a variety of tasks comprising many different areas of Natural Language Processing and even much more general areas of Computer Science. Despite all of this, no work has been done in automatic summarization using Latent Dirichlet allocation. As with many other areas of statistical natural language processing, the potential impacts of

LDA are very large and beneficial. Many of the current leading methods within automatic summarization are attempting to solve the same problems that LDA addresses. In fact, the most recent work in the field has focused on modeling document topics which is exactly the problem that LDA is designed to solve.

3.1 Overview of LDA

LDA is a generative model, which means that it attempts to describe how a document is created. It is a probabilistic model because it says that a document is created by selecting topics and words according to probabilistic representations of natural text. For instance, the words that I use to write this paragraph pertain to a subtopic of this entire paper as a whole. The actual words I use to compose it are chosen based on that topic. The inherent probability in modeling the selection of each word stems from the fact that natural language allows us to use multiple different words to express the same idea. Expressing this idea under the LDA model, to create a document within a corpus, imagine that there is a distribution of topics. For each word in the document that is being generated, a topic is chosen from a Dirichlet distribution of topics. From that topic, a word is randomly chosen based on another probability distribution conditioned on that topic. This is repeated until the document is generated.

The basic idea behind modeling a corpus with a Dirchilet distribution over topics is that documents will have multiple topics and those will overlap. For instance, within a corpus of documents about Princeton University, there will be individual papers that discuss the Computer Science Department. There will likely be certain words that are used more frequently when discussing the Computer Science Department than other departments on campus such as: *Computers, Algorithms, Graphs, Data, Modeling*, and *Networks*. Other departments, such as Sociology may have papers discussing topics that use words like: *Gender, Race, Age, Economics*, and *Networks*. LDA views corpora as a whole and picks out the topics from there. If documents were compared individually, it might be the case

that certain topics were not picked up on, and it is only when the entire corpus is looked at that certain topics are noticeable. In this example, words like "Networks" might only appear a few times in papers pertaining to either department. Essentially, LDA is creating a more realistic model of the corpus, and thereby, the individual documents. Words that appear less frequently in single documents, but are prevalent in many different documents are likely indicative of a common topic between those documents. When generating a summary, the ability to pick out the nuances in document topics will allow for more pertinent information to be included with less chance of repetition which will ultimately yield a better summary.

A key assumption in LDA is that words follow the "bag-of-words" assumption — or rather that order does not matter. A word's use is to be a part of a topic and it will convey the same information no matter where it is in the document. This assumption says that "Harry hired Sally" is the same thing as "Sally hired Harry." In both cases, the set of words is the same along with the frequency of each word. This assumption is necessary for the probabilities to be exchangeable which allows for more mathematical methods to be able to apply. Though it occasionally treats semantically different sentences as the same thing, it works well on an overall document. On the other hand, language modeling would require a different assumption. Blei et al. leave this up to future work — suggesting using trigram models instead of a unigram model[5]. Regardless, this will not have much of an impact on SLDA which extracts entire sentences and will thus be less affected by individual word semantics.

3.2 Applications of LDA

LDA was initially developed to model discrete data sets in general, though primarily textual documents. The original paper written about the model focused on three applications: document modeling, document classification, and collaborative filtering. Since then, its applications have naturally increased in scope as much other research has been done using it as a framework — yet not in a summarization context.

In the three initial applications discussed when first published, Latent Dirichlet allocation performed very well. In the task of document modeling, LDA performed better than pLSI and a mixture of unigram models, as was their hypothesis. pLSI overfitted probabilities of previously seen documents when determining the topics in a new document. As expected, the major advance of LDA over pLSI was that it easily assigned probabilities to a previously unseen document. Their second application, regarding document classification, had results suggesting that LDA might be useful as a fast filtering algorithm for feature selection[5]. The last task that they outlined went beyond simple text documents. The EachMovie collaborative filtering data experiment tried to determine user preferences of movies. Instead of having a text document, you have a user, and instead of individual words, you have movies chosen by the user. The data set was evaluated using a held out estimator, and again it performed better than pLSI and a mixture of unigrams.

LDA is a robust and generic model that is easily extensible beyond the small empirical data set discussed when published. Numerous papers in a variety of fields have been published applying LDA to a vast range of areas. It has been applied to tasks ranging from fraud detection in telecommunications to finding bugs in source code[36, 22]. In spite of the wide range of applications, LDA has not been applied to automatic document summarization though the possibility is quite feasible.

3.3 Derivation of LDA

In order to discuss Latent Dirichlet allocation and then Summarization by Latent Dirichlet allocation, it is useful to first discuss what a Dirichlet distribution actually is.[1] A Dirichlet distribution models how proportions vary[29]. A k-dimensional Dirichlet random variable θ can take values in the (k - 1)-simplex and has the probability distribution on this simplex

[1]The equations in this section (and much of the discussion related to them) come from Blei et al. 2003. For more complete derivations of the formulas please refer to it. For a more complete discussion of Dirichlet distributions, see Minka 2000 and for complexity analysis of LDA, see Blei et al. 2003

of:

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{i=1}^{k} \alpha_i)}{\prod_{i=1}^{k} \Gamma(\alpha_i)} \theta_1^{\alpha_1 - 1} \cdots \theta_k^{\alpha_k - 1} \tag{3.1}$$

θ_k, by definition must be greater than zero and $\sum_k \theta_k = 1$[5].

In the context of LDA, a document is generated by selecting each word in a document according to the following algorithm. Given that β is the probability matrix for determining the probability word w will get chosen given the topic, a word, w_n, is chosen from $p(w_n|z_n, \beta)$. This is a multinomial probability conditioned on the topic z_n, where the topic z_n has been chosen from the Multinamial(θ). θ was in turn, chosen from our Dirichlet distribution based upon α. This process is repeated for each document, $\mathbf{w}$, in a corpus D.

To obtain the probability of a corpus, we simply take the product of the marginal probabilities of single documents:

$$p(D|\alpha, \beta) = \prod_{d=1}^{M} \int p(\theta_d|\alpha)(\prod_{n=1}^{N_d} \sum_{z_{dn}} p(z_{dn}|\theta_d) p(w_{dn}|z_{dn}, \beta)) d\theta_d \tag{3.2}$$

α and β are corpus level parameters where α determines the shape of the Dirichlet distribution and β is a matrix of size k (number of topics) by the number of distinct words in the document (V). The values inside of β are the probabilities that a word occurs given the topic.

The only observable part of this model is the collection of words within documents in our corpus. Given this, LDA attempts to determine what the hidden variables are:

$$p(\theta, \mathbf{z}|\mathbf{w}, \alpha, \beta) = \frac{p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta)}{p(\mathbf{w}|\alpha, \beta)} \tag{3.3}$$

Unfortunately, this problem is intractable, so LDA attempts to determine these through an heuristic called Variational Inference. This method simplifies the model above and adds free variational parameters. The Dirichlet parameter γ and $(\phi_1 \ldots \phi_N)$ are the variational

parameters. We can now rewrite equation 3.3 as:

$$q(\theta, \mathbf{z}|\gamma, \phi) = a(\theta|\gamma) \prod_{n=1}^{N} q(z_n|\phi_n) \tag{3.4}$$

The variation parameters are found using an iterative algorithm which stops after either minimizing the Kullback-Leibler divergence to the true posterior or by reaching the maximum number of iterations. The equations for the update functions are:

$$\phi_{ni} \propto \beta_{iw_n}^{E_q[log(\theta_i)|\gamma]} \tag{3.5}$$

$$\gamma_i = \alpha_i + \sum_{n=1}^{N} \phi_{ni} \tag{3.6}$$

For a document of length m words and k topics, each update takes $O(mk)$ time [30]. The next goal is to attempt and find α and β which is also intractable.[2] This is approximated by finding the maximum likelihood estimates for each document in the previous step. This process is also iterative and is repeated, finding the variational parameters and then finding the maximum likelihood estimates. It is described by:

$$\beta_{ij} \propto \sum_{d=1}^{M} \sum_{n=1}^{N} \phi_{dni}^{*} w_{dn}^{j} \tag{3.7}$$

Upon finding close approximations of the hidden variables, we now know the distribution of topics and the innate underlying structures of the documents in our corpus. This has numerous useful applications from text classification to generation of new documents, but for the purpose of this paper, we can now begin to extract a summary.

3.4 Motivation for Summarization

Knowing the distribution of topics is a major step toward generating a summary. From the distribution of topics, we can pick out sentences that are most closely related to topics and

[2] Again, see Blei et al. 2003 for a more complete discussion of why

by covering the most topics well, we have a good model for a summary. A good summary conveys the most information without repeating itself and being able to select sentences from separate topics. By using LDA for summarization, we are able to figure out the multiple topics within a document and generate better summaries.

Though the ultimate summaries that we generate are single-document, SLDA has a multi-document aspect to it by running the LDA variational inference algorithm on all of the documents at once. Though no direct information is extracted from another document in the corpus for a particular summary, indirectly the topic distribution of the document is affected and this allows for a better representation of the document and therefore a better summary. This melding of classifications allows for a better representation of an individual document and creation of a better summary.

Chapter 4

Summarization by Latent Dirichlet Allocation

In this chapter, I will provide the details of my implementation so that my work could be repeated. Those readers who are not interested in the specific decisions and procedures used in obtaining my results can skip this chapter and proceed to chapter 5. After a brief overview section, section 4.2 contains information about how I obtained my corpus. Section 4.3 discusses how the topics were obtained through variational inference using open source code. The last section, section 4.4 discusses how SLDA uses a Zipfian distribution to model the words in a topic and extract sentences — an extension over the Poisson distribution modeling of topics presented in Blei et al. 2003.

4.1 Overview

Summarization by Latent Dirichlet allocation is a simple extension of LDA. It works by preprocessing a corpus of documents' terms into a frequency metric. This is then applied to the normal LDA method's variational inference algorithm. After the hidden variables are determined by the inference, we model the distribution of words in the topics according to a Zipfian distribution. Individual words are scored based upon their relevance to a topic

and the combined scores of all of the terms in a sentence form the score of the sentence. These are sorted and the top k sentences are selected to form a summary.

