Supervised Machine Learning for Email Thread Summarization

by

Jan Ulrich

B.S. Computer Science, University of Texas at Austin, 2006

B.A. Economics, University of Texas at Austin, 2006

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

THE FACULTY OF GRADUATE STUDIES
(Computer Science)

The University Of British Columbia
(Vancouver)

September 2008

Abstract

Email has become a part of most people's lives, and the ever increasing amount of messages people receive can lead to email overload. We attempt to mitigate this problem using email thread summarization. Summaries can be used for things other than just replacing an incoming email message. They can be used in the business world as a form of corporate memory, or to allow a new team member an easy way to catch up on an ongoing conversation. Email threads are of particular interest to summarization because they contain much structural redundancy due to their conversational nature.

Our email thread summarization approach uses machine learning to pick which sentences from the email thread to use in the summary. A machine learning summarizer must be trained using previously labeled data, i.e. manually created summaries. After being trained our summarization algorithm can generate summaries that on average contain over 70% of the same sentences as human annotators. We show that labeling some key features such as speech acts, meta sentences, and subjectivity can improve performance to over 80% weighted recall.

To create such email summarization software, an email dataset is needed for training and evaluation. Since email communication is a private matter, it is hard to get access to real emails for research. Furthermore these emails must be annotated with human generated summaries as well. As these annotated datasets are rare, we have created one and made it publicly available. The BC3 corpus contains annotations for 40 email threads which include extractive summaries, abstractive summaries with links, and labeled speech acts, meta sentences, and subjective sentences.

While previous research has shown that machine learning algorithms are a promising approach to email summarization, there has not been a study on the impact of the choice of algorithm. We explore new techniques in email thread summarization using several different kinds of regression, and the results show that the choice of classifier is very critical. We also present a novel feature set for email summarization and do analysis on two email corpora: the BC3 corpus and the Enron corpus.

Table of Contents

Abstract . . . . **i**

List of Tables . . . . **vii**

List of Figures . . . . **ix**

Acknowledgments . . . . **xi**

1 Introduction **1**
1.1 Uses for Email Summarization 1
1.2 Supervised Machine Learning Email Thread Summarization . . . 2
1.3 Outline of the Thesis 3
1.4 Contributions 3

2 Related Work **5**
2.1 Summarization 5
2.1.1 Extractive vs. Abstractive Summarization 5
2.1.2 Informative vs. Indicative Summarization 6
2.1.3 Single-Document vs Multi-Document Summarization . . 7
2.2 Machine Learning 8
2.2.1 Supervised vs Unsupervised 8
2.2.2 Classification 8
2.2.3 Meta Classification Techniques 9
2.3 Email Summarization 10
2.3.1 Unsupervised Extractive Email Summarization 10
2.3.2 Supervised Extractive Email Summarization 11
2.3.3 Features to Describe Email Sentences 12
2.4 Datasets / Corpora 12
2.4.1 Public Availability of Corpora 13
2.4.2 Annotation 13
2.5 Evaluation Metrics 14

3 The British Columbia Conversation Corpus (BC3) 17
3.1 The Need for an Annotated Corpus 17
3.2 Available Email Corpora 18
3.2.1 The Enron Corpus 18
3.2.2 TREC Enterprise Data 18
3.2.3 PW Calo Corpus 19
3.3 The Chosen Corpus: W3C 19
3.4 Previous Email Thread Annotations 20
3.5 The BC3 Annotation 20
3.5.1 Speech Acts 21
3.5.2 Meta Sentences 22
3.5.3 Subjectivity 22
3.6 The Annotation Framework 23
3.7 The Annotation Study 24
3.8 Corpus Statistics 27
3.9 Availability 28
3.10 Summary 29

4 Supervised Machine Learning Summarization 31
4.1 Regression-based Summarization 32
4.1.1 Normalization by Sentence Length 33
4.1.2 Classifiers 33
4.1.3 Two Summarization Techniques for Comparison 33
4.2 Sentence Features 34
4.2.1 Analysis of the Feature Set Proposed in Previous Work . . 36
4.2.2 Speech Acts as Additional Features 36
4.3 Experimental Setup 37
4.3.1 Evaluation Metric: Weighted Recall 37
4.4 Results 38
4.4.1 Statistical Analysis 38
4.4.2 Comparing Different Classifiers 39
4.4.3 Continuous vs Binary Labels 41
4.4.4 Normalization by Sentence Length 44
4.4.5 Feature Sets 44
4.4.6 Classifier Runtime Analysis 46
4.5 Summary 49

5 Conclusions and Future Work 51
5.1 Conclusions 51
5.2 Future Work 53

Bibliography **55**

A Ethical Conduct for Research Involving Humans Certification . . . **61**

B Annotator Instructions **63**

List of Tables

3.1 The different annotations for the BC3 corpus. Three annotators were used per email thread. 21
3.2 Various statistics on the BC3 corpus. 28
3.3 Kappa agreement for the different annotations. 28

4.1 Features used for classification arranged in groups. 35
4.2 Significance of T-test comparisons between classifiers for the Enron dataset is shown in bold with the Bonferroni correction. H_A : $Y_{axis} - X_{axis} \neq 0$. P-values are shown for the better classifiers in each pairwise comparison. 40
4.3 Comparing different types of classifiers for summarization using weighted recall for evaluation and the baseline R14 feature set. . . 42
4.4 Comparing feature sets in a continuous and normalized setting. . . 43
4.5 Average weighted recall and significance with Bonferroni correction of T-test comparisons against the R14 feature set for the BC3 dataset. 47

List of Figures

3.1 Imported emails automatically get parsed into fields and their sentences get segmented. 23
3.2 Summarizing the email thread with both abstractive and extractive summaries. 25
3.3 Annotating the email thread for speech act and meta features. . . . 26

4.1 Summarization comparison on the Enron dataset. R14 are the original features presented in [42]. 39
4.2 Comparison of different classifiers in the BC3 corpus. R14 are the original features presented in [42]. 40
4.3 Performance differences between binary and continuous classifiers in the Enron corpus. 44
4.4 Performance differences between binary and continuous classifiers in the BC3 corpus. 45
4.5 A comparison of summarizer performance using annotators scores normalized by sentence length. R14 are the original features presented in [42]. 46
4.6 Comparison of different features on the BC3 corpus. R14 are the original features presented in [42]. SA_A are the annotated labels, SA_S are the sentence level Ciranda [11] labels and SA_E or the email level Ciranda labels. Meta (M) and Subj (S) are manually annotated labels. 47
4.7 Comparison of different features on the Enron corpus. R14 are the original features presented in [42]. SA_S are automatically generated sentence level labels and SA_E are automatically generated email level Ciranda labels. CWS is the clue word score [7]. 48
4.8 A runtime analysis of different summarization algorithms. 49

Acknowledgments

I would like to thank my parents, Fritz and Rita, for supporting me throughout my education, helping me all the way through my masters, and giving me the right advice. I would also like to thank my fiance Kirsten Greene for standing by me through my endeavor through graduate school. She can bring a smile to my face like no one else can. My entire family has helped me along the way.

I would like to thank Giuseppe Carenini and Raymond Ng for being supportive supervisors throughout my research career at UBC. They have given me the freedom to follow my ideas while providing guidance to keep me on track. Our entire research group has been instrumental throughout my work.

Gabriel Murray has been a great person to work with while doing this research. He administrated the annotation process with me and was key in selecting the W3C corpus subset that we used. His experience working with the annotated AMI corpus [10] was very useful for creating the BC3 corpus. With his help the entire annotation process went smoothly. I would like to thank him for editing a draft of this thesis.

This thesis work is definitely a group effort and I could not have done it alone.

This research was funded under a BC Industrial Innovation Scholarship in Intelligent Systems in cooperation with the BC Innovation Council and Business Objects, an SAP company.

Chapter 1

Introduction

Getting information off the Internet is like taking a drink from a fire hydrant. — Mitch Kapor

Email has become a part of most people's everyday lives. It is a convenient form of communication due to its speed and lack of cost. However with these two benefits also comes a cost. Many people receive a large enough number of emails every day that it takes up a significant amount of their time. Summarization is a promising way to reduce the information overload. Summarization can be used to make various parts of using email easier. The problem of email overload is only going to keep increasing. For example, if you have a car with OnStar, it can already email you if it needs an oil change. As more electronic devices gain access to the internet, they will also have the ability to send their owner emails. We therefore want to create tools to effectively deal with this abundance of emails.

1.1 Uses for Email Summarization

Email summarization has many more uses than just summarizing incoming emails. For example important sentences of a new email can be highlighted. In this way a user can still read the entire email, but reading time will be greatly reduced since a user can pay more attention to the highlighted sentences. In the business world, email summarization can be used as a form of corporate memory, where the thread summaries represent all the previous business decisions that have been made. As another possibility, it also allows a new team member to easily catch up on an ongoing conversation.

An interesting approach would also be to use the summarization before an email is actually sent. It can be used along the lines of a spell or grammar checker to improve the quality of a written email. It would be like a content checker that could show which sentences it deems as not important. The user would then have a chance to revise the email into a more concise form.

Summaries can also be specialized for specific purposes. For example, a search result can be enhanced with appropriate summary sentences. Given a search term, the appropriate email results can be displayed. Along with these results the appropriate summary sentences that are also pertinent to the query term can be displayed with each search result. This will allow the user to find the correct email even if the subject line is not very revealing or the query term is not entirely accurate or specific.

It can be seen that there are many different uses for email summarization that are not always straight forward. A particular user's needs determine the appropriate way to use email summarization.

1.2 Supervised Machine Learning Email Thread Summarization

In this thesis we present a method of summarizing email threads. Email threads are of particular interest to summarization because they contain a great deal of structural redundancy due to their conversational nature. Because there are many emails, the amount of text in the thread also tends to be long which leads itself to summarization.

Given an email thread our algorithm creates an extractive summary, which means a chosen subset of sentences from the original email thread. Each sentence receives an importance score and only the most important are selected. We use a trained machine learning algorithm to assign this importance score. In order for this to work, we need a dataset of email threads where the important summary sentences have already been labeled. Since there was no previously publicly available dataset to do this, we labeled part of the W3C email threads for important sentences.