4.2 Obtaining a Wikipedia Corpus

The first step in summarizing Wikipedia articles is to obtain the corpus. The English version of Wikipedia is free and easily obtainable from the following link http://download.wikimedia.org/enwiki/ . Periodically, entire dumps of the Wikipedia site are downloaded and made available (without pictures). For this project, I obtained the latest download in XML[1]. The download, once uncompressed, yielded a 21 gigabyte file with each article appended after the other using the XML tag <page>. In order to work with the articles, it is first necessary to split them up into multiple articles and a usable format.

Using awk scripts, I separated the initial one million three hundred thousand articles into that many separate files. The files were clearly arranged according to a hash function — so I was guaranteed to have a random subset of Wikipedia. There are over 2.8 million English articles on Wikipedia, but that count is restricted by a few parameters[2]. Redirect pages and categories, amongst other things do not count in that total — but they are included in the downloaded file. There is no published statistic on the ratio of redirect page to actual article, but from my sample of 1.3 million, more than 50% were redirect pages. Thus, the sample of pages that I took was not one-half of all of English Wikipedia, but closer to one-fifth. Redirect pages arise because individual authors of articles will refer to a similar concept using separate words and create links to separate pages. Other editors of articles on Wikipedia will later realize that these disjoint links should link to the same page and one of the links will redirect to only one of the page.

After removing all of the redirect pages, I cleaned up my sample a bit more by removing any page that contained a colon. This is the format used by Wikipedia to signify a category and is not actually an article. For intsance, “File:” indicates a picture (which is not

[1]Obtained in mid-March, 2009 and can be found by the reference label: 2009-Mar-13 01:27:21

obtainable) and "Template:" equates to small pictures such as nations' flags.

I then separated a random one thousand articles from that directory to serve as my corpus. The complete list of the one thousand random article titles can be found in appendix D.1. I imposed a couple of limitations on the corpus: First off, I only allowed file names composed in basic Ascii format to be allowed. This was done for ease of computation. It would be relatively simple for later work to expand this to Unicode also known as UTF-8 (the format used by Wikipedia) and this is one possibility to expand this research. The assumption used in this project is that this work is focused on English text summarization, and the majority of concepts in English can easily be expressed in a written title of Ascii characters.

Secondly, I limited file sizes to be greater than 2,000 bytes. This criteria was imposed because summary length was set to 10 sentences and it is worthless to generate a 10 sentence summary if the actual article is only 10 sentences or slightly larger.

4.3 Calculating Topics

After obtaining my corpus, I preprocessed the documents before they could be summarized. Again, I removed non-Ascii characters from the document. I also removed section headings, links, XML tags, and other non-useful data while converting whatever encoded information into text that was possible such as inserting quotation marks where applicable (&quot = ").

Unfortunately when running LDA on my corpus, I was unable to use all one thousand articles at a time. Doing so required reading and writing gigabyte size files every iteration of the algorithm and was too computationally complex for this current research — though is easily feasible with more time and larger storage spaces in future work. Instead, I separated the one corpus into ten, one hundred document corpora and ran the LDA algorithm on each one separately. This was much more computationally feasible.

For each of the ten corpora, I parsed each word, converted it to lower case and checked

if it was a stop word according to the stop list given in appendix A and found out the total number of terms. Then, for each document, I calculated the frequency of each word in the term list storing the information in a vector. Combining all of the vectors formed the matrix A defined in LDA.

A was then fed into an open source C code implementation of LDA maintained by David Blei.[2] I set the parameter k, number of topics, to equal 50 for each of the ten corpora of size one hundred documents. I limited α to equal $\frac{1}{k}$ or 0.02 and did not estimate it because that ratio works well empirically. I set the tolerance level to 10^{-6} and the maximum number of iterations to fifty.

After inferring the hidden variables, I used the included python script from LDA implementation to match 100 terms to each of the 50 topics. I then combined these ten term files into one large term file so that my ten separate corpora were now combined to estimate the original corpus of one thousand documents.

4.4 Zipfian Distribution of Words

In Blei et al. 2003, a document is comprised of N words. The set of N words is chosen based upon a probability distribution[5]. In the paper itself, N is conditioned according to a Poisson distribution (ξ). As the authors mention, this need not be the case, and furthermore N is an ancillary variable so the choice of distribution will not affect any other parts of the model. The possibility to use any term distribution model is left open to be determined based on the application.

After determining the topics in the aggregate file defined in the previous section, I ranked sentences in each document based on how they fit within topics. First off, sentences were passed through a few simple syntactic filters to remove punctuation, convert to lower case, and remove stop words — the exact same process as the preprocessing phase before the LDA inference algorithm was run. Instead of the Poisson distribution mentioned in the paper, I assumed that words in a topic occurred according to a Zipfian distribution. A

[2]Available http://www.cs.princeton.edu/ blei/lda-c/index.html

Zipfian distribution is simply a distribution that behaves Zipf's law. The frequency f of a word in a sorted list of frequencies is related to its' position r in the list. Formally, Zipf's law is:

$$f \propto \frac{1}{r} \tag{4.1}$$

Zipf's law better models data at the corpora level — not individual documents — but topics are taken from the corpus as a whole so I felt that this would be a legitimate distribution to use [25]. One potential area for future work would be to compare the impact of a number of different distributions and the impact that it has on the summaries. In this system, the weight of a word in a topic is modeled as a simple Zipfian distribution where the first word in a topic list is weighted twice as much as the second word, and so on. For each word in a sentence, the algorithm finds the weights of that word in all topic lists (which could be 0) and sums them together. This process is repeated for every word in a sentence and then those scores are added together to get the total score of a sentence. The sentences are sorted based on their scores and the top k-1 sentences are selected to form the summary. The first sentence of the summary is selected as described below.

Due to the corpus chosen for this project, the first sentence in the summary is the first sentence in an article that contains the title (if it exists). Generally, the first sentence in a Wikipedia article is written to convey the most amount of information about the article title. Even for very small values of k, the weighting scheme used in my system generally selected this sentence — it just did not necessarily put it first. This makes sense because the first sentence should capture at least one topic detailed in the document if not more and so should receive a high ranking. The decision to include this portion in my system was a difficult one because the system is designed to be robust and extensible. It should perform just as well on other corpora. Ultimately, I decided that even in other corpora, the first time that the title is mentioned would likely be an important sentence to include in the summary and was probably written by the author in such a way that in minimized

anaphoric references and was a good segue into the start of the document. If a sentence meeting this criteria was not found then the top k sentences were selected for the summary according to the method outlined in the previous paragraph.

Chapter 5

Rival Methods

A good summary should capture and present the most relevant and informative aspects of a document without repeating itself. As mentioned in chapter 2, there are many different ways that previous work has attempted to accomplish this task from — redundancy penalties to attempting to model the semantics of a document. For this paper, I have focused on modeling the topics of a document.

Lots of recent work has been done in other areas of NLP research involving modeling documents. Latent Semantic Indexing (LSI), also known as Latent Semantic Analysis (LSA) attempts to improve upon simple $tf * idf$ and is successful in capturing some basic linguistic notions[5]. In order to account for some deficiencies in LSI, probabilistic LSI (pLSI) was developed, which models a document as having multiple topics and each word is chosen probabilistically from a single topic. pLSi is limited by the fact that it cannot assign probability to a previously unseen document outside of its training set[5].

Recent work involving Non-negative Matrix Factorization claims to improve upon LSI methods for modeling subtopics when applied to summarization [18]. Unfortunately, the algorithm chosen has numerous drawbacks. It is computationally expensive and does not guarantee a local minimum in the approximation of the matrix heuristic. Despite these deficiencies, the results were very good on the DUC 2006 data set compared to LSI and suggest that more complicated and sophisticated methods to model subtopics in a document

should yield even better results.

As SLDA attempts to summarize by a similar hypothesis, I wished to test the results against each other. Being that they are both new additions to the field (Non-negative Matrix Factorization's application to summarization was only published earlier this year), I felt that the two methods should be compared to a well known and successful algorithm. For this, I choose TextRank (one of the leading graph based approaches to summarization) because it attempts to model interrelatedness of sentences in the document — though is not focusing on the notion of topics. I think that the two models based upon document subtopics will outperform TextRank because of the better representation of a textual document that their models bring.

5.1 Overview

As will be discussed more in depth in Chapter 6, a method is required to test the results of my system. In order to accomplish this, I decided to take two other leading methods in the field and use them as a comparison against mine. The first method that I decided to use was published earlier this year in the *Journal of Information Processing and Management*. The method, Non-negative Matrix Factorization, had been used for semantic analysis of text documents — but had not been used in summarization until recently. As it was very recently published and claims to have great results, along with intending to model document topics, I think it would contrast well with Summary by Latent Dirichlet allocation in an evaluation.

The other method that I have decided to use is TextRank. As briefly described in the related works section 2.2, TextRank is one of the premiere summarization methods that relies on graphs. As with LexRank, it derives from the famous algorithm PageRank. Graph ranking algorithms have performed very well in other NLP tasks (in addition to IR), and it seems like a legitimate hypothesis that they would do well in summarization as well — which is essentially what the results have shown. Modern extractive research is still pursuing this

path, and selecting an algorithm from these methods seems to be a worthwhile comparison for my algorithm. In modern graph-based summarization research, TextRank and LexRank are the most well-known and oft cited. LexRank is designed for multi-document summarization, whereas TextRank is designed for single-document summarization. My method is designed for single-document extraction, so it was a fairer comparison to use TextRank [41].

In order to fully develop the comparison between all three methods, it is beneficial to describe in detail the methodology of each algorithm. For both TextRank and NMF, source code was not available and I was left to recreate the author's work according to their publications'. Care was taken to ensure that a fair assessment of each method would be used in this experiment. Across all three methods, the same list of stop words was used along with other sentence preprocessing techniques from chapter 4. Anytime ambiguity was encountered, decisions were made based upon what would both keep the integrity of the original authors' works intact and not disadvantage a method within the evaluation metrics used. For instance, R. Mihalcea says "it can be run through syntactic filters", but never explicitly details which factors she used of the method outlined in her analysis [28]. In this case, it was resolved by deciding to only use the common stop list as a syntactic filter because it both preserved her work and was grounds for a fair evaluation.

5.2 Non-Negative Matrix Factorization

Non-negative Matrix factorization, or NMF, was first published as Positive Matrix Factorization by P. Paatero and U. Tapper in 1994[4]. It did not receive much notice until D. Lee and S. Seung published their famous article, "Learning the parts of objects by non-negative matrix factorization" in *Nature* in 1999[16]. Lee and Seung discuss how the algorithm is good at modeling the generation of visible variables from hidden variables — presenting results of facial image recognition and semantic analysis of documents.