In this thesis we describe the process of annotating a new corpus, or dataset, for email summarization. We then create a new summarization framework that takes advantage of continuous training data labels. We test this summarization framework with different feature sets and classifiers to find the best summarization algorithm.

1.3 Outline of the Thesis

The thesis is divided into five chapters. The first chapter, this one, introduces the nature of the work and provides the motivation. The second chapter provides the background in the field of summarization and introduces the work upon which this is built. In Chapter 3 we describe the new BC3 corpus and the effort and method of building it. In Chapter 4 we introduce and test our summarization framework. Finally in the last chapter we conclude by recapping the achievements of our work. The bibliography and appendix are at the end of the thesis.

1.4 Contributions

As a preview we will outline the specific contributions of this thesis:

- Email corpora are rarely released to the public due to privacy concerns. This is a critical problem that makes email research especially difficult. Furthermore in the field of email summarization, there were no publicly available annotated corpora at all. For this reason we have taken one of the few available email datasets and provided annotations for summarization and other activities. We annotated the email threads with both abstractive and extractive summaries. We also annotated additional sentence features including speech acts, meta sentences, and subjectivity. For this annotation, a web interface and database was also created. Both the corpus and the annotation software have been made publicly available.

- Based on human judgments, it is not clear that there is a single best summary for a given email. Yet previous machine learning summarization methods have used binary labels in their training data. That means that sentences were either included or not, black or white. In our method the training label for each sentence is the average score from several annotators. We then do regression on the training data. In this way the importance score assigned in the summarization process is also continuous and allows for variable length summaries. The highest scoring sentences are added to the candidate summary until the summary length is met. The scores are normalized by length to create the best summary for a given word limit. The final step also eliminates redundant sentences from the summary.

- There has not been a comparison of different classification algorithms for machine learning-based email summarization. We have therefore created a framework where we can test various different classifiers. We use regression algorithms since we are training on a continuous class. To justify the switch to a continuous framework, we compare classifiers to regression algorithms. Then we also compare various regression algorithms to themselves. We find that bagging with regression trees and gaussian processes works the best, but that bagging can be computed much faster.

- We have expanded the feature set beyond a base class of features that have previously been used in email summarization [42]. We introduce other features that email researchers have tried to label before, but that have not been used for email summarization. For example we used speech acts as labels for email summarization. Since emails are conversations, certain speech acts usually contain important information (e.g., request or commit). We used both automatic speech act annotation tools as well as manually annotated speech acts as features in our summarization algorithm. By drawing from work in related fields, we included meta sentences which have been shown to be very useful in meeting summarization [38]. We have also included subjectivity as it improved summarization quality in another email summarization approach [8]. We found that speech acts, meta sentences, and subjectivity are all very useful features, but that the automatically annotated speech acts did not improve weighted recall significantly.

Chapter 2

Related Work

2.1 Summarization

The goal of text summarization is to provide the critical meaning of the text in a concise form. Given the amount of textual information available, automatic summarization would significantly speed up information uptake by the reader. It has been the goal of many researchers to create such automatic summarization tools.

One of the first attempts to create such summaries dates back a literature abstract creator by Luhn [32]. The program analyzed the frequency of the words in the literature to determine which sentences were the most important. This program was able to summarize *New York Times* articles by picking sentences with high occurrences of significant terms. This work was the beginning of the field of text summarization, which has now branched out into many more specific applications. The different approaches can be categorized by several distinct dimensions which are all described in detail in the following sections.

2.1.1 Extractive vs. Abstractive Summarization

There are two types of summarization: extractive and abstractive. The program created by Luhn [32] was an example of an extractive summarizer. An extractive summarizer segments the document into sentences and then chooses the most important sentences for the summary typically by examining the extracted sentence features. In this way a summary can be created without having to generate new sentences and therefore without the necessity of a syntactic level understanding.

Edmundson [20] further elaborated on the extraction work done by Luhn by including pragmatic words, title words, and structural indicators. These new components were proven to work better than just using high-frequency content words as done in previous extraction work.

An abstractive summary is a rewritten form of the original text. It expresses the same ideas in a shorter form. One way to do this is to provide a template for what kinds of information is wanted in the summary. For example a summary template for an election might be, in what country the election was held, what position was being voted for, who or which party won and by how much, and the date for the election. The system can then collect these pieces of information and create an abstractive summary. This method, proposed by DeJong [16] in 1982, was the first attempt at abstractive summarization to the best of our knowledge.

Another approach to abstractive summarization is to try and syntactically parse the text and then prune off unimportant fragments in the syntax tree. This approach was originally proposed by Knight and Marcu [28] to compress sentences in newspaper articles.

Abstraction tends to be a much more complicated task than extraction as it typically requires deeper understanding of the sentence's semantics. Extractive approaches have been more common as they rely on information that can be more easily extracted from text (e.g. word counts, position of the sentence in the document). Abstractive approaches that rely on templates are also domain dependent while this is not true for extractive approaches. Furthermore it has been shown that at least in the domain of news, extractive summaries are more effective than abstractive summaries [36]. In this thesis we focus on extractive summarization by choosing the most informative sentences using extracted sentence features and annotated training data.

2.1.2 Informative vs. Indicative Summarization

The intended use of a summary is important for its creation. An informative summary contains all the salient information in the original document and omits all the unimportant parts. The informative summary is meant to replace reading the original document. An indicative summary on the other hand has a different purpose. It is meant to help users to find out whether the original document might be useful to them. Therefore it introduces the topics and theme of the document, but it does not give away the actual details.

The tradeoff between informative and indicative summaries is explored in the work by Kan et al. [26], where both indicative and informative summaries are created for documents in the medical domain. They decided to present both of the summaries since informative summaries are good for users browsing a particular topic and trying to learn more about it and indicative summaries are good for users who are trying to find a document that answers a particular question.

We have chosen to focus on informative summarization as we want to give a user a quick overview of an email conversation. This can be good for someone who is trying to get up to speed with their co-workers' discussion. It can also be used to highlight sentences, and therefore let the user understand the information in the email by just reading the highlights.

2.1.3 Single-Document vs Multi-Document Summarization

Summaries can be over just a single document or a collection of documents. The vast numbers of online documents make multi-document summarization ever more important. The multi-document summaries take the relationships between all the documents into account. In an early attempt at multi-document summarization [35], many news articles were summarized together. In multi-document summarization the issues that have to be solved include redundancy reduction, identification of the differences in the documents, and coherence due to possible topic shifts.

Redundancy reduction has been performed using MMR (Maximal Marginal Relevance) [5]. In this technique a candidate sentence is compared to the sentences already chosen and if it is too similar it is discarded. Finding the differences in documents and maintaining coherence are still open research problems.

MEAD is a summarization framework that can handle both single-document and multi-document summarization [41]. The framework performs extractive summarization on a collection of documents. It consists of three different components: a feature extractor, a classifier, and a reranker. MEAD first segments the original text into sentences. Then the feature extractor converts each sentence into numerical values, called features. These include sentence length, position of the sentence in the document, and the occurrence of different words in term frequency vectors. In the second step, these features are fed into a classifier which assigns an importance score for each sentence. Finally the reranker adjusts these scores to reduce redundancy in the final summary. The highest scoring sentences are then selected for the summary. The MEAD framework gives researchers the ability to change any of the three components to try out different summarization techniques. We have used it structure our summarization software.

In our approach we conduct multi-document summarization since we choose to summarize an entire email thread where each email is considered a single document. Most single emails can be read entirely without too much trouble. However summarization is very useful when having to read an entire conversation's worth of email. By summarizing an email thread, we can take advantage of the information contained in the conversation structure.

2.2 Machine Learning

Machine learning is the process of using observations to build predictions and a course of action for the future. The observations from the real world are often noisy, and deterministic models do not take the possibility of an error in observation into account. Therefore the field of statistical machine learning was introduced to create a model of the world using probabilistic inference [19].

2.2.1 Supervised vs Unsupervised

Machine learning can either be supervised or unsupervised, depending on whether labeled training data or unlabeled training data is supplied. One form of unsupervised machine learning is clustering, where the data is divided into a number of clusters. A classical method for doing such clustering is the expectation-maximization (EM) algorithm [17]. The algorithm first estimates the possible positions of the clusters. Then it calculates the likelihood of these positions given the data. The process iterates until it converges.

Supervised approaches have labeled training data. This means that annotation is required before the algorithm can be trained. Supervised approaches are usually used to do classification or regression. One example of supervised learning is handwriting recognition [46]. Here handwritten letters are scanned and labeled with the correct letters. Then a classification algorithm is trained on this data so that newly written letters can also be recognized.

2.2.2 Classification

One of the questions asked in probabilistic inference is, ”Given a new observation described by a set of features, how does it fit into the model of the environment that has already been built?” It is therefore a classification task among the different possibilities in the model. Classification is therefore closely related to learning.

Naive Bayes is one of the oldest and simplest classification techniques. Its application to machine learning was first explored by Duda et al. [19]. It is naive because it assumes independence of the different features given the class label. However it is simple to build and has had remarkable success even if the independence assumption is not met [18].

More advanced classification techniques include neural nets [1] and support vector machines [15]. Neural nets are inspired by the human brain which uses billions of neurons that are interconnected to make decisions. A neural net consists of several layers of artificial neurons that are interconnected at each level. If an artificial neuron receives a signal stronger than its threshold, it fires and passes its signal to all the artificial neurons it is connected with. The strength of each of these connections can be changed by using weights. The algorithm adjusts the weights of the connections as it learns.

Support vector machines use the idea of kernels to separate the data into a positive side and a negative side for classification [15]. The original feature space is transformed into a higher order dimensional space using kernels such that a single hyperplane can separate the positive data from the negative data. This hyperplane is held in place by the ”support vectors” which are the data points closest to the hyperplane. The placement of the hyperplane is found by maximizing the distance between the hyperplane and the closest positive and negative point.