5.2.1 Overview of NMF

The motivation behind the authors work in "Automatic generic document summarization based on non-negative matrix factorization", is that "LSA-related methods of document summarization may fail to extract meaningful sentences" because they do not model subtopics correctly[18]. This, the authors attribute to the fact that the singular vectors obtained from LSA are not sparse and have numerous negative and positive weighted terms. Instead, they propose a method based upon Non-negative Matrix Factorization. Their system is unsupervised, generic, and extractive. The premise behind their choice of NMF is that "people use only non-negative information when recognizing an object as a combination of partial information" concluded from Lee and Seung's results in *Nature*[18, 16]. The factoring algorithm used is a multiplicative update algorithm and is the first one given in "Algorithms for Non-negative Matrix Factorization"[17].

In order to create a fair experiment, I attempted to recreate Lee et al.'s work based on the information put forth in their paper. In my discussion here, I will attempt to use the same notation that they present in their paper — but they are relatively inconsistent with their notation so I have opted to only use one form.

5.2.2 NMF Algorithm

Non-negative Matrix Factorization, NMF, iteratively decomposes an m x n matrix A into two other matrices, W and H. W is a m x r matrix and H is a r x n. Together they approximate A.

$$A \approx WH \tag{5.1}$$

The matrix A is composed of m terms and n sentences. A is the representation of the original document according to a frequency weighting scheme. The values of A are the frequencies that a term appears in a sentence. In their paper, Lee et al. test a variety of weighting schemes. While replicating their work, I decided to use the Binary weighting scheme because it performed the best overall in precision evaluation in all ROUGE measures[18]. The Binary

weighting scheme is very simple — the value of $A_{ji} = 1$ iff term j occurs one or more times in sentence i. Other wise $A_{ji} = 0$. The rational behind decomposing A into two other matrices is that W will represent semantic features and H will represent semantic variables.

To calculate W and H, we must first calculate A. That is a relatively straight forward task of breaking a document down into sentences, preprocessing those sentences to remove punctuation, and then applying the same stop words list that I apply to my own algorithm. From A, we start with H and W filled with non-negative values and iteratively update both matrices until they are within a certain error rate of A, or the maximum defined number of iterations is reached. Because of the computational complexity of the algorithm, I was forced to limit the maximum number of iterations to 10 and the tolerance level to .0001. The tolerance rate is calculated using the Frobenius norm:

$$\Theta_E(W, H) \equiv \|A - WH\|_F^2 \equiv \sum_{j=1}^{m} \sum_{i=1}^{n} (A_{ji} - \sum_{l=1}^{r} W_{jl} H_{li})^2 \tag{5.2}$$

The update rules for calculating H and W are:

$$H_{li} \leftarrow H_{li} \frac{(W^T A)_{li}}{(W^T W H)_{li}} \qquad\qquad W_{jl} \leftarrow W_{jl} \frac{(AH^T)_{jl}}{(WHH^T)_{jl}} \tag{5.3}$$

Upon reaching a satisfactory level of approximation for W and H, the information is used to select sentences for a summary. To create a summary of length k sentences, the k sentences with the highest Generic Relevance of a Sentence (GRS) are selected. To calculate the GRS, the following function is applied to W and H after they have been approximated as described above:

$$\text{GRS of a } j\text{th sentence} = \sum_{i=1}^{r} (H_{ij} * \text{weight}(H_{i*})) \tag{5.4}$$

$$\text{weight } (H_{i*}) = \frac{\sum_{q=1}^{n} H_{iq}}{\sum_{p=1}^{r} \sum_{q=1}^{n} H_{pq}} \tag{5.5}$$

5.2.3 Deficiencies

In Lee and Seung's 2001 paper, a formal proof is given that states that the algorithm is guaranteed to be locally optimal [17]. In reality, that is incorrect. The algorithm only demonstrates a continual descent property and thus could actually descend to a saddle point instead of a local minimum [4]. Furthermore, computationally, it is a very expensive algorithm requiring $O(m^2k)$ operations for every iteration.

5.2.4 Use in My Work

Despite these deficiencies, this is still one of the most recent publications in the field of Automatic Summarization — having only been published earlier this year. It is a novel application within this field and having very little formal reviews of their work yet, I felt that it would be worthwhile to test it against my own work. Lee et al. published good results from NMF summarization and I felt that it would be beneficial to test it alongside both my method and a famous algorithm (TextRank).

5.3 TextRank

TextRank is an algorithm developed by Rada Mihalcea in the Computer Science Department at the University of North Texas [26, 27, 28]. It is one of the preeminent graph-based summarization algorithms and is often cited within summarization literature. It has been applied to many different fields within Natural Language Processing in addition to summarization.

For the discussion of TextRank, I will use the notation put forth by R. Mihalcea even when discussing the algorithms influencing TextRank. Let $G = (V, E)$ be a directed graph with the set of vertices V and edges E. E is by definition a subset of V x V. For a given vertex V_i, $In(V_i)$ is the set of vertices that point to V_i and $Out(V_i)$ is the set of linked to by V_i.

5.3.1 Influences on TextRank

TextRank is based upon Google's PageRank algorithm and Kleinberg's HITS of ranking relevance of webpages in the related NLP task of Information Retrieval [15, 32]. Motivated by explaining the derivation (and errors stemming from it), I will explain TextRank in the context of PageRank and HITS. The basic method of these algorithms is to weight a website according to pages linked to and from the site. The model attempts to represent how people browse the internet. A random walker variable is added to the model to account for the fact that people do not start off at a specific webpage and click on hyperlinks until they reach another page, but sometimes jump to a specific page by typing in the URL. Formally, this is defined as:

$$PR(V_i) = (1 - d) + d * \sum_{V_j \in In(v_i)} \frac{PR(V_j)}{|Out(V_j)|} \tag{5.6}$$

In this equation, d represents the random walk effect that dampens scores of a vertex and is limited between 0 and 1. TextRank uses a value of 0.85 for d that is also recommended by Brin and Page in their seminal paper [32]. From this equation, it is easy to see that the weight of a vertex is determined by the sum of the weights of vertices linking to it. The

weight of each vertex pointing to it is normalized by the number of edges leaving from it. In other words, if a page has a score of 3, but has three edges directed outwards from it, the influence on the score of each one of those pages' ranks is only going to be 1. It is important to note now, as will become apparent later in our discussion, that directed cycles are permissible in this model.

In addition to PageRank, TextRank derives part of the algorithm from Hyperlinked Induced Topic Search (henceforth referred to as HITS). HITS weights websites based upon the direction of edges at a specific vertex. There are two weights per vertex, a hub weight score and an authority weight. Hubs are pages with lots of outgoing links, whereas authorities are pages with numerous incoming links. The scores are normalized so that the squared values of both authority and hub scores sum to one. Like PageRank, HITS is an iterative algorithm that updates the scores. Updates are done according to the operations:

$$HITS_A(V_i) = \sum_{V_j \in In(v_i)} HITS_H(V_j) \tag{5.7}$$

$$HITS_H(V_i) = \sum_{V_j \in Out(v_i)} HITS_A(V_j) \tag{5.8}$$

5.3.2 TextRank Algorithm

The TextRank algorithm models a single document as a graph. The vertices of the graphs are individual sentences and the edges between them are weights obtained by the similarity of two sentences. Essentially, TextRank combines both PageRank and HITS to obtain a model of a document. In the paper, the term *recommendation* is used to refer to the relation between two sentences. In other words, a sentence talking about certain concepts in a text refers to other sentences in the text that are talking about the same concept — and this is accomplished through using similar words. The formal definition of the TextRank algorithm is as follows:

$$PR^W(V_i) = (1-d) + d * \sum_{V_j \in In(v_i)} w_{ji} \frac{PR^W(V_j)}{\sum_{V_k \in Out(V_j)} w_{kj}} \tag{5.9}$$

Here w_{jk} is the weights of the edges between two vertices defined by a weighting function. TextRank is a generic algorithm also applied to keyword extraction in addition to summarization. For the task of summarization, weights between vertices are the similarities of the two sentences. Let S_i be a sentence of N_i words defined by $S_i = a_0^i, a_1^i, \ldots, a_{N_i-1}^i$. Let S_j be similarly constructed. The similarity between those two sentences is defined as:

$$Similarity(S_i, S_j) = \frac{|\{a_k | a_k \in S_i \& a_k \in S_j\}|}{log(|S_i|) + log(|S_j|)} \tag{5.10}$$

5.3.3 Graph Representations of Documents in TextRank

According to Mihalceas, the graph of the document can best be represented in three different ways: An undirected graph, a directed forwards graph, and a directed backwards graph. A directed forwards graph orients vertices from a sentence to sentences that occur later in the text. A directed backwards graph does the opposite, pointing vertices from a sentence to previous sentences in the graph. Using the concept of recommendation discussed above in relation to directed backwardsness, a specific sentence will recommend a sentence prior in the text. An easy way to think about this is that this sentence, which is discussing recommending, will remind you of the sentences discussed previously in this text where I explained the term. It will not recommend to you of any future sentences that you will encounter later on – though they might refer to this.

Empirically, the paper states that directed backwards representation of texts performed the best overall of the three graph representations. Theoretically, this does not make much sense to me and I will demonstrate why I think that this outcome is a result of flawed methodology and implementation of the algorithm. In most texts, sentences refer to sentences both before and after them. This is the fundamental problem of anaphoric references that I discussed earlier. TextRank's results were evaluated based on the DUC 2002 corpus which was composed of Newspaper/Newswire documents. It is a reasonable assumption to believe that there might be a systematic bias in that type of documents. News stories are often very condensed and succinct – they are already by definition a summary. They are

a shortened account of an event that has happened, represented in a limited space. Furthermore, because multiple events are presented in a news source, each document needs to capture a readers attention quickly and convey information in a limited span. I conjecture that there are fewer forward references in Newspaper/Newswire documents than the average document. Therefore, it would seem intuitive that a directed backwards graph would model this class of documents better than an undirected graph.