Weka is is a collection of machine learning software for data mining tasks [48]. It provides implementations of many different machine learning algorithms. This is a great tool for trying out different classifiers without having to code all of them. It is written in Java and open sourced so that researchers can add their own implementations as well. We use Weka for the implementation of our classification and regression algorithms. In this thesis we will compare the performance of different classifiers that can be used to extract summary sentences.

2.2.3 Meta Classification Techniques

Meta classification techniques can boost the accuracy of classifiers by increased computation. The classification is repeated several times with a change in the training set. Then a final result is produced from all the different classifiers. Two of the most popular meta classification methods are bagging and boosting.

Bagging stands for bootstrap aggregating which improves the accuracy of unstable classification algorithms [4]. Unstable classification algorithms create estimators with high variance which means small changes in the features create different classification labels. It does this by bootstrapping the training set to produce many different training sets from a subset of the original training data. This means that each resulting training set contains several copies of different values in the original training set. Many different classifiers are then trained on the bootstrapped training sets and a majority vote is taken at the end to determine the final classification. The classification repetition using slightly different training sets makes the results of unstable classifiers much more robust.

Boosting is also a meta classification approach which improves the accuracy of classification algorithms by increasing their confidence margins [22]. In boosting each data point in the training set is assigned a weight which determines the number of duplicates present. In the beginning this weight is 1 for all data points. The classifier is then trained iteratively with the weights adjusted at the end of each iteration. The misclassifications or data points with low confidence at each step are the ones that get their weights increased for the next step. In this way the algorithm is putting more emphasis on the data points which were misclassified in the last round. The final classification is the result of a vote of the classifiers built at each iteration step.

It might seem strange that boosting keeps improving its accuracy with more iterations even though no new data is added to the classifier. The reason for this is that with each iteration the algorithm becomes more confident in its vote, or margin, for classifying each data point because a new vote is added in each iteration that focuses on the data points with the lowest margin [43].

Extractive summarization lends itself well to machine learning. The text can be separated into sentences and then the problem becomes a classification task on sentences. Machine learning algorithms can be trained to select sentences for extraction given features for each sentence. In this way machine learning can be used for summarization.

2.3 Email Summarization

Email is an especially useful case for summarization since most people are bombarded with many emails each day. By using sentence extraction in different ways, the process of managing one's email folders can be eased. For example sentence highlighting allows users to skim an email by reading the most important sentences. Alternatively a generated summary can make it easier to access email on the small screen of a mobile device.

Previous efforts in email thread summarization have adopted techniques developed for general multi-document text summarization and applied them to email summarization by including email specific elements. Approaches have included unsupervised and supervised machine learning.

2.3.1 Unsupervised Extractive Email Summarization

Unsupervised techniques can automatically summarize emails without the need of any prior training. This is useful as it does not require a large annotated dataset to train from. The Clue Word Summarizer [7] is one such unsupervised approach which takes advantage of the email thread structure when creating summaries. The email thread structure is analyzed and a fragment quotation graph is created by dividing emails into smaller fragments if only part of the email is quoted in the conversation. The sentence importance is based on clue words which are words that appear in semantically similar form in the parent or child nodes of the current fragment in the fragment quotation graph.

Wan and McKeown [47] worked on a more specific problem which was finding the topic of an email conversation. The topic was then summarized using the most descriptive sentence for the topic. The approach used term frequency vectors and SVD to find the different and most important topics. Heuristics were finally used to fine tune the system.

The advantage of unsupervised approaches is that they do not need a training dataset. However because they do not have this additional information, many supervised approaches can outperform unsupervised summarization methods.

2.3.2 Supervised Extractive Email Summarization

Supervised machine learning has previously been used successfully to summarize email threads [42]. This work focused on feature selection for the purpose of email summarization. Traditional features used in text summarization (e.g. document position, length, and centroid similarity) were compared to conversation and email specific features (e.g. subject similarity, number of recipients, and number of responses). It was found that email and conversation specific features improved recall and precision significantly compared to general text features. For the sentence extraction, Ripper [12], a binary rule-based classifier, was the only algorithm that was used.

Creating good summaries involves more than just picking the correct sentences from the original text. Sentence order and coherence is also important. For example Shrestha and McKeown [45] detect question-answer pairs in email conversations using trained classifiers. Standard text features (e.g. length, cosine similarity) and email thread features (e.g. number of intermediate messages, ratio of number of messages sent earlier in the thread). Ripper was then used to identify question-answer pairs. McKeown et al. [37] use this to augment the Ripper based extractive sentence summarizer with additional sentence to improve coherence. If an answer is included in the summary then the complementary question is added as well, and vice-versa.

Another summarization technique by Zajic et al. focused on sentence compression [49]. They apply syntactic compression rules to a parse tree of the email text. Given an email thread, they try a multi-document summarization approach as well as a single document summarization approach, finding that the email thread structure is important and therefore the multi-document approach fares better.

Corston-Oliver et al. [14] studied how to identify ”action” sentences in email messages and then rewriting them as action-item summaries. They used an SVM with a linear kernel to identify sentences containing a task. The sentences were then rewritten into a task oriented imperative and populated into the Outlook ”to-do” list. They performed sentence extraction, but by then rewriting the sentences, their approach moves towards abstractive summarization.

Most summarization approaches still focus on extraction. There have been some abstractive approaches, but even those have not been true abstraction. True abstraction means recognizing sentences which constitute something new and replacing them with the new text which is not found in the original document [40]. Due to the maturity of the field, we therefore use supervised extractive summarization for our summarization system as outlined in Chapter 4.

2.3.3 Features to Describe Email Sentences

When using statistical machine learning, in order to effectively describe email sentences, many information rich features are needed. Some research has therefore focused on a subset of the overall summarization goal, which is creating sentence specific features.

Carvalho and Cohen [11] focus on automatically labeling emails as different speech acts. They use machine learning to classify emails based on unweighted bag of words and bigrams as features. For these speech act labels to be useful in extractive email summarization, a higher granularity of labels at the sentence level is needed.

Rambow et al. [42] use standard text base features such as sentence position, length, centroid similarity, and tfidf sum. Then they added conversation specific features such the position of the email in the email thread. Finally they added email specific features such as subject similarity, the number of responses, and the number of recipients. In their work they compare these different classes of features and their results show that email specific features did significantly improve summarization performance.

Carenini et al. [7] apply the clue words feature as the main tool for summarization. As was described in Section 2.3.1 clue words are words that appear in semantically similar form in the parent or child nodes of the conversation graph. These words are used as a feature to represent the importance of each sentence.

In this thesis we combine and compare all these different feature sets. As a baseline we will use all the features used by [42]. Then we combine features from Carvalho and Cohen [11] and Carenini et al. [7].

2.4 Datasets / Corpora

Datasets of actual email conversations are needed for training (in supervised approaches) and testing. To be useful for research an email summarization the email conversation need to be annotated, in the sense that important sentences have been labeled by human annotators. The more data is available the better the algorithms can perform and the results of the evaluations become more accurate. The following datasets have been used in related work. Datasets can also be called corpora.

2.4.1 Public Availability of Corpora

It is often very difficult to get access to real email conversations as the use of personal emails is a privacy concern. Even if the emails are anonymized, the content of the email conversations can still be revealing. For these reasons there are not many publicly available corpora for email research. Often when researchers get access to emails for research it is under the condition that they do not make the actual emails publicly available such as in the Columbia corpua. However there are a few corpora that are publicly available such as the Enron Corpus.

The Enron Corpus

The Enron dataset was released after the legal investigation into the Enron corporation [27]. This is an invaluable dataset since it contains uncensored messages from a corporate environment. The dataset contains over 30,000 threads, but they end up being rather short with an average thread size of just over four and a median of two.

The Columbia Corpus

The Columbia corpus contains 96 email threads from the members of the board of the ACM student organization at Columbia University. Most of the emails regarded planning events for the student association. Each thread contains an average of 3.25 email [42]. This corpus was not publicly released.

2.4.2 Annotation

In order for email corpora to be used for training and testing, they have to be annotated for email summarization. Usually this means that important sentences need to be marked by humans in order to evaluate the generated summaries. More details on this will be presented in Section 2.5.

Previous work on corpora annotation has been done in the meeting domain. The AMI corpus was created by simulated business meetings that were later annotated [9]. Multiple annotators labeled gestures, name entities, topic segmentation, and summarization. These annotations can be used for training machine learning techniques and to better understand the dataset. The annotations were made publicly available as stand-off XML.

The Enron Corpus

Part of the Enron corpus was annotated for summarization at the University of British Columbia. The annotations have not been released.

From the Enron corpus, 39 email threads were selected from the 10 largest email inbox folders such that each thread contained at least four emails. These email conversations were annotated using manual sentence extraction. The 50 annotators that were recruited for the study were undergraduates and graduate students.

Each annotator had to summarize four email threads by selecting which sentences to include. Each thread was therefore annotated by five annotators. The annotators were asked to pick 30% of the original sentences such that the summary contained the overall information in the email and could be understood without referencing the original email thread. Sentences were labeled as either *essential* or *optional*, where an *essential* sentence is vital to the understanding of the summary and an *optional* sentence elaborates on the meaning of the conversation but is only included if there is space in the summary.

The Columbia Corpus

Two annotators were told to each write summaries for each thread in the Columbia corpus. The summaries were supposed to be 5%-20% of the original length with a maximum of 100 lines and the annotators were instructed to write in the past tense, use speech-act verbs, and embedded clauses [42]. These hand written summaries were then used to find the important sentences in the original documents. SimFinder [23] was used to find the similarity between each sentence in the summary and each sentence in the corresponding thread. The highest similarity score for each sentence in the email thread is used. A threshold is set and any of the sentences with a score above the threshold are included and the others are not. The threshold is set so that about 26% of the sentences are included.

2.5 Evaluation Metrics

Evaluating summaries is tricky as there is no single best summary that human judges can agree on. Because it is time-consuming and expensive to have human judges rate summary output or to do extrinsic tasks, the development of automatic metrics that predict these external measures is crucial. Different methods have been proposed that try to deal with these issues.