5.3.4 Proof of Single Iteration Convergence

Focusing on the implementation of TextRank, I was surprised by a figure representing convergence curves. It is true that scores for nodes are related to other nodes and so calculating the rank for this node will require prior knowledge of the weights of the other nodes. Initially, it would seem that guessing a random weight for each node and then calculating scores would require multiple iterations and imply the necessity of a convergence curve. In reality, this only needs to be true for the undirected graph case. Shown in the paper, convergence curves are drawn for both undirected and directed graphs. To see why there is no need for a convergence curve, multiple iterations, or tolerance limits, consider this simple proof for a Directed Backwards graph representation:

From equation (5.9) we have the TextRank algorithm. By definition of Directed Backwards, a sentence can only reference sentences that occurred before it, and be referenced by sentences that occur after it. For vertices, $i, j_0, j_1, \ldots, j_n - 1$, where $n - 1 \leq ((N-1) - i)$, let $0 \leq i < j \leq N - 1$. Therefore we have:

$$PR^W(V_i) = (1-d) + d * (V_{j_0} w_{j_0 i} \frac{PR^W(V_{j_0})}{\sum_{V_k \in Out(V_{j_0})} w_{kj_0}} + V_{j_1} w_{j_1 i} \frac{PR^W(V_{j_1})}{\sum_{V_k \in Out(V_{j_1})} w_{kj_1}} + \cdots + V_{j_{n-1}} w_{j_{n-1} i} \frac{PR^W(V_{j_{n-1}})}{\sum_{V_k \in Out(V_{j_{n-1}})} w_{kj_{n-1}}}) \tag{5.11}$$

By the similarity equation (5.10), it is plain to see that the weights w_{ji} and w_{kj} will not change between iterations because sentences are static and w_{ji} and w_{kj} are therefore constants.

By definition of a Directed Backwards Graph applied to a text, the last sentence, S_{N-1}, cannot be referenced by any other sentences so:[1]

$$PR^W(V_{N-1}) = (1-d) + d * 0 = .15 \tag{5.12}$$

Due to the fact that we know $PR^W(V_{N-1})$, we can calculate $PR^W(V_{N-2})$:

$$PR^W(V_{N-2}) = (1-d) + d * \sum_{V_j \in In(v_i)} w_{ji} \frac{PR^W(V_j)}{\sum_{V_k \in Out(V_j)} w_{kj}} = \frac{.15}{|Out(V_j)|} \tag{5.13}$$

$|Out(V_j)|$ is by definition, a constant. It is easy to repeat this step until $PR^W(V_0)$, and get the optimal vertices values in one iteration. QED.

This algorithm is only applicable because the graph is directed *and* there are no directed cycles. Though cycles will exist because this is a highly connected graph, it is impossible to be directed back to an initial node through a cycle. If either of the premises above were not true, an iterative algorithm would actually be needed. A similar proof for Directed Forward Graphs is trivial as you start with $PR^W(V_0)$ and continue until reaching $PR^W(V_{N-1})$ in one iteration. HITS and PageRank do not satisfy the second condition because webpages can be part of a directed cycle.

Though I do not know the exact reason why R. Mihalcea has multiple iterations for the directed graphs method in her paper, I hypothesize two possibilities. Either her algorithm started off at a random vertex and therefore had a very small probability of being either the first or last vertex, or the chart refers to a data set that is not guaranteed to be free of directed cycles. It is possible that she developed the algorithm using a data set not related to summarization and therefore had to have an iterative algorithm, such as her discussion on keyword extraction. It is my hypothesis that her algorithm starts on a random vertex and because it has edges into it from nodes with unknown $PR^W(V_j)$ scores that they are not optimal immediately. In the worst case scenario of a fully connected graph starting from V_0, it will only take N iterations to converge. The first iteration will guarantee the weight of

[1]Here d is set to 0.85 as recommended by L. Page and S. Brin

the last sentence. The second will guarantee the weight of the second to last sentence, and so on until the first sentence. Due to this fact, I am predicting that her results are better for Directed Backwards Graphs because they converge to the optimal scores quicker. In spite of the fact that there are probably systematic biases in the DUC 2002 corpus, I think that given many iterations to allow the undirected graph method to converge, a broader corpus would yield better results than the Directed Backwards approach. Despite these potential shortcomings of TextRank, it is a widely cited algorithm in the field of Automatic Summarization and I decided that it would be useful to test against my methods to compare results.

Chapter 6

Summary Evaluation

One of the toughest parts of the field of automatic summarization is the intrinsic difficulty in evaluating results. Though it also is nontrivial to evaluate results in other areas of NLP, the complexities caused by summarization evaluation outpace many related fields. It is quite tough to determine what is the best summary of a document. I am sure that very little agreement would be made over what are the ten most important sentences to extract from this thesis to create a summary — regardless of the number of people doing so. That problem would be magnified even more if abstraction was the goal instead of extraction. A famous example is Rath et al. 1961, which found that people agreed very little over what was the correct summary[34].

The method of comparing summaries to a reference set is referred to as Gold-Standards. As already alluded to, there are numerous problems of determining what is the best summary to use as the gold-standard. As K. Sparck Jones' points out, "there is little solid information about what makes summaries work in contexts" [14]. Despite these deficiencies in gold-standards, every academic field must have a way to evaluate itself and the research done within it so some benchmark must be set. Within summarization, the gold-standard in recent years has been the Document Understanding Conferences and the ROUGE evaluation metrics.

Instead of relying on these flawed metrics, I introduce a revolutionary new method

for summary evaluation. Using the commercial service Mechanical Turk, which distributes simple human intelligence tasks to large numbers of people, I rely on mass human consensus to determine the best summary — much like Wikipedia relies on mass human consensus to determine what the best content of an article is. Despite all of the discussion about good summaries being defined by precision, recall, lack of repetition, and so forth, a good summary is inherently a subjective judgement. Therefore, if a population sample prefers one summary over another, even if it rates lower according to our gold-standard metrics, it should still be considered the better summary. By comparing summary results over a broad population, we avoid many of the inherent flaws in other evaluation methods and provide a foundation for evaluating summarization tasks unlike any other method used to date.

6.1 Document Understanding Conference

In the area of automatic summarization, the main conference in recent years has been the Document Understanding Conference (DUC). The Document Understanding Conferences spawned from the SUMMAC conference in 2001 and recently became part of the Textual Analysis Conference (TAC) in 2008[41]. DUC was a series of evaluations conducted by the NIST (National Institute of Standards and Technology)[6]. The corpora for each conference were comprised of Newspaper/Newswire stories from numerous large and well-read international newspapers and news sources. The conference uses data sets in multiple languages which promotes development in many areas of Natural Language Processing in addition to summarization and also forces generalization of methods. Both test data and training data were provided for every conference. Manual summaries were generated to compare with the results of the automatic systems submitted to the conference.

Though there are many benefits to the DUC conferences, there are some intrinsic problems with this method. First off, the corpora of Newspaper/Newswire stories is systematically biased. No one would argue with the fact that an academic paper is written using a different voice than an article occurring in the New York Times (part of the DUC 2003

corpus). As I have mentioned previously, news stories are already a summary of an event that occurred and contain condensed information initially. Furthermore, in order to fit in a limited space, it is likely that documents will contain fewer forward references than an academic paper.

Karen Sparck Jones also brings up an interesting point in her article "Automatic summarising: The state of the art" about the "*extractive* paradigm" [14]. When summarizing a document, a human reader will often create an abstractive summary as opposed to an extractive summary so there will always be a disconnect between these methods and human generated summaries. Evaluations done in the DUC conferences are only for extractive summaries and do not deal with abstractive. Another interesting out-come of these conferences is that the move from internal systems-oriented evaluation to external purpose-oriented evaluation was more difficult than other NLP tasks. As Dr. Sparck Jones points out, in information retrieval, it is possible to evaluate an IR task well using only relevance metrics, and with machine translation, it is possible to rate according to text segment comparisons. Unfortunately for summarization, it is not possible to evaluate using only one way. Regardless, attempts have been made — especially with the ROUGE evaluation metrics — which fall more into the category of text segment comparisons as used in machine translation.

6.2 ROUGE

ROUGE has been the gold standard of evaluating text summaries in recent years. It is an acronym for Recall-Oriented Understudy for Gisting Evaluations [20]. ROUGE measures the similarities between summaries using a few different methods. They are: ROUGE-N, ROUGE-L, ROUGE-W, and ROUGE-S. The major concern in using ROUGE metrics is the fact that it compares summaries — implying that your corpus must have summaries already available to work with. This is one of the major benefits of the DUC conferences, because they provided numerous summaries to compare results to.

The first, ROUGE-N, "is an n-gram recall between a candidate summary and a set

of reference summaries." It is a recall measure where the numerator is the summation of the maximum number of n-grams in both a candidate summary and the set of reference summaries ($Count_{match}$). The denominator is the sum of all of the n-grams in the reference summaries. It is closely related to the machine translation measure BLEU.

$$\text{ROUGE-N} = \frac{\sum_{S \in ReferenceSummaries} \sum_{n-gram \in S} Count_{match}(n-gram)}{\sum_{S \in ReferenceSummaries} \sum_{n-gram \in S} Count(n-gram)} \tag{6.1}$$

The next two methods, ROUGE-L and ROUGE-W, deal with matching subsequences. ROUGE-L is simply the longest common subsequence. There are a variety of modifications that can be made to this method to account for different inputs such as sentences or long summaries, but the basic methodology remains the same. It is important to note that ROUGE-L does not require the subsequences to be continuous, but rather allows for other information to be interspersed within subsequence as long as the order remains the same. Thus, if the longest common sequence is "the car", a summary with "the car was ..." receives the same ROUGE-L weight as "the very, large, red car." In order to deal with this, another metric, ROUGE-W, is also used. ROUGE-W is a weighted longest common subsequence that keeps track of a variable k which is the length of the current consecutive match.

The final method, ROUGE-S, rates summaries based on skip-bigrams. A skip-bigram is any two words in a summary. Therefore, a sentence of length m has (m choose 2) possible skip-bigrams. The ROUGE-S score is calculated based upon the number of co-occurrences of skip-bigrams between a candidate summary and the reference summary, normalized over the number of skip-bigrams in the reference summary. As with the previous three methods, there are a few modifications that can be made to this method which are suited better for certain summarization tasks — such as creating a newspaper headline.