One evaluation method is ROUGE [30], which was created to automatically be able to compare summaries. Therefore human generated gold standards are needed for comparison with the automatically generated summaries. ROUGE is a collection of several different n-gram comparisons. The bigram based comparison has been shown to correspond well with human evaluations of newswire data summaries and is being used in the Document Understanding Conference [30]. Because there is no single best summary, multiple reference summaries are usually used as a gold standard.

The ROUGE evaluation metric was created for newswire data. It has not been shown to correlate well with human evaluations in the meeting domain [31], which is similar to emails in informality and conversational nature. We have therefore chosen to use another evaluation metric.

The pyramid method [39] is a different evaluation approach which relies on the fact that there is no single best summary. A set of gold standard summaries written by human annotators is needed. These summaries, also called model summaries, are then analyzed and divided into summary content units (SCU) which are not larger than a clause and are repeated throughout the summaries. The SCUs are then arranged in a pyramid where each level represents the number of times the SCU occurred in the model summaries. For a given length, the optimal summary then contains the SCUs from the highest tiers possible. A given summary is scored according to the tiers of the SCUs the summary contains. The pyramid score is the ratio of the given summary divided by the score of the optimal summary.

The fact that a single best summary does not exist is important for evaluation. Since we are evaluating extractive summaries we haven chosen to use a simplified version of the pyramid method, called weighted recall, which can use sentence indexes instead of SCUs.

Chapter 3

The British Columbia Conversation Corpus (BC3)

Corpora, or datasets, are fundamental for evaluation, as well as training data in statistical machine learning. Research in email summarization has suffered from a lack of publicly available annotated email corpora because releasing email datasets has many privacy concerns. In this chapter we describe our attempt to tackle this problem. We developed a new email corpus that was annotated for summarization and publicly released. We call it the British Columbia Conversation Corpus (BC3) because it was annotated at the University of British Columbia, and although we started by annotating email threads, we plan on expanding it to other conversations, such as online chats and blog discussions. An annotation framework was also developed in the creation of the corpus to allow annotators to easily summarize and label the email threads.

3.1 The Need for an Annotated Corpus

While researchers in email summarization ideally want to work with realistic, large, and richly annotated data, privacy concerns mean that large collections of real email data cannot typically be shared. This has left the field with each researcher training and testing her new approach on her own dataset making comparisons across different approaches extremely difficult. Also with this lack of large public datasets comes the risk of overfitting algorithms to the limited available data. We believe that a public email corpus, complete with summarization annotations for training and evaluation purposes, would greatly benefit the research community.

3.2 Available Email Corpora

Below we describe several relevant datasets that are publicly available, and assess their suitability for further annotation and research.

3.2.1 The Enron Corpus

The most popular email corpus for research has been the Enron dataset [27], which was released during the legal investigation into the Enron corporation. This is an invaluable dataset since it contains uncensored messages from a corporate environment. Because this corpus consists of real, unaltered data, there were initially integrity problems that needed to be fixed. The dataset contains over 30,000 threads, but they end up being rather short with an average thread size of just over four and a median of two emails. The dataset consists of employee's email folders, so it is also an accurate depiction of how users use folders. The Enron corpus is the most widely used email corpus in research. Researchers have used it for a variety of purposes: to create automatic email foldering [3], visual data mining [24], and organizational social network discovery using emails [44].

3.2.2 TREC Enterprise Data

In the TREC Enterprise Track[1], the task is to search through an organization's multimodal data in order to satisfy a given information need. Organizations often possess a great variety of data, from emails to web pages to spreadsheets and word documents. Being able to search, summarize and link these documents can greatly aid corporate memory. In recent years, TREC Enterprise has used two multimodal datasets for Question-Answer (QA) tasks: the W3C corpus and the CSIRO corpus.

The W3C Corpus

The W3C corpus is data derived from a crawl of the World Wide Web Consortium's sites at w3c.org. The data includes mailing lists, public webpages, and text derived from .pdf, .doc and .ppt files, among other types. The mailing list subset is comprised of nearly 200,000 documents, and TREC participants have provided thread structure based on reply-to relations and subject overlap. There are more than 50,000 email threads in total. W3C data has been annotated for QA topic relevance for use in TREC Enterprise 2005 and 2006.

[1] http://www.ins.cwi.nl/projects/trec-ent/

The CSIRO Corpus

Subsequent to TREC 2006, TREC Enterprise began using similar data from the Australian organization CSIRO. Because the QA topics and relevance annotations were drawn up by CSIRO employees themselves rather than by outside annotators, this dataset and annotation together represent more realistic information needs. However this data does not include any mailing lists and therefore contains no emails [2].

3.2.3 PW Calo Corpus

As part of the CALO project[2], an email corpus was gathered during a four-day exercise at SRI, in which project participants took part in role-playing exercises and group activities. The resultant PW Calo corpus [13] totals 222 email messages. The dataset also includes correlated meeting recordings, but currently the corpus is unfortunately not freely available.

3.3 The Chosen Corpus: W3C

Our goal has been to release an annotated corpus for email thread summarization. We wanted to do the annotation work, but we needed to choose a corpus to annotate. Out of the possible corpora listed above, The PW Calo corpus was not freely available and the Enron corpus had already been partially annotated [7]. This left the W3C corpus.

One possible limitation of using the W3C corpus for email summarization is that summarizing mailing lists can be considered to be a slightly different task than summarizing emails from a user's inbox. In the latter case there will be missing or hidden data that can sometimes be reconstructed [6], while this is much less of an issue with a mailing list. Mailing lists might also not be as personal as an email going out to only one person.

An advantage of using the W3C corpus is it's size of 50,000 email threads. Even though we are annotating only a small portion of the mailing list threads, having a very large amount of related but unannotated data could prove very beneficial for the development and refinement of summarization systems. Though much of the W3C email data is very technical, we ultimately chose 30 threads that were somewhat less technical than average, so that the annotators would face less difficulty in making their summarization annotations. An additional criterion for selecting these threads was that they are sufficiently long to justify being summarized. The average number of emails per thread in this subset was just under 11,

[2]http://caloproject.sri.com

and the number of participants per thread was just under 6.

Before the data was annotated, some pre-processing was needed. Because the automatic sentence segmenters we applied yielded sub-optimal results on email data, we carried out manual sentence segmentation on the relevant emails. We also formatted the emails so that all quoted text is in the same format, preceded by one or more quote brackets, e.g. $>>$.

3.4 Previous Email Thread Annotations

The Enron corpus has previously been annotated for summarization [7]. A subset of threads were selected, and important sentences were ranked by annotators in a user study. Each thread was summarized (using extraction) by 5 annotators which were asked to select 30% of the original number of sentences and label them as essential or optional. Each sentence was then given an overall importance score based on the combined annotations. An essential sentence carried three times the weight of an optional sentence. Using several annotators is assumed to produce a more meaningful result by diminishing the effect of individual differences. 39 threads were annotated in this way to produce a corpus for evaluation.

This annotation was targeted towards extractive summarization and it is unclear how it could be used with an abstractive summarization technique. It also has several arbitrary restrictions including the requirement of 30% sentence selection as well as only two choices for sentence importance. Annotators had to label sentences as either non-important, optional, or essential. The weight of an essential sentence compared to an optional sentence was also somewhat arbitrary. Annotators might have had a different conception of what is an optional sentence and an essential sentence. However the annotator's directions were straight-forward and the annotation of selecting sentences was easier for the annotator than writing a summary from scratch as we have required in the BC3 annotation.

3.5 The BC3 Annotation

In our annotation, we wanted to perform a more thorough annotation than previously done in other email summarization projects. In BC3 the email threads have been annotated both for abstractive and extractive summarization. Sentences have been labeled with different features that can be used for training machine learning summarizers. The annotation of each thread includes a written abstract summary where each sentence in the summary is linked to an original sentence in the email thread that contains highly similar content. Furthermore for each thread an extractive summary has been annotated by selecting a subset of original sentences. We annotated both kinds of summaries so that this corpus can be useful to both

Annotated Property	Properties
Abstractive summary	250 word limit
Original sentence links	At least one link per abstractive sentence
Extractive summary	Binary label for inclusion
Speech Acts	At the sentence level and not mutually exclusive
Meta Sentences	
Subjective Sentences	

Table 3.1: The different annotations for the BC3 corpus. Three annotators were used per email thread.

researchers performing abstractive summarization and those performing extractive summarization (or a combination of the two).

The links between the abstractive summary and the sentences in the thread require a little more explanation. These links are specified by the annotator between each human written sentence and corresponding extracted sentences containing highly similar information. In exceptional cases we allowed the annotators to provide no links after a summary sentence, but this was a rare occurrence. The links show the transformation of each idea from the original text to the corresponding summary sentence. By having to form these links after the summary was written, it makes the annotator justify both their extractive choices as well as their written summary. These links can be very useful in evaluating extractive summaries generated by a system: The more the sentences selected by the system are linked to sentences in the human written abstract the better. Such an annotation approach has been used in the ICSI [25] and AMI [9] corpora in meeting summarization.

For use in machine learning summarization systems, the corpus is also labeled with additional binary features at the sentence level. These include speech acts, meta sentences, and subjectivity.

3.5.1 Speech Acts

Taking the original work of Carvalho and Cohen [11] as inspiration, we decided to annotate speech acts in the new corpus. In their original work, emails were classified as *Propose*, *Request*, *Commit*, *Deliver*, *Meeting*, and *deliveredData*. These categories are not mutually exclusive. The intuition is that sentences expressing these speech acts tend to be more important ones for summarization. For our annotation we used a subset of the original speech acts that are considered the most informative. The BC3 annotation consists of the following categories: *Propose*, *Request*, *Commit*, and *Meeting*. A *Propose* sentence proposes a joint activity; a

Request asks the recipient to perform an activity; a *Commit* sentence commits the sender to some future course of action; and a *Meeting* sentence is regarding a joint activity in time or space. *Deliver* is excluded because most emails deliver some sort of information and *deliveredData* is excluded because attachments or hyperlinks can be automatically parsed from an email message, so they do not need to be annotated for.

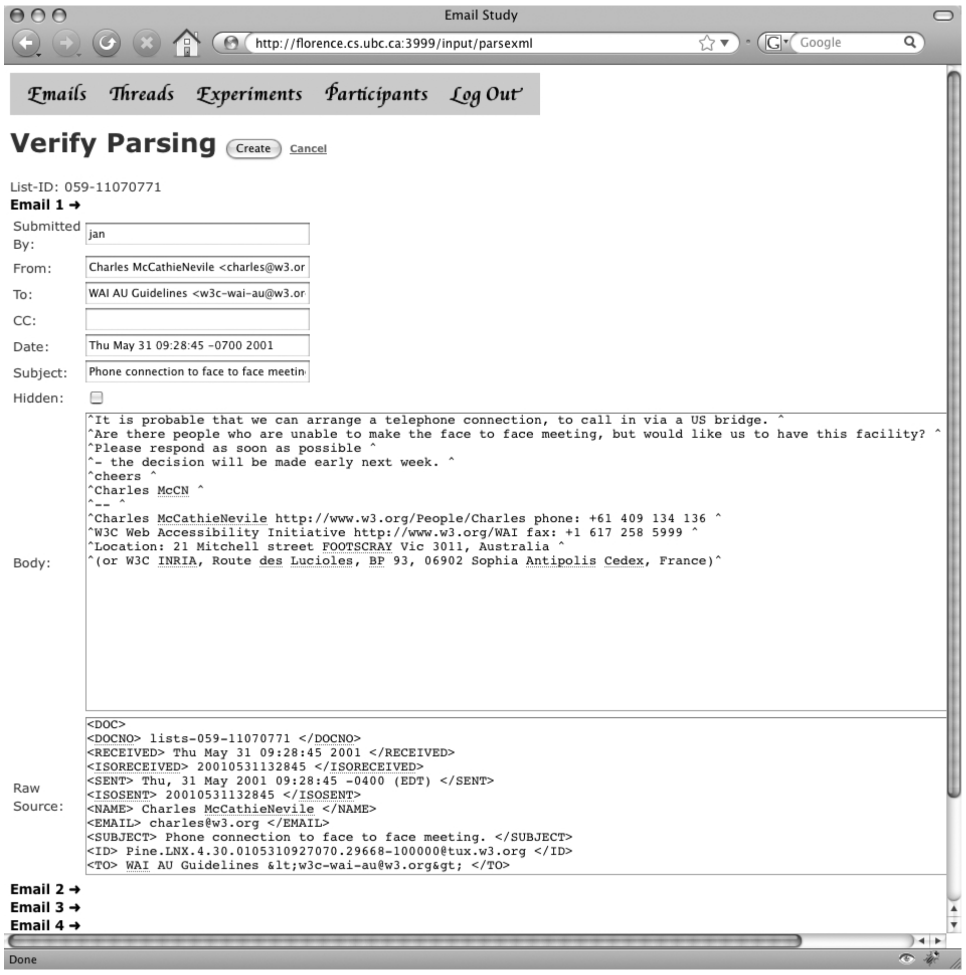
The order of these annotations also becomes interesting in summarization, as can be seen in the work by Carvalho and Cohen [11]. If a request is chosen as an important sentence then the corresponding commit should also be included in the summary. This sequential relationship can be obtained from the email ordering in the threads.

3.5.2 Meta Sentences

In work on automatic summarization of meeting speech, Murray and Renals [38] made use of meta comments within meetings. Meta comments are sentences where a speaker is referring explicitly to the meeting itself or to the discussion itself, e.g. "So we've decided on the user interface." They found that including these comments in meeting summaries improved summary quality according to several metrics. In order to verify whether a similar benefit can be found in email summarization, we have enlisted our annotators to label sentences as meta or non-meta, where a meta sentence refers explicitly to the email discussion itself. These features have proven to help email summarization as can be seen in Section 4.4. We hypothesize that the detection of such sentences will also aid the coherence of our extractive summaries.

3.5.3 Subjectivity

We also chose to label subjective sentences as subjectivity was found useful for email summarization [8]. The label *Subj* means that the writer or speaker is expressing an opinion or strong sentiment. The sentence reveals the so-called "private state" of the writer. In more technical terms, a subjective sentence is when an experiencer holds an attitude towards a target. The attitude can be positive, negative or neutral. A sentence is only labeled as subjective if it reveals the private state of the writer and NOT when the writer is reporting someone elses private state. If a writer feels strongly about something, it can be an indication that the sentence is important.


Email Study
http://florence.cs.ubc.ca:3999/input/parsexml
Google
Emails Threads Experiments Participants Log Out
Create
Cancel
Done

Verify Parsing

List-ID: 059-11070771

Email 1 ➜

Submitted By:	jan
From:	Charles McCathieNevile <charles@w3.or
To:	WAI AU Guidelines <w3c-wai-au@w3.or
CC:	
Date:	Thu May 31 09:28:45 -0700 2001
Subject:	Phone connection to face to face meetin
Hidden:	☐

Body:

```
^It is probable that we can arrange a telephone connection, to call in via a US bridge. ^
^Are there people who are unable to make the face to face meeting, but would like us to have this facility? ^
^Please respond as soon as possible ^
^- the decision will be made early next week. ^
^cheers ^
^Charles McCN ^
^-- ^
^Charles McCathieNevile http://www.w3.org/People/Charles phone: +61 409 134 136 ^
^W3C Web Accessibility Initiative http://www.w3.org/WAI fax: +1 617 258 5999 ^
^Location: 21 Mitchell street FOOTSCRAY Vic 3011, Australia ^
^(or W3C INRIA, Route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex, France)^
```

Raw Source:

```
<DOC>
<DOCNO> lists-059-11070771 </DOCNO>
<RECEIVED> Thu May 31 09:28:45 2001 </RECEIVED>
<ISORECEIVED> 20010531132845 </ISORECEIVED>
<SENT> Thu, 31 May 2001 09:28:45 -0400 (EDT) </SENT>
<ISOSENT> 20010531132845 </ISOSENT>
<NAME> Charles McCathieNevile </NAME>
<EMAIL> charles@w3.org </EMAIL>
<SUBJECT> Phone connection to face to face meeting. </SUBJECT>
<ID> Pine.LNX.4.30.0105310927070.29668-100000@tux.w3.org </ID>
<TO> WAI AU Guidelines &lt;w3c-wai-au@w3.org&gt; </TO>
```

Email 2 ➜
Email 3 ➜
Email 4 ➜

Figure 3.1: Imported emails automatically get parsed into fields and their sentences get segmented.

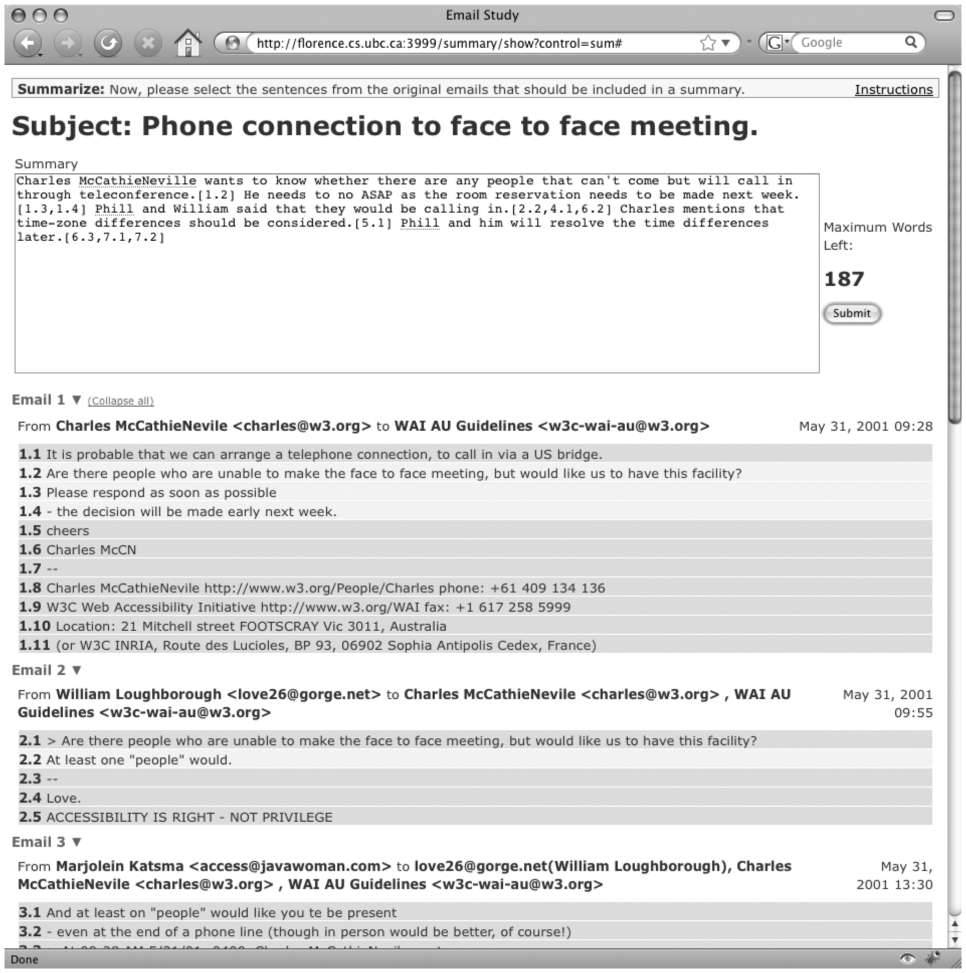
3.6 The Annotation Framework

A web based interface was created for the annotation. This is convenient for the annotator whose job of selecting and labeling sentences is made easier. It is also good for the researcher since the data is automatically compiled and ready to use for research purposes. The interface that was created is a web database application written in Ruby on Rails which processes the original emails and records the annotation results. The researcher can import emails in raw source or XML format and the application then parses the header fields. Figure 3.1 shows how the system parsed an imported xml file. The researcher can verify that the parsing is correct before the data is written to the database. The researcher then combines the emails into the threads that need to be summarized. This database of emails can then be used to run annotation studies. The researcher selects which threads should be summarized by the current annotator and then starts the annotation study. Now the researcher is logged out of the web application for security purposes and the annotator can take over to perform the annotation. The results are stored in an MySQL database which can transform the results to the format needed by the researcher. The researcher can also log in and view past results from previous annotators. The interface therefore allows for complete control over the annotation process. This framework is made publicly available as described in Section 3.9.