6.3 Mechanical Turk

Despite the many benefits of the ROUGE evaluation methods, this was not a feasible way to test my results. There are no mass, readily available summaries of Wikipedia articles in existence — and even if there were, it would not matter because it would be almost impossible to verify that the input texts were the same due to the fact that Wikipedia is a dynamic knowledge source. Furthermore, though my methods are very robust and extensible, the hypothesis of this thesis involves the knowledge available on Wikipedia so my results need to test summaries of Wikipedia articles. In order to use the ROUGE metrics, I would either need to create reference summaries manually or use a leading method in the field to automatically generate the reference set. As I have outlined above, I believed that the application of Latent Dirichlet Allocation to summarization would yield results at least as good as the leading methods in the field, if not better. Neither wanting to constrain myself to a reference set that might be inferior than my results, nor wanting to evaluate based on the DUC corpora which has fundamental, systematic differences from Wikipedia, I evaluated my results using a human population sample.

Continuing along the lines of the rest of the thesis, testing summaries generated from mass human knowledge would be best accomplished if evaluated by mass numbers of people. The company, Amazon.com, provides a service that is perfect for this called Mechanical Turk [1]. The premise behind the service is that humans are inherently better at performing certain tasks than machines, but it is often desirable to have an automated process for a large number of tasks. An easy example is that of deciding whether or not a picture of a human is male or female — a rather trivial task for a person, but very complicated for a computer. Essentially, the service allows individuals and companies to post jobs that would require a computer to pass a Turing Test in order to complete. The tag-line "Artificial Artificial Intelligence" refers to the fact that a mass number of people performing the same task appears automated even when the work being done is requires human intelligence as opposed to machine intelligence. People all around the world, though primarily in the

United States, can log onto the website https://mturk.com and sign-up to be a "worker" and perform "Human Intelligence Tasks" also known as "HITs." Workers are paid as consultants and generally receive fractions of a dollar for quick, simple tasks. This provided a perfect opportunity to test the results of my summarization algorithm.

Using my method, alongside NMF and TextRank, I generated summaries of 1,000 randomly selected Wikipedia articles. Each article was summarized into 10 sentence extractive summaries by all three algorithms. The resulting summaries were evaluated twenty times each using the Mechanical Turk service discussed above. For completing a comparison, a worker was paid $0.02. A worker is not allowed to review the same topic twice, so a single worker could have at most evaluated all 1,000 article summaries, though this did not occur. From my perspective, I received 20,000 data points to evaluate the results of my methods equating to twenty points for each article. Thus, if a certain method worked better on certain articles, I would be able to figure this out and as a whole evaluate if this was a systematic trend or not. Furthermore, with twenty data points per article, I had a statistically significant sample to decide which algorithm worked better.

When a worker accepted one of my HITs, he or she was presented a page that asked which summary they would prefer to read if they did not want to read the full Wikipedia article on the subject. The template is available in appendix B. They were presented with three links that opened up three new windows with the one summary from each method. To ensure a valid experiment, one-third of the articles had the first link correspond to NMF summaries, one-third of the articles had the first link correspond to LDA summaries, and the final one-third of the articles had the first link correspond to TextRank summaries. Thus, if there is a bias with people either only clicking on the first link either due to an Anchoring Bias[1] or because they realized it would be difficult for me to verify if they read the links, it would not affect my results. The only effect from this would be that the variance in my results would appear smaller than it actually is. Along this same line of logic, if a slightly more sophisticated malevolent worker decided to randomly select which summary

[1]See Bazerman and Moore, Chapter 2 for common decision making biases

was better, the same results would apply. Though the reward for evaluating each summary is not very great, workers are concerned with their approval ratings which allow them to do tasks. A requester can review the work before a worker gets paid and decide whether or not the worker actually did the task. Thus, I hypothesize that a worker would look at the risk versus reward and decide not to do this task than risk affecting their approval rating.

Evaluating on a large human population provides a paradigm-shifting method of evaluating summarization methods. Unlike any other methods out there, this overcomes many of the deficiencies persistent in current evaluation systems. The mass aspect of the judges will minimize systematic biases that one person may have and illuminate common factors between them — much like the multi-document aspect of LDA. Conventional NLP metrics are insufficient in evaluating summarization tasks and due to the scale factor, this is a completely new method that solves many of the deficiencies.

Chapter 7

Results

Results for the three different summarization methods were evaluated using Mechanical Turk. Each summarization algorithm generated summaries for the same one thousand randomly chosen Wikipedia articles. Each article was compared twenty times using workers on the Mechanical Turk website. The overall results show that SLDA was preferred the most times over the other two methods. The two methods that were based upon document topic modeling, SLDA and NMF, were shown to be better than TextRank. The results are shown to be statistically significant.

My results show that Summarization by Latent Dirichlet allocation is a top performing method for generating single-document extractive summaries. Modeling subtopics in a document is a successful approach to creating summaries. Furthermore, the new evaluation model outlined in this paper is shown to be effective and better than random selection.

7.1 Summary Examples

Before explaining the results of the Mechanical Turk evaluations, here are the results and summaries of the actual methods for a random article "Z-DNA" created by the three systems. As contrast, ten random sentences are chosen from the source input to demonstrate the superiority of these methods over mere probabilities. The full source text for this article

is available in Appendix C.

Example 1: Summarization by Latent Dirichlet Allocation

> Z-DNA is one of the many possible double helical structures of dna. It was resolved as a left-handed double helix with two anti-parallel chains that were held together by Watson-Crick base pairs (see: x-ray crystallography). Z-DNA was the first single-crystal X-ray structure of a DNA fragment (a self-complementary DNA hexamer d(CG) 3 /). Formation of this structure is generally unfavourable, although certain conditions can promote it; such as alternating purine -pyrimidine sequence (especially poly(dGC) 2 /), negative dna supercoil ing or low salt and some cation s (all at physiological temperature, 37C, and pH 7. The Z-DNA conformation has been difficult to study because it does not exist as a stable feature of the double helix. While no definitive biological significance of Z-DNA has been found, it is commonly believed to provide torsional strain relief (supercoiling) while DNA transcription occurs. In 2007, the rna version of Z-DNA, z-rna, was described as a transformed version of an a-rna double helix into a left-handed helix. An algorithm for predicting the propensity of DNA to flip from the B-form to the Z-form, ZHunt, was written by Dr. This algorithm was later developed by tracy camp, p. It was solved by Andrew Wang, alexander rich, and co-workers in 1979 at mit.

Example 2: Non-negative Matrix Factorization

> Z-DNA is one of the many possible double helical structures of dna. null The Z-DNA helix is left-handed and has a structure that repeats every 2 base pairs. It was resolved as a left-handed double helix with two anti-parallel chains that were held together by Watson-Crick base pairs (see: x-ray crystallography). It is a left-handed double helical structure in which the double helix winds to the left in a zig-zag pattern (instead of to the right, like the more common b-dna form). In 2007, the rna version of Z-DNA, z-rna, was described as a transformed version of an a-rna double helix into a left-handed helix. Z-DNA is quite different from the right-handed forms. Formation of this structure is generally unfavourable, although certain conditions can promote it; such as alternating purine -pyrimidine sequence (especially poly(dGC) 2 /), negative dna supercoil ing or low salt and some cation s (all at physiological temperature, 37C, and pH 7.3-7.4). Instead, it is a transient structure that is occasionally induced by biological activity and then quickly disappears. Z-DNA is thought to be one of three biologically active double helical structures along with a- and B-DNA.

Example 3: TextRank

> Z-DNA is one of the many possible double helical structures of dna. It is a left-handed double helical structure in which the double helix winds to the left

> in a zig-zag pattern (instead of to the right, like the more common b-dna form). Z-DNA is thought to be one of three biologically active double helical structures along with a- and B-DNA. Z-DNA was the first single-crystal X-ray structure of a DNA fragment (a self-complementary DNA hexamer d(CG) 3 /). Whenever a segment of Z-DNA forms, there must be B-Z junctions at its two ends, interfacing it to the B-form of DNA found in the rest of the genome. The crystallisation of a B- to Z-DNA junction in 2005 provided a better understanding of the potential role Z-DNA plays in cells. It was resolved as a left-handed double helix with two anti-parallel chains that were held together by Watson-Crick base pairs (see: x-ray crystallography). In 2007, the rna version of Z-DNA, z-rna, was described as a transformed version of an a-rna double helix into a left-handed helix. It was solved by Andrew Wang, alexander rich, and co-workers in 1979 at mit. Formation of this structure is generally unfavourable, although certain conditions can promote it; such as alternating purine -pyrimidine sequence (especially poly(dGC) 2 /), negative dna supercoil ing or low salt and some cation s (all at physiological temperature, 37C, and pH 7.3-7.4).

Example 4: Random Sentence Selcection

> 3-7. 56 nm) —- —- — – ? —-! oregonstate. 38 nm) —- The Z-DNA conformation has been difficult to study because it does not exist as a stable feature of the double helix.

As can be seen from all three of these examples, the summaries generated are much better than the randomly generated sentences. Though great care was taken to ensure that the initial XML files were properly cleaned into ascii text and that formatting characters were removed, certain structures such as tables had a variable format that made it quite difficult to catch every case as demonstrated by the random sentences in example 4. In this case, there were numerous quotation marks that caused the formatting to be processed verbatim by the scripts. In addition, this further shows the strength of the algorithms demonstrated because they were able to deal with the non-standard input sufficiently and ignore “sentences” that would not be useful to the summary. Additionally, a look at the source text in Appendix C clearly demonstrates how successful and sophisticated these three methods are.

Another interesting thing to note about these summaries is that they are very similar which means a few things for my methods. First off, SLDA is comparable to top methods

in the field of summarization. Second, with my new evaluation scheme, small differences between methods are aggregated over many trials and with many documents to decide which one is better — something that would not be as evident using a gold-standard comparison.

7.2 Statistics

Results from the Mechanical Turk evaluation are shown in table 7.1. Overall, the SLDA method received the most selections. Non-negative Matrix Factorization also performed well, but TextRank was not rated as well. Though workers were presented with the choice to choose the result that the surveys were indistinguishable, only 115 of the overall responses indicated this, or 0.5%. This does not eliminate the possibility that people were randomly selecting their choices and decided not to click on the indistinguishable option thinking I might not notice.

In order to test statistical significance, I ran a Chi-square test. First, I removed the 115 selections that did not rate a preference for a total sample size of 19,884. The null hypothesis is that any single one of the summaries was chosen at random and therefore should be one third of the sample size or 6,628. There are three different models being evaluated which means that there are two degrees of freedom. From this constraint, the χ^2 value was 5.84. At a p-value of 0.05, the χ^2 value to reject the null hypothesis is 5.99. Though this would clearly pass at a p-value of 0.10, it is possible that some people randomly clicked and could be removed from the data set.

Table 7.1: Overall Responses to Mechanical Turk Evaluation

Algorithm	Number of Responses
SLDA	6,742
NMF	6,669
TextRank	6,473
No Choice	115
Total	19,999

Removing values from the data set is quite tricky because there is little to determine what was actually random and what was not. Instead of viewing the problem in this way, I decided that I would only accept evaluations where the worker spent more than ten seconds on a HIT. Each HIT involves clicking on three different URLs and then deciding which one would better summarize the given title. It is unreasonable to assume that a worker did this in ten seconds or less, so I eliminated those evaluations from my data set (table 7.2) and ran another chi-square test. After eliminating those evaluations, I still had 8,032 results to compare. Of this new set, thirty had not chosen a method. Again testing whether the algorithms were chosen at random, I also removed those thirty from the set leaving 8,002 total or 2,667.33 for the expected value in the null hypothesis. This time, the χ^2 value was 32.43. A p-value of 0.001 has a χ^2 value of only 13.82 — indicating that we should reject the null hypothesis and accept that the ranking of the methods was not chosen at random.

Table 7.2: Responses to Mechanical Turk Evaluation taking more than 10 seconds

Algorithm	Number of Responses
SLDA	2,812
NMF	2,761
TextRank	2,429
No Choice	30
Total	8,032

Due to the fact that the articles were randomly presented to workers in the survey, we

can conclude that biases associated with the presentation of the survey would not have affected the results. Similarly, workers who selected one choice consistently over the others without actually reading the summaries would not affect the results either.

In both evaluations of the data set, the SLDA method performed the best, followed by NMF. TextRank performed the worst overall. We are able to conclude from the chi-square tests that the results were not generated at random and conclude that the TextRank algorithm is inferior to the other two methods on this corpus. Though SLDA was rated higher than NMF, the values for TextRank are the furthest from the expected values. This indicates that it could have been the deciding factor in our previous chi-square tests. Essentially, this states that the ratings of the TextRank algorithm were influential enough to cause us to reject the null hypothesis that the results were generated at random. Therefore, it is necessary to compare each method against the others individually. It is worthwhile to note here that the conclusions drawn from this are not as strong as in the previous section because it is difficult to account for all possible variables. Having previously seen the TextRank summaries may have biased a worker in such a way that he or she was then predisposed to select SLDA over NMF when that would not have been the case if only the two were provided initially. Nonetheless, it is worth pursuing these results further and seeing whether or not we can decisively state which algorithm is the best for this corpus. Using the responses of more than ten seconds, I compared each method against the other two individually. The null hypothesis still remains that the summary rankings were randomly selected and that equates to an expected value of the mean of the number of selections of the two methods. As the results in table 7.3 clearly show, both LDA and NMF were statistically superior to TextRank when compared individually, but compared to each other the null hypothesis is accepted.

Table 7.3: χ^2 values of Individual Comparisons

	SLDA	NMF	TextRank
SLDA		0.46	27.98
NMF	0.46		221.23
TextRank	27.98	21.23	

Chapter 8

Conclusion

Overall, this was a very successful thesis. I demonstrated Summarization by Latent Dirichlet allocation as a leader in the field of summarization. I introduced a radically new way to evaluate summarization methods. I also showed that summarization methods that are based upon modeling the subtopics of a document perform better than one of the leading graph based approaches in the field. Finally, I also demonstrated that by blurring the line defining single-document and multi-document summarization that traditional classifications within summarization are not necessarily distinguishable, opening the way for future work.

8.1 Future Work

There is a lot of potential for future work based upon this research. One of the most interesting parts about Wikipedia is that there exists links between pages that indicate related topics. An interesting addition to my research would be to make use of this information. Even extracting sentences from a linked article and not the initial, would blur the lines between extractive and abstractive summarization. Information not directly included in the original article could be included in a summary, but it still is not necessarily abstraction. This is how I perceive that the most progress in this field will be made in the next decade — not only using corpora similar to Wikipedia, but rather using the ever increasing

amounts of available information outside of the corpora to include in the summary. I foresee generic summaries based upon Information Retrieval tasks and query-focused summaries that go beyond the original single-document summary methods used here. I am currently researching the effects of using the link structure within Wikipedia in summary generation, but it was not included in this paper due to time-constraints, budget-constraints, and storage-constraints. My next work intends to transcend these limitations.

In addition to the above, there is a lot of potential to modify these methods. For instance, Wikipedia is available in multiple languages and the only thing that is specific to English in the above methods is the definition of a sentence (punctuation in other languages can be different) and the list of stop words. It would be interesting to see how the system performs on other languages. Furthermore, due to the fact that many articles are written both in English and another language, it would be easy to modify some of the methods outlined here to experiment with the Machine Translation subfield of Natural Language Processing.

Summarization by Latent Dirichlet allocation can also be applied to many other corpora outside of Wikipedia. If SLDA is to be accepted as a leading algorithm in the field, it needs to be evaluated on many more data sets. It is a generic algorithm and I hypothesize that it will perform very well on almost any input source. Further comparisons directly against NMF summarization would be useful because they are both attempting to model subtopics in documents.

Additionally, the evaluation method defined in this paper has many potential applications in comparing other summarization methods or even extending to other NLP task evaluations. It is a paradigm shifting method that can have huge implications in defining what is subjectively good in many different applications.

8.2 Accomplishments

A big accomplishment of this work was that topic modeling of documents is a good method for automatic summarization. Additionally, methods successful in other areas of Natural Language Processing will likely work well within summarization and vice versa. By questioning the classical classifications of summarization, I have opened up more areas to study within summarization that could lead to major breakthroughs that had not previously been thought of. I redefined the way automatic summaries are evaluated and found some solutions to deficiencies in traditional evaluation.

Finally, I demonstrated that given two other leading methods in the field of summarization, that SLDA could create a summary that was chosen as the best selection the most times in a sample of 20,000. Summarization by Latent Dirichlet allocation was proven as a successful model for summarization tasks and will likely be investigated further along with causing researchers to try and apply LDA to even more tasks. In conclusion, SLDA was a major success and will transform how researchers view automatic summarization in the future.

Bibliography

[1] Mechanical turk, April 2009. `http://mturk.com/`.

[2] Wikipedia, April 2009. `http://www.wikipedia.org/`.

[3] Max H. Bazerman and Don A. Moore. *Judgement in Managerial Decision Making*. John Wiley and Sons, 7th edition, 2008.

[4] Michael W. Berry, Murray Browne, Amy N. Langville, V. Paul Pauca, and Robert J. Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. *Computational Statistics and Data Analysis*, 52(1):155 – 173, 2007.

[5] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993 – 1022, January 2003.

[6] Document Understanding Conference. Document understanding conference website. `http://duc.nist.gov/`.

[7] Daniel M. Dunlavy, Dianne P. O'Leary, John M. Conroy, and Judith D. Schlesinger. Qcs: A system for querying, clustering and summarizing documents. *Information Processing and Management*, 43(6):1588 – 1605, 2007. Text Summarization.

[8] H. P. Edmundson. New methods in automatic extracting. In *Advances in Automatic Text Summarization*. MIT, 1969.

[9] Gunes Erkan and Dragomir R. Radev. Lexrank: Graph-based centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22, 2004.

[10] Sanda Harabagiu, Andrew Hickl, and Finley Lacatusu. Satisfying information needs with multi-document summaries. *Information Processing and Management*, 43(6):1619 – 1642, 2007. Text Summarization.

[11] Stacy President Hobson, Bonnie J. Dorr, Christof Monz, and Richard Schwartz. Task-based evaluation of text summarization using relevance prediction. *Information Processing and Management*, 43(6):1482 – 1499, 2007. Text Summarization.

[12] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1), 1972.

[13] Karen Sparck Jones. Automatic summarizing: factors and directions. In *Advances in Automatic Text Summarization*. MIT, 1999.

[14] Karen Sprck Jones. Automatic summarising: The state of the art. *Information Processing and Management*, 43(6):1449 – 1481, 2007. Text Summarization.

[15] Jon Kleinberg. Authoritative sources in a hyperlinked environment. In *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms*, 1999.

[16] Daniel D. Lee and Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401, October 1999.

[17] Daniel D. Lee and Sebastian Seung. Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems*, 13, 2001.

[18] Ju-Hong Lee, Sun Park, Chan-Min Ahn, and Daeho Kim. Automatic generic document summarization based on non-negative matrix factorization. *Information Processing and Management*, 45(1):20 – 34, 2009.

[19] Shao Fen Liang, Siobhan Devlin, and John Tait. Investigating sentence weighting components for automatic summarisation. *Information Processing and Management*, 43(1):146 – 153, 2007.

[20] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Proceedings of the Workshop on Text Summarization.*

[21] H. P. Luhn. The automatic creation of literature abstracts. In *Advances in Automatic Text Summarization.* MIT, 1958.

[22] Stacy K. Lukins, Nicholas A. Kraft, and Letha H. Etzkorn. Source code retrieval for bug localization using latent dirichlet allocation. In *Proceedings of the 2008 15th Working Conference on Reverse Engineering*, 2008.

[23] Inderjeet Mani. *Automatic Summarization.* J. Benjamins Pub. Co., 3 edition, 2001.

[24] Inderjeet Mani and Mark T. Maybury. *Advances in Automatic Text Sumamrization.* MIT Press, 1999.

[25] Christopher D. Manning and Hinrich Schutze. *Foundations of Statistical Natural Language Processing.* MIT Press, sixth edition, 1999.

[26] Rada Mihalcea. Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, July 2004.

[27] Rada Mihalcea. Language independent extractive summarization. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, June 2005.

[28] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, July 2004.

[29] Thomas P. Minka. Estimating a dirichlet distribution. Technical report, MIT, 2000.

[30] Indraneel Mukherjee and David M. Blei. Relative performance guarantees for approximate inference in latent dirichlet allocation. *Neural Information Processing Systems*, 2009.

[31] Jahna Otterbacher, Gunes Erkan, and Dragomir R. Radev. Using random walks for question-focused sentence retrieval. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, October 2005.

[32] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[33] Chris D. Paice and Paul A. Jones. The identification of important concepts in highly structured technical papers. In *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Retrieval*, 1993.

[34] G. J. Rath, A. Resnick, and T. R. Savage. The formation of abstracts by the selection of sentences. In *Advances in Automatic Text Summarization*. MIT, 1961.