3.7 The Annotation Study

We annotated 40 different email threads that we selected from the W3C corpus. Each thread was annotated by three different annotators to reduce overall subjectivity. The annotation study was run on individual computer stations and annotators were supervised by researchers. The study consisted of two steps: a thread summarization step producing both an abstractive and an extractive summary, followed by a sentence feature labeling step.

First the annotators were asked to fill out a profile of their personal information and given instructions on how to annotate. These instruction can be viewed in Appendix B. Before the annotator started with annotating actual threads, a sample thread was annotated for practice. The annotation started with the email thread being displayed allowing each email to be collapsed for navigational ease. The annotator was expected to read the entire email conversation to become familiar with the content. In the first stage annotators were asked to write a 250 word summary describing the information contained in the email thread. This summary length was chosen because it matches the length of the DUC2007 human written model summaries in the main summarization task [34]. The annotation webpage can be seen in Figure 3.2 where the annotator wrote the summary in the text box on top while referencing the email thread below. Then each sentence in the written summary was linked to at least one original sentence with highly similar content. The annotator did this by putting the corresponding sentence numbers behind each written sentence. Annotators were then asked to create an extractive summary by selecting important sentences from the original text. The interface shown in Figure 3.2 allows the annotator to click any of sentences in the original text to indicate that they are important. There were no restrictions on the number of sentences which had to be extracted. Extracted sentences did not necessarily have to be linked nor did linked sentences have to be extracted. After the annotators created the summaries,


Email Study

http://florence.cs.ubc.ca:3999/summary/show?control=sum#

Summarize: Now, please select the sentences from the original emails that should be included in a summary. Instructions

Subject: Phone connection to face to face meeting.

Summary

```
Charles McCathieNeville wants to know whether there are any people that can't come but will call in
through teleconference.[1.2] He needs to no ASAP as the room reservation needs to be made next week.
[1.3,1.4] Phill and William said that they would be calling in.[2.2,4.1,6.2] Charles mentions that
time-zone differences should be considered.[5.1] Phill and him will resolve the time differences
later.[6.3,7.1,7.2]
```

Maximum Words Left: **187**

Email 1 ▼ (Collapse all)

From **Charles McCathieNevile <charles@w3.org>** to **WAI AU Guidelines <w3c-wai-au@w3.org>** May 31, 2001 09:28

1.1 It is probable that we can arrange a telephone connection, to call in via a US bridge.
1.2 Are there people who are unable to make the face to face meeting, but would like us to have this facility?
1.3 Please respond as soon as possible
1.4 - the decision will be made early next week.
1.5 cheers
1.6 Charles McCN
1.7 --
1.8 Charles McCathieNevile http://www.w3.org/People/Charles phone: +61 409 134 136
1.9 W3C Web Accessibility Initiative http://www.w3.org/WAI fax: +1 617 258 5999
1.10 Location: 21 Mitchell street FOOTSCRAY Vic 3011, Australia
1.11 (or W3C INRIA, Route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex, France)

Email 2 ▼

From **William Loughborough <love26@gorge.net>** to **Charles McCathieNevile <charles@w3.org> , WAI AU Guidelines <w3c-wai-au@w3.org>** May 31, 2001 09:55

2.1 > Are there people who are unable to make the face to face meeting, but would like us to have this facility?
2.2 At least one "people" would.
2.3 --
2.4 Love.
2.5 ACCESSIBILITY IS RIGHT - NOT PRIVILEGE

Email 3 ▼

From **Marjolein Katsma <access@javawoman.com>** to **love26@gorge.net(William Loughborough), Charles McCathieNevile <charles@w3.org> , WAI AU Guidelines <w3c-wai-au@w3.org>** May 31, 2001 13:30

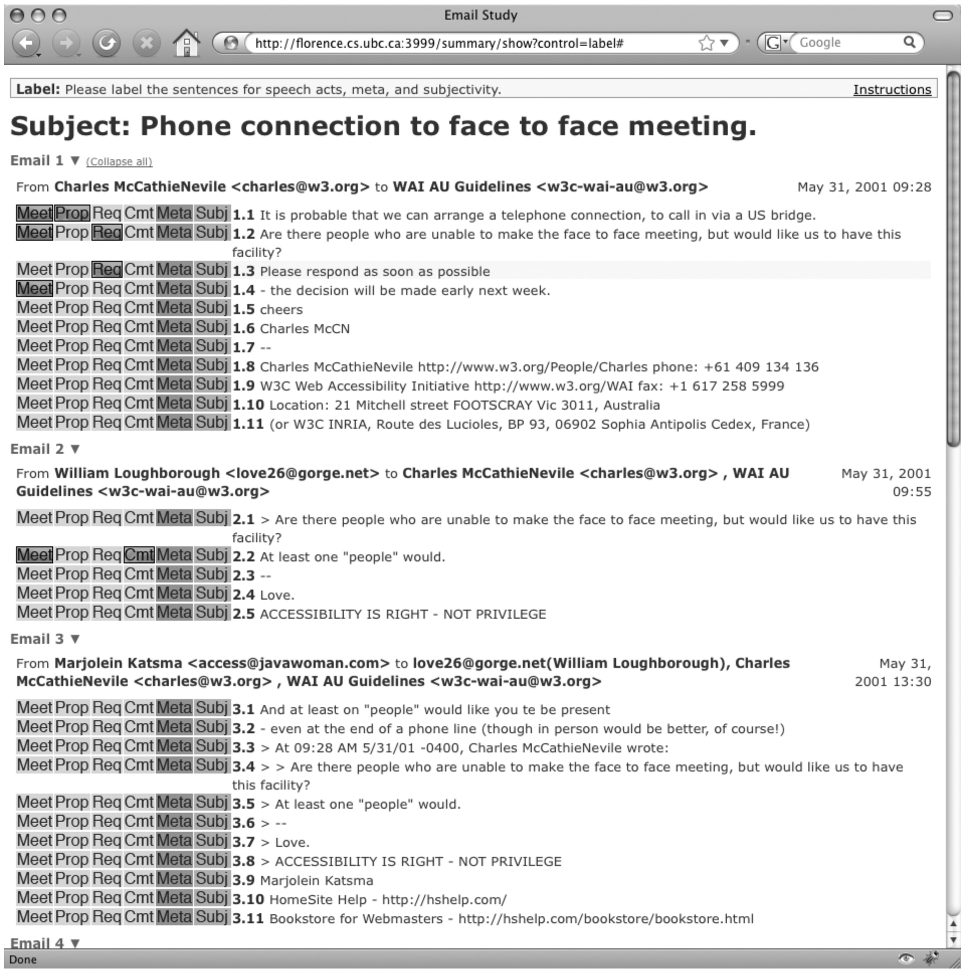
3.1 And at least on "people" would like you te be present
3.2 - even at the end of a phone line (though in person would be better, of course!)

Done

Figure 3.2: Summarizing the email thread with both abstractive and extractive summaries.

they were asked to label additional binary features for each sentence. These were meta sentences, subjective sentences and speech acts which included: *Proposal*, *Request*, *Commit*, and *Meeting*. Figure 3.3 shows how were able to click on the varies labels next to the sentences to label them. By going through all the steps we just described, the user created an abstractive and an extractive summary with links between them as well as additional labeled sentence features.

The study was performed with 10 annotators who each annotated 12 different threads. The annotators were undergraduate and graduate students as well as other people affiliated with the University of British Columbia. Some of their areas of


Email Study
http://florence.cs.ubc.ca:3999/summary/show?control=label#
Google
Instructions
(Collapse all)
Meet Prop Req Cmt Meta Subj
Done

Label: Please label the sentences for speech acts, meta, and subjectivity.

Subject: Phone connection to face to face meeting.

Email 1 ▼

From **Charles McCathieNevile <charles@w3.org>** to **WAI AU Guidelines <w3c-wai-au@w3.org>** May 31, 2001 09:28

1.1 It is probable that we can arrange a telephone connection, to call in via a US bridge.
1.2 Are there people who are unable to make the face to face meeting, but would like us to have this facility?
1.3 Please respond as soon as possible
1.4 - the decision will be made early next week.
1.5 cheers
1.6 Charles McCN
1.7 --
1.8 Charles McCathieNevile http://www.w3.org/People/Charles phone: +61 409 134 136
1.9 W3C Web Accessibility Initiative http://www.w3.org/WAI fax: +1 617 258 5999
1.10 Location: 21 Mitchell street FOOTSCRAY Vic 3011, Australia
1.11 (or W3C INRIA, Route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex, France)

Email 2 ▼

From **William Loughborough <love26@gorge.net>** to **Charles McCathieNevile <charles@w3.org> , WAI AU Guidelines <w3c-wai-au@w3.org>** May 31, 2001 09:55

2.1 > Are there people who are unable to make the face to face meeting, but would like us to have this facility?
2.2 At least one "people" would.
2.3 --
2.4 Love.
2.5 ACCESSIBILITY IS RIGHT - NOT PRIVILEGE

Email 3 ▼

From **Marjolein Katsma <access@javawoman.com>** to **love26@gorge.net(William Loughborough), Charles McCathieNevile <charles@w3.org> , WAI AU Guidelines <w3c-wai-au@w3.org>** May 31, 2001 13:30

3.1 And at least on "people" would like you te be present
3.2 - even at the end of a phone line (though in person would be better, of course!)
3.3 > At 09:28 AM 5/31/01 -0400, Charles McCathieNevile wrote:
3.4 > > Are there people who are unable to make the face to face meeting, but would like us to have this facility?
3.5 > At least one "people" would.
3.6 > --
3.7 > Love.
3.8 > ACCESSIBILITY IS RIGHT - NOT PRIVILEGE
3.9 Marjolein Katsma
3.10 HomeSite Help - http://hshelp.com/
3.11 Bookstore for Webmasters - http://hshelp.com/bookstore/bookstore.html

Email 4 ▼

Figure 3.3: Annotating the email thread for speech act and meta features.

study included History, English, Philosophy, and Computer Science. We chose annotators with diverse backgrounds so that the summaries would not be biased in that respect. Annotators were screened to have satisfactory writing skills by submitting a paragraph describing their current occupation. The work was spread over several days and the annotators were be compensated for their work. The supervising researchers were certified to perform user studies on humans according to the Tri-Council Policy Statement. My certificate can be seen in the appendix A.