[35] Xiaojun Wan, Jianwu Yang, and Jianguo Xiao. Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 552 – 559, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[36] Dongshan Xing and Mark Girolami. Employing latent dirichlet allocation for fraud detection in telecommunications. *Pattern Recognition Letters*, 28(13):1727 – 1734, 2007.

[37] Shiren Ye, Tat-Seng Chua, Min-Yen Kan, and Long Qiu. Document concept lattice for text understanding and summarization. *Information Processing and Management*, 43(6):1643 – 1662, 2007. Text Summarization.

[38] Jen-Yuan Yeh, Hao-Ren Ke, Wei-Pang Yang, and I-Heng Meng. Text summarization using a trainable summarizer and latent semantic analysis. *Information Processing and Management*, 41(1):75 – 95, 2005. An Asian Digital Libraries Perspective.

[39] David Zajic, Bonnie J. Dorr, Jimmy Lin, and Richard Schwartz. Multi-candidate reduction: Sentence compression as a tool for document summarization tasks. *Information Processing and Management*, 43(6):1549 – 1570, 2007. Text Summarization.

[40] Benyu Zhang, Hua Li, Yi Liu, Lei Ji, Wensi Xi, Weiguo Fan, Zheng Chen, and Wei-Ying Ma. Improving web search results using affinity graph. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, 2005.

[41] Lin Zhao, Lide Wu, and Xuanjing Huang. Using query expansion in graph-based approach for query-focused multi-document summarization. *Information Processing and Management*, 45(1):35 – 41, 2009.

Appendix A

List of Stop Words

The stop list of English words here is taken from Manning and Schutze page 533 [25] with a few words added. Due to the intrinsic nature of the corpus and the task, there were some words that occurred more often than the broad and general list given in this text. The words "is" and "are" will not make much difference in evaluating summaries so they were added to the stop list. Also, the word "Wikipedia" was added to the stop list because this corpus was very biased towards that word. The complete stop list used in my system is as follows:

Table A.1: List of Stop Words

a	also	an	and	are
as	at	be	but	by
can	could	do	for	from
go	have	he	her	here
his	how	i	if	in
into	is	it	its	my
of	on	or	our	say
she	that	the	their	there
therefore	they	this	these	those
through	to	until	we	what
when	where	which	while	who
wikipedia	with	would	you	your

Appendix B

Mechanical Turk Question Template

Which Summary of a Wikipedia Article is Better?

If you did not feel like reading a full Wikipedia Article, which of these summaries would you prefer to read instead?

${term}

Decide which summary is the best. Click on the three links to view them (in new window).

Summary 1

Summary 2

Summary 3

- ○ Summary 1 is best
- ○ Summary 2 is best
- ○ Summary 3 is best
- ○ Impossible to Differentiate

Submit

Appendix C

Z-DNA Source Text

Here is the source text from the article on Z-DNA. As mentioned in the main paper, lots of work was done to ensure that source texts were properly cleaned. Nonetheless, certain formatting characters still got through. The main reason for this document is that the table contains lots of quotation marks which the scripts interpreted as being actual quotes and did not process them. Overall, this is more a testament to the algorithms defined because they were able to ignore unique cases such as this.

> Z-DNA is one of the many possible double helical structures of dna. It is a left-handed double helical structure in which the double helix winds to the left in a zig-zag pattern (instead of to the right, like the more common b-dna form). Z-DNA is thought to be one of three biologically active double helical structures along with a- and B-DNA. Z-DNA was the first single-crystal X-ray structure of a DNA fragment (a self-complementary DNA hexamer d(CG) 3 /). It was resolved as a left-handed double helix with two anti-parallel chains that were held together by Watson-Crick base pairs (see: x-ray crystallography). It was solved by Andrew Wang, alexander rich, and co-workers in 1979 at mit. The crystallisation of a B- to Z-DNA junction in 2005 provided a better understanding of the potential role Z-DNA plays in cells. Whenever a segment of Z-DNA forms, there must be B-Z junctions at its two ends, interfacing it to the B-form

of DNA found in the rest of the genome. In 2007, the rna version of Z-DNA, z-rna, was described as a transformed version of an a-rna double helix into a left-handed helix. Z-DNA is quite different from the right-handed forms. In fact, Z-DNA is often compared against B-DNA in order to illustrate the major differences. The Z-DNA helix is left-handed and has a structure that repeats every 2 base pairs. The major and minor grooves, unlike A- and B-DNA, show little difference in width. Formation of this structure is generally unfavourable, although certain conditions can promote it; such as alternating purine -pyrimidine sequence (especially poly(dGC) 2 /), negative dna supercoil ing or low salt and some cation s (all at physiological temperature, 37C, and pH 7.3-7.4). Z-DNA can form a junction with B-DNA in a structure which involves the extrusion of a base pair. The Z-DNA conformation has been difficult to study because it does not exist as a stable feature of the double helix. Instead, it is a transient structure that is occasionally induced by biological activity and then quickly disappears. It is possible to predict the likelihood of a DNA sequence forming a Z-DNA structure. An algorithm for predicting the propensity of DNA to flip from the B-form to the Z-form, ZHunt, was written by Dr. p. shing ho in 1984 (at MIT). This algorithm was later developed by tracy camp, p. christoph champ, sandor maurice, and jeffrey m. vargason for genome-wide mapping of Z-DNA (with P. Shing Ho as the principal investigator). Z-Hunt is available at [http://gac-web.cgrb.oregonstate.edu/zDNA/ Z-Hunt online]. While no definitive biological significance of Z-DNA has been found, it is commonly believed to provide torsional strain relief (supercoiling) while DNA transcription occurs. The potential to form a Z-DNA structure also correlates with regions of active transcription. A comparison of regions with a high sequence-dependent, predicted propensity to form Z-DNA in human chromosome 22 with a selected set of known gene transcription sites suggests there is a correlation.adar" Biophysicist Alexander Rich of the Massachusetts Institute of Technology in Cam-

bridge never doubted the relevance of Z-DNA, which he and colleagues unveiled in 1979 using x-ray crystallography. In 2003, his research team noticed that a poxvirus virulence factor, called E3L, mimicked a mammalian protein that binds Z-DNA. In 2005, Rich and his colleagues pinned down what E3L does for the poxvirus. When expressed in human cells, E3L increases by five- to 10-fold the production of several genes that block a cells ability to self-destruct in response to infection. Rich speculates that the Z-DNA is necessary for transcription and that E3L stabilizes the Z-DNA, thus prolonging expression of the anti-apoptotic genes. He suggests that a small molecule that interferes with the E3L binding to Z-DNA could thwart the activation of these genes and help protect people from pox infections. — class"wikitable" —-!Geometry attribute!A-form!B-form!Z-form —- —Helix sense ——align"center"— right-handed ——align"center"— right-handed ——align"center"— left-handed —- —Repeating unit ——align"right"— 1 bp ——align"right"— 1 bp ——align"right"— 2 bp —- —Rotation/bp ——align"right"— 32.7 ——align"right"— 35.9 ——align"right"— 60/2 —- —bp/turn ——align"right"— 11 ——align"right"— 10.5 ——align"right"— 12 —- —Inclination of bp to axis ——align"right"— +19 ——align"right"— 1.2 ——align"right"— 9 —- —Rise/bp along axis ——align"right"— 2.3 (0.23 nm)——align"right"— 3.32 (0.332 nm)——align"right"— 3.8 (0.38 nm) —- —Pitch/turn of helix ——align"right"— 28.2 (2.82 nm)——align"right"— 33.2 (3.32 nm)——align"right"— 45.6 (4.56 nm) —- —Mean propeller twist ——align"right"— +18 ——align"right"— +16 ——align"right"— 0 —- —Glycosyl angle ——align"center"— anti ——align"center"— anti ——align"center"— C: anti, brgt; G: syn —- —Sugar pucker ——align"center"— C3'-endo ——align"center"— C2'-endo ——align"center"— C: C2'-endo, brgt;G: C2'-exo —- —Diameter ——align"right"— 23 (2.3 nm)——align"right"— 20 (2.0 nm)——align"right"— 18 (1.8 nm) —- —colspan"4"—Sources: — –

Appendix D

List of Summarized Wikipedia Articles

Table D.1: List of Summarized Articles

1204	1217	1265	1614
1622	1637	1651	1758
1895	1906	1961	1990
24-dinitrophenol	8x8	aaliyah	aashayein
abhidharma	abu-	accenture	accessibility
acosta	acrobunch	actinomycosis	adecco
adium	adoption	aftenposten	agag
agami	agent	aggadah	aguardiente
ahool	aissawa	akhenaten	al-bassa
alanine	alchemy	alcimoennis	alcuin
algae	alpha-fetoprotein	alphatia	alsamixer
alternity	alzounoub	amaragadhi	amargasaurus
ambiguity	amblyopone	ammolite	ammonia
anaikot	androcles	angariones	anti-mormonism
appenzell	apsis	apura	aputheatre
aragorn	archaeology	archivist	aristotelianism
arkadimon	armimex	arnhem	artimation
asanee-wasan	aschersleben	ashhurst	asparagine
assen	associationalism	astereae	astrophotography
atarisoft	atavus	athenaeum	auctoritas
autococker	autostrada	avanigadda	avebury
avnoj	azzone	babeland	badarri

badfinger	bagbana	ballachulish	ballater
ballens	ballenstedt	ballintra	balting
banasura	banguele	banjara	barghawata
barsauli	bazartu	bedhaya	belinus
benedicaria	benvolio	beriberi	bhubaneswar
bigha	biography	biolinguistics	biopolis
birdcage	bitlis	blockhosts	blodeuwedd
bloemfontein	bloomability	boffin	boletaceae
bonnington	bothwell	botola	bradypodion
brain-	branigan	brdy	breakingviews
britwell	bromley	bromocriptine	brookeborough
bta-6	bufori	buick	bukovina
burdak	buritis	buru	burzenland
bvn	bwin	c-base	cagayan
calasca-castiglione	caldones	calf	callbox
camaldolese	cameco	cameri	candiolo
canouan	capicola	casteldidone	castlebay
castletownroche	cataphract	catmull	cauo
cawood	cayenne	cblt	celsius
cephalon	cetiosaurus	chaining	chair
challedon	changeup	chapati	charly
chat	chillits	chink	chmp-fm
chom-fm	chopard	chorale	chrysant
churwell	citynews	cjay-fm	cloonfush
codevilla	codonopsis	coimbra	colletorto
commonplace	company	comparative	comsec
connexin	coprophilia	cordic	corfe
cornmeal	costermonger	cotopaxi	courchevel
cowes	cowpea	creote	cross-selling
cross-wes	cryosurgery	crystalis	cryx
cumbarjua	curculin	cuso	cyber-ark
cybernator	cybiko	cyclophane	czersk
dadabadi	daemonfey	daivadnya	dalregementet
danja	day-fine	dbcs	debora
decartelization	deconstruction	deebo	deicide
delphinium	demonland	derivatization	desperadas
destrii	devs	dhammacakkapp	diagram
diemoth	digestion	dillichaur	dingli
diplom	diponegoro	dischidia	disemb
divinity	djerba	dodecagon	doenjang
dolgellau	dondurma	dourou	dream-hunters
drive2gether	drying	dufourspitze	duklja
dulcinian	dunloy	dwarakesh	dwup-fm
dyxx-tv	e.tv	e21c	easyrider
ebbo	echinoderm	eco-in	ecolinguistics