3.8 Corpus Statistics

In this section we present some statistical details about the BC3 corpus. This information should provide a more in depth understanding of the BC3 corpus.

Three annotators annotated each thread. There was a total of 10 different annotators that each did 12 annotations. The annotation of a thread took about 30 minutes and the annotators therefore worked in several sessions. As can be seen in Table 3.2, the email threads that were picked to be included in this corpus were fairly long with over six emails on average. We chose longer conversations to increase the need for summarization. Longer conversations are also more interesting because there is much more of a conversation structure than in a two message email thread. The summaries that were extracted had an average length of 17.5% of the original number of sentences. No guidelines were given on summary length. Therefore this is the length that annotators felt was appropriate. With shorter email threads, summaries would consist of only a couple of sentences or they would not maintain the same summarization ratio.

Annotators do not always agree on what an ideal summary should be [39]. For this reason we used three annotators per thread. To show how much these three annotators agreed, we use the Kappa statistic [21]. The coefficient is a statistic for inter-rater agreement. It is calculated as follows:

$$\kappa = \frac{\bar{P} - \bar{P}_e}{1 - \bar{P}_e}$$

This is Fleiss' kappa which measures reliability of more than two annotators. $\bar{P}$ represents the average inter-rater agreement. Kappa takes chance into account, which is represented by $\bar{P}_e$. By subtracting out $\bar{P}_e$, we are left with the percent agreement above chance. The kappa values calculated for all the different annotations can be seen in Table 3.3. In the BC3 corpus, the annotators had a kappa agreement of 0.50 for the extracted sentences. The kappa scores ranged from 0.21 to 0.55 for the binary annotated sentence features. In summarization low kappa values can be expected, since summarization is very subjective. However the guidelines for the sentence labels (speech acts, meta, and subjectivity) could be more tightly defined in an effort to increase kappa values. In their work of classifying requests and commitments, Lampert et al. [29] received higher kappa values in the range of 0.63 to 0.87. However they only used two annotators which were both professors of Computer Science and they also generalized different types of requests and commitments to come up with these kappa values. Nevertheless, it should be noted that high kappa scores do not always imply a good feature. Meta Sentences have the lowest kappa agreement, but out of all the labeled sentence features they increase weighted recall the most as can be seen in Table 4.4.

Corpus Statistic	Values
Number of Threads	40
Annotators per Thread	3
Average Length of Threads	6.425 Emails
Average Length of Threads	80 Sentences
Average Words per Thread	792 Words
Average Abstract Summary Length	122 Words (15.4% of original words)
Average Extractive Summary Length	14 Sentences (17% of original sentences)

Table 3.2: Various statistics on the BC3 corpus.

Annotations	κ
Extractive summary sentences	0.5
Linked sentences	0.49
Propose	0.33
Request	0.55
Commit	0.36
Meeting	0.49
Meta	0.21
Subjectivity	0.37

Table 3.3: Kappa agreement for the different annotations.

3.9 Availability

The BC3 annotated corpus is available for research purposes. The annotation framework is also made publicly available to promote further annotation. W3C is a huge dataset which means there are many more threads that can be annotated.

The BC3 corpus is released under the Creative Commons Attribution-Share Alike 3.0 Unported License, which means it can be used for commercial or non-commercial purposes. You are allowed to copy, share, distribute, or change the corpus as long as it is attributed and licensed with the same license or a compatible one.

The BC3 framework is released under the MIT license, which also allows for commercial and non-commercial uses. This license is specifically written for software and is also used to license Ruby on Rails. It has the same properties that allow you to copy, share, distribute, or change the corpus as long as it is attributed and licensed with the same license or a compatible one.

Both the corpus and the framework are available online at:

http://cs.ubc.ca/labs/lci/bc3.html

The website describes the corpus and gives an outline of the framework architecture to make it easier to change the framework. When downloading the software, the user is asked to provide some simple information such as their name and how many researchers will be using it. This helps us keep statistics on who is using the corpus.

Less than a month after launch, there are already 5 research groups for a total of 15 researchers from around the world using the BC3 corpus. Researchers that have downloaded the corpus belong to organizations including SRI International, ISI, and Simon Fraser University.

3.10 Summary

Due to the scarcity of annotated corpora for email thread summarization, we annotated a subset of the W3C corpus, a large publicly available email dataset. Having learned from previous corpora annotations, we chose to annotate the corpus with both extractive and abstractive summaries. Annotators created links between the abstractive summary sentences and the sentences in the original email thread. We also labeled additional sentence features that can be used in machine learning based summarization. These include the basic speech acts of individual sentences as well as whether a sentence is a meta sentence and whether it is subjective. The online framework that was used to annotate the corpus as well as the corpus itself are publicly available at http://cs.ubc.ca/labs/lci/bc3.html.

Chapter 4

Supervised Machine Learning Summarization

This chapter will describe our approach to summarizing email threads using supervised machine learning. As described in Chapter 2, there are two types of text summarization: extractive summarization, which is a sentence classification task, and abstractive summarization, which creates a rewritten summary of the original text. We focus on extractive summarization and we tackle the classification task by applying statistical machine learning methods.

We explore new techniques for email thread summarization by using several regression-based classifiers and novel features. As a preview we introduce our contributions below:

- In previous machine learning summarization approaches, even those that were using multiple annotators, the sentence labels in the training data were binary. This meant that a threshold had to be set to decide which sentences to include. Our method circumvents this by using a continuous value to label our sentences. This means no information is lost by setting a threshold to collapse the continuous value into a binary label.

- Existing studies base analysis on predetermined classifiers. There has not been a comparative study between different classifiers for email summarization. Our results show that there is quite a large variance among different classifiers, and that Bagging and Gaussian Processes achieve the best results.

- Linguistic discourse features have also not been explored in much detail. We will show that accurate speech act labels will improve the quality of summarization.

An outline of the chapter is as follows. In Section 4.1 we describe the regression-based classification framework and classifiers. Section 4.2 describes what features we are using for machine learning. In Section 4.3 we explain the experimental setup and summary evaluation. Then in Section 4.4 we present and explain our results. Finally in Section 4.5 we provide a summary of the chapter. In all our experiments, both the Enron and BC3 corpora were used for training and testing.

4.1 Regression-based Summarization

Often summarization is simplified to a binary classification of whether a sentence will be included in a summary or not. This simplifies the summarization task to training an extractive machine learning classifier. It is often the case that summaries of different lengths are required. A classifier with a binary output would have to be retrained for different lengths, which is a lot of work and makes the summarizer not generally applicable. The solution to this problem is to use a classifier with a posterior probability of including the sentence in a summary. Sentences with the highest probability can then be included in the summary up to the desired summary length.

There is however another limitation due to a binary classification approach in the training data of most summarization frameworks. Sentences in the training set need to be labeled as either included or not. Many times annotators do not agree on which sentences should be included in a summary as can be seen by kappa values of 0.5 in the BC3 corpus. So some sort of thresholding is needed to describe whether a sentence should be labeled as included or not. In summary evaluation the pyramid method [39] is based on the fact that humans disagree on ideal summaries as described in Section 2.5. This idea can be carried over to machine learning training. Our solution is to use a combined annotator score. This gives us a continuous importance range for sentence labels. In the BC3 corpus the score is the number of annotators that selected a sentence. For the Enron corpus the score is a linear combination of the annotators selections, where annotators can either select a sentence as essential, optional, or not important for the summary. Formally, the scores in the two corpora are computed as follow:

$$score(enron) = \sum_{i \in Annotators} 3 * \#ESS_i + \#OPT_i$$
$$score(BC3) = \sum_{i \in Annotators} \begin{cases} 1_i & \text{if selected;} \\ 0_i & \text{otherwise.} \end{cases}$$

These scores are then fed into a regression-based classifier to output an importance rank. Summaries can be created up to a desired length based on each sentence's classification score.

4.1.1 Normalization by Sentence Length

The summaries that are generated are limited to 30% of the original number of words in the document. Our extractive summarization approach works at the sentence level. To make our summarization approach generate concise summaries limited by word length, sentence length needs to be taken into account. We therefore normalize the sentence annotator score by sentence length. The normalized score represents the information or importance content of the sentence per word. If the annotator scores remain unnormalized, then the summary will include long sentences instead of suitable shorter substitute sentences.

4.1.2 Classifiers

With our continuous framework, we compared different regression-based classifiers. We used the WEKA [48] implementation of the different classification algorithms.

For our SVM classifier we used SMOreg which is an optimization algorithm for training support vector regression using a polynomial kernel. The algorithm implements sequential minimal optimization using extreme chunking by converting nominal attributes into binary ones and optimizing the target function for them.

Simple Linear Regression builds a linear regression model by repeatedly selecting the feature providing the lowest squared error. The sentence's relative position in the email was found to be the best feature.

For a tree based classifier, we used REPTree which is a regression based decision tree algorithm. It build the tree using information variance reduction.

Bagging was done on this decision tree classifier. Bagging is a bootstrapping algorithm which averages the output of several decision trees trained on random subsets of the training data. By using several bootstrapped training sets, an unstable classifier like decision trees improves stability and accuracy.

Gaussian Processes was used with an RBFKernel [33].