edu
efflorescence
efrafa
egeus
eia-708
ekwendeni
elbistan
eliduc
emmer
emsland
englandspiel
entamoeba
enthronement
ephedrine
epikleros
ergoloid
ergotamine
eriskay
erotophobia
escats
eschatology
esterel
estradiol
eudoxis
euphemism
eurocentrism
exalted
exile
exocet
extrav
fakenham
falconview
faline
farrukhsiyar
favignana
fedikhola
ferdinand
ferrimagnetism
ferroics
fieseler
fight-o
filicide
film4
findhorn
flambards
flamenco
flatbow
flax
flicky
flushwork
flypast
focke-wulf
folia
follyfoot
force-feeding
formal
freelancer
freighthopping
frobeni
fructidor
fulham
furl
fyresdal
g-a-y
gabrie
gachibowli
gadag-betigeri
gaillardia
gamosa
ganglioside
ganiga
garff
garhwali
gas-guzzler
gatomon
gavalou
geetha
geordie
gerbera
germane
gestratz
ghostkeeper
ghurni
gimzo
gizmo5
glaurung
glycerophospholipid
goblini
gobowen
gokak
gornate-olona
grande-terre
grapsidae
grimspound
grossglockner
groupthink
gualtieri
guzman
gyantse
gyruss
haemophilia
hagwon
halbarad
halebidu
hallucination
handcycle
handoff
haninah
hanukkah
hanwell
harkat-ul-jihad-al-islami
harpy
hashid
hattusa
haworthia
hedge
helina
hellevoetsluis
hemiscylliidae
herbad
herborn
herihor
hertz
herzliya
hexamine
hilandar
hindutva
hits!
hoffa
homeobox
homicide
homogenization
hooverville
huangjiu
huayl
huedin
huitzilopochtli
humanzee
humongous
hypergammaglobulinemia
hyperparameter
hypnos
hypochlorite
ic4a
icr
idioteque
iloveyou
impartiality
inch
incrementalism
indige
injil
inotify
inque
intrade
iochroma
iqaluit
iserlohn
islam
isojoki
isolat
isotropy
itaguaru
ittre
j-break
jacobitism
jaffna
jalgaon
jethrahiya
jingnan
jinju
jogannath
jujyfruits
july
jumpgate
junkers
jurisd
kalapanakuioiomoa
kalevala
kamancheh

kamarak	kamx	kaninchen	karakol
kartli	kaskus	kathmandu	kathoey
katorga	kattuputhur	katunjebawala	kavminvodyavia
kcos-lp	keepass	keeshond	kenite
kentmere	kerry	kfog	khangars
kharukyanhi	khorugh	khtc	killerpilze
kilmartin	kimari	kishangarh	kitab-verlag
kitchen	klac	knoebels	koleda
kologo	konnagar	konvas	koseki
kostroma	koussan	kpxm	kqds-tv
kripa	kscf	ksitigarbha	ktnf
kumasi	kurtha	kymco	labialisation
lamhirh	lamivudine	lamotte	langstone
larnaca	lasgun	laterina	lawapa
laysan	league	lebrija	lechmere
lectionary	leicester	lelydorp	leskovac
leukemia	lictor	liebl	lightmap
linamar	lincomycin	lipis	listowel
llwynypia	llywelyn	lokeren	longbenton
lorne	loveshy	lowestoft	lucian
lucknow	lugger	lyphard	lyre
m-dot	maastunnel	madhavaram	madhesh
mag-7	maggiora	magnetism	magnitogorsk
maharahj	maidenhead	malabathrum	maliki
maling	mamurras	mana	manchukuo
mange	mappillai	margrave	mariel
marincello	marksman	marthandam	martock
marwanid	marwat	matangi	mauerpark
mauja	mcarthu	mecca-cola	meketaten
melamed	melor	melsungen	melter
meris	merupuri	mesothelae	mestanza
meurthe	microdeal	microdisney	microserfs
mineiros	minitrack	minturno	miraj
misdirection	miser	mizan	modiba
modularity	momus	mondia	monrupino
montegridolfo	montemignaio	morifade	morning
motilin	mucilage	multiav	multicategory
murter	musha	myachi	mysecurecyberspace
n-terminus	n-universes	naburimannu	nadeem-shravan
nagios	naiman-beg	nakskov	nanepashemet
nanobe	naplps	narayangaon	nasa
naturalism	navaratna	nazism	neo-blox
neofeudalism	neoism	nesophontes	netrakali
neuroacanthocytosis	nicktropolis	nicotiana	nidor
nidwalden	nieheim	nizhalkuthu	nmvtis

non-repudiation	npc	nuakhai	nundhaki
obwalden	oceanography	ocotea	odenwald
ofeq	ogrish.com	okhla	omitara
openldap	operon	oppland	orahovac
orhei	orkin	orthoptera	osker
oslolosen	osteochondrodysplasia	otahuhu	otopeni
oumpah-pah	outfielder	overdetermination	oxymorphone
palavakkam	pallantium	panaji	panzerwaffe
paramagnetism	paramore	parke-davis	parmenion
peckforton	peleus	peltigera	pemalite
pencaitland	pened	peperomia	percodan
perforce	perspectivism	perverb	petitiononline
petrophysics	pharaoh	phenylbutazone	philately
philipp-reis	photoanimation	pianoro	pienza
pilotta	pinarello	piozzano	plantain
plasmid	pleaching	pneumothorax	podocarpaceae
poibrene	polonnaruwa	polymarchs	pomfret
poodle	porch	pornography	possum
poweranimator	preconditioner	prediction	presbyphagia
priapism	pridne	primaris	pro-verb
prolactinoma	prolotherapy	promelectronica	proporz
pseudo-seneca	pseudofeces	psychohistory	psychro
puppeteer	pyaasa	qaitbay	qoph
qsig	quarkonium	queenie	quetzalcoatlus
quim	quraysh	qwest	rahimabad
rakia	ranikhet	rapsoul	rasayana
rashmirathi	rawmarsh	rawmill	rayzz
razgrad	redlist	redstockings	reggae
regiomontanus	rendu	restorationism	retablo
rhotacism	ridgehead	riposto	rollback
ronquil	ronsecco	rootes	rosariazo
rotation	rotisserie	rovia	royden
rubano	rubric	ruhengeri	rumia
rummy	runamuck	runebound	runway
rushden	ruthin	sabbateans	sadhu
sadlermiut	sahg	saimin	sainte-chapelle
sainte-marie-sur-mer	saldaea	saltator	samaja
sampoerna	sanctus	sannidal	santhenar
sarissa	sarnico	satsix	sauraha
saxe-gotha-altenburg	saxitoxin	sbb	sberbank
scantlebury	sceat	scouting	seacroft
sealdah	seibal	sejmik	selenite
seokgatap	shadowslayers	shakargarh	shamrain
shinran	shippalgi	shoshenq	shuttlecock
sialidosis	siepac	sikatahan	sinestro

sinsheim	sinusitis	sisal	sittard
siyahamba	skaro	skewbald	sociotherapy
sokolni	solutia	sonambient	sopara
souba	soundism	sparkford	sparty
spatha	spice	spiddal	spin-flip
sporophyte	sprinkles	starlicide	starspot
starveillance	steveless	stewartby	stornoway
storyville	stotting	stowey	streamsql
studland	sudaan	sudhagad	suitport
suliban	sunscreem	supercomputer	superstation
superuser	surjaha	surveillance	swordbearer
synteny	t-55agm	tadawul	taichung
taiheiki	takaloo	talamona	talawang
tamale	tampopo	tanapox	tanygrisiau
tap	tapper	taranchi	taxome
telegony	telemusik	telnet	tenchijin
tendinitis	tennikoit	teratornis	ternate
terrassa	thallium	thanatos	thayer
thelwall	thimphu	thirunamam	thrones
throwdown	thuvakudi	thyroxine	tibbetibaba
tihany	tikoloshe	timaru	tinder
tiruththanka	tlaquepaque	tlatlaya	toilet
tomyris	tonality	transgrid	trethomas
trilobyte	trisagion	truphone	truss
trypanophobia	ty5	typo3	uddhava
udhailiyah	ugod	uhlsport	ulsterbus
ultan	umbria	underwater	unfa
unicap	unto	upholstery	urarina
uropeltis	vaccaei	vakill	vallisneria
varadero	vashti	vastogirardi	vector
vectrex	venzone	vera-ellen	vertebroplasty
vibhishana	vibhuti	vidisha	vigone
viloxazine	viridian	vita-mix	vma-144
vmfa-134	vmr-1	vt-8	vytenis
wally	waterloo	wattle-eye	wbdt
wbjb-fm	wbki-tv	wbvp	wcbn-fm
wcpx	wedgwood	wehrmacht	welfreighter
wenedyk	wespe	weston-on-trent	wfly
wfmd	wgtz	wheedle	wikipedia:layout
willenhall	winston-salem	witf-tv	wjbe
wkzv	wmtw	woay-tv	wqbk-fm
wqus	wrdv	wrentit	wrin
ws-reliablemessaging	wsou	wtam	wwjt-lp
wxct	wxlf	wxrt-fm	xenocrates
xianning	xo-1b	yahir	yahui

yano	yawn	yersinia	yingkou
yog-sothoth	yumi	z-dna	zaccanopoli
zanina	zeolite	zerophilia	zlarin