4.1.3 Two Summarization Techniques for Comparison

The supervised machine learning classifiers are compared to two unsupervised classifiers: ClueWordSummerizer(CWS) [7] and MEAD [41]. CWS uses the notion of clue words to score a sentence's importance. Clue words are stemmed words that are present in both parent and child nodes in a conversation graph. The intuition is that words that appear throughout a conversation are important and therefore the sentences that contain them are as well.

MEAD, as described in Section 2.1.3, is a summarization framework with three components: a feature generator, a classifier which computes sentence importance scores, and a sentence reranker. We use the framework in our summarization system by replacing the three components. However it comes with default components which have been shown to work on multi-document text summarization. These include the heuristic that any sentence with less than nine words is automatically discarded. Both centroid and position features are computed. The centroid feature is computed using tf*idf (term frequency * inverse document frequency) vectors. These vectors contain counts of all the words present in a particular sentence, where each word count is divided by the occurrence of that particular word in a large corpus of documents. To create a centroid score, the cosine similarity is computed for that particular sentence's tf*idf vector to the average document tf*idf vector. The second feature, position, is the reciprocal of the sentence's position. Thus earlier sentences have higher position scores. These two scores are combined by averaging them together to create the final sentence importance score. It is this default system that is meant by the label "MEAD" in the graphs.

4.2 Sentence Features

Proven features from multi-document text summarization and email specific features have previously been used for summarization. Due to the conversational nature of emails we have decided to add speech act labels for each sentence. For example, if an author commits to a future course of action, there is a high probability that this is an important sentence. The list of features can be seen in Figure 4.1.

The features in Table 4.1 are calculated for each sentence in the email thread. The work by Rambow et al. [42] showed that including email specific features improved recall and precision of important sentences. Therefore we start with their text and email specific features as a baseline comparison for our feature sets. There are small differences in the feature sets used in the two different corpora. In the original feature set used by Rambow et al. [42], one of the features was "fol_Quote" which signified whether the sentence followed text from a previous email. In our corpora the emails are preprocessed to remove repeated quoted content. We therefore no longer have the "fol_Quote" feature. The Enron corpus was extracted by work in [7]. In their extraction process, if the text was quoted and only occurred once in the available emails, they assumed the original email was lost and extracted this test as a hidden email. In the Enron corpus, "Is_Hidden" is therefore one of the available features. In the BC3 corpus "fol_Quote" is also no longer available as a feature, as the emails are preprocessed and quotes are removed manually. "Is_Hidden" is not available as a feature as the BC3 emails come from a mail server which has a recorded of all the emails and therefore no part of the conversation is missing. In the BC3 corpus a conversation structure was never analyzed beyond

Text Features	Thread Line Number	The position of the sentence in the thread.
	Relative Pos. in Thread	The position in the thread as a percentage.
	Centroid Similarity	The idf component is computed for the whole corpus.
	Local Centroid Similarity	The idf component is computed for the current thread.
	Length	The number of words in the sentence.
	TF*IDF Sum	The sum of the sentence's tf*idf values.
	TF*IDF Average	The average of the sentence's tf*idf values.
	Is Question	Whether the sentence is a question based on punctuation.
Email Features	Email Number	The temporal position of the current email in the thread.
	Relative Pos. in Email	The temporal position of the current email as a percentage.
	Subject Similarity	The content word overlap of the sentence with the subject line.
	Num_Replies	The number of replies based on the conversation structure.
	Num_Recipients	The number of email addresses in the To and Cc fields.
	Is Hidden	Whether the email is not in the inbox, but quoted in another email.
Generated Speech Act Features (at the sentence and email level)	Request	The sentence/email asks the recipient to perform some activity.
	Propose	The sentence/email proposes a joint activity.
	Commit	The sentence/email commits the sender to a course of action.
	Deliver	The sentence/email delivers information.
	Meet	The sentence/email concerns a joint activity constrained in time.
	Data	The sentence/email contains an attachment, link, or other data.
Annotated Speech Act Features	Request	The sentence asks the recipient to perform some activity.
	Propose	The sentence proposes a joint activity.
	Commit	The sentence commits the sender to a course of action.
	Meet	The sentence concerns a joint activity constrained in time.
Other Annotated Features	Meta	The sentence refers to this email thread.
	Subj	The writer is expressing an opinion or strong sentiment.

Table 4.1: Features used for classification arranged in groups.

the thread level. Therefore ”Num_Replies” is not available as a feature. We also add speech, meta, and subjectivity features to the BC3 corpus as explained below in more detail.

4.2.1 Analysis of the Feature Set Proposed in Previous Work

Before we decided to use all the previously used features in from previous work in email summarization, we tested their effectiveness on our corpora. Feature selection was performed on the fourteen baseline text and email features from Rambow et al. [42], called R14 in the figures. Greedy forward selection found that all features increased merit on both the Enron and BC3 corpora. We therefore used all fourteen features as a baseline in our system.

4.2.2 Speech Acts as Additional Features

Email is a more formal communication method than online messengers or blogs. When people use email, they usually have a specific purpose in mind. This specific purpose is often shrouded in small talk to make a request less direct. In summarization the specific purpose for writing an email is the important part. We therefore want to use speech acts which embody this specific purpose.

Email threads in which speech acts, such as request and commit, are important features include those that try to schedule meetings, assign responsibilities, or plan strategies. These types of email threads frequently occur in both of our corpora and are common for many organizations. For example the following two sentences were selected by our algorithm using speech act features, but the algorithm missed them if these features were not included:

> When is the deadline?
>
> I’m prepared to decide by email so we can formally respond by email.

These sentences come from a thread in the BC3 corpus that is concerned with rescheduling a teleconference. The first sentence is a request to find out when the deadline is to formally respond to previous comments. The second sentence is a proposal to formally respond by email.

As features we use automatically annotated speech acts using Ciranda [11] as well as manually annotated speech acts from the BC3 corpus as specified in Section 3.5.1. Ciranda[1] is a publicly available tool that classifies emails as *Propose*, *Request*, *Commit*, *Deliver*, *Meeting*, and *deliveredData*. These categories are not

[1] http://www-2.cs.cmu.edu/~vitor/codeAndData.html

mutually exclusive. Since a smaller granularity would be useful for email summarization, we ran the classifier individually on each sentence (SA_S) as well as on the email level (SA_E). These features can be seen in their different groupings in Figure 4.1.

4.3 Experimental Setup

A machine learning summarizer uses a classifier to compute a sentence importance score for each sentence. This score is then used to decide which sentences to include in the summary. Thirty-two features were generated for each sentence in the email corpus that was used. The summarizer used two software packages in its implementation. MEAD [41] was used as the summarization framework. It works in three components. First features are generated, then a classifier scores sentences, and finally a reranker produces the final summary. WEKA [48] was used as the implementation of machine learning classification algorithms. These two packages were combined to create several different machine learning summarizers. The tests were run to create summaries that were limited by length to 30% the original number of words.

To evaluate the different summarizers, the corresponding summaries were compared to a human generated gold standard. The machine learning summarizers were evaluated using 10-fold cross-validation. This was performed on the Enron corpus, which was introduced in Section 2.4.2 and the BC3 corpus, which was introduced in Chapter 3. The Enron corpus had 39 threads and the BC3 corpus had 40 threads. Thus the email threads were divided into ten sets where 90% of the data was used for training and 10% of the data was used for testing. Since the Enron corpus could not be split evenly, we we chose 4 test threads randomly for each of the 10 cross-validations and then randomly added the others as training data. This left us with 40 summaries for each corpus that could be compared to the corresponding gold standard.

4.3.1 Evaluation Metric: Weighted Recall

Weighted recall was used to evaluate the results of both datasets. This was repeated for all the different classification algorithms. Human users almost always cannot agree on a single perfect summary for a given email thread. We therefore have several annotators summarize an email thread and then score the machine-generated summary against all of them. In the Enron corpus there were 5 annotators per thread and in the BC3 corpus we had 3 annotators per thread. We measure the recall score against an ideal summary weighed across all annotators. The recall score measures the percentage of important sentences that were chosen and the precision score measures the percentage of the chosen sentences that were important. The f-measure is the harmonic mean of precision and recall. The reason we use a recall score instead of an f-measure, is because the length of the summary is fixed. Since our summaries are limited by length, it is much more interesting what the fraction of relevant sentences is, i.e. the recall score. This score has also been called pyramid precision in [8], although in weighted recall the gold standard (GS) summary is the highest scoring summary given a summary length of 30% and in [8] it is the highest scoring sentences up to a length of 30%. In the Enron corpus sentences were annotated as essential, optional, or not selected, while in the BC3 corpus sentences were either selected or not.

$$WeightedRecall = \frac{\sum_{i \in Sent_{Sum}} score_i}{\sum_{i \in Sent_{GS}} score_i}$$

$$score(enron) = \sum_{i \in Annotators} 3 * \#ESS_i + \#OPT_i$$
$$score(BC3) = \sum_{i \in Annotators} \begin{cases} 1_i & \text{if selected;} \\ 0_i & \text{otherwise.} \end{cases}$$

We also performed some initial evaluations with ROUGE (see Section 2.5), but they did not agree with the weighted recall results. The ROUGE evaluation metric was created for newswire data. It has not been shown to correlate well with human evaluations in the meeting domain [31], which is similar to emails in informality and conversational nature. We therefore chose to use weighted recall as our evaluation metric.

4.4 Results

In our summary algorithm evaluation, we compared different classifiers, with different feature sets and varying types of annotated class labels. With so many possible configurations for the summary algorithms, it becomes hard to understand the results. We provide two tables (Table 4.3 and Table 4.4) that present all of the details of our evaluation. To better understand this vast amount of information, we describe each of the different aspects separately. In the following sections we provide a description of our results as we compared different classifiers, continuous vs. binary labels, raw vs. normalized labels, and different feature sets.

4.4.1 Statistical Analysis

For statistical analysis we first performed the ANOVA test to see if there was any significance in the results. In all of our comparisons this was the case so we subsequently performed individual t-tests to test for significance between the features.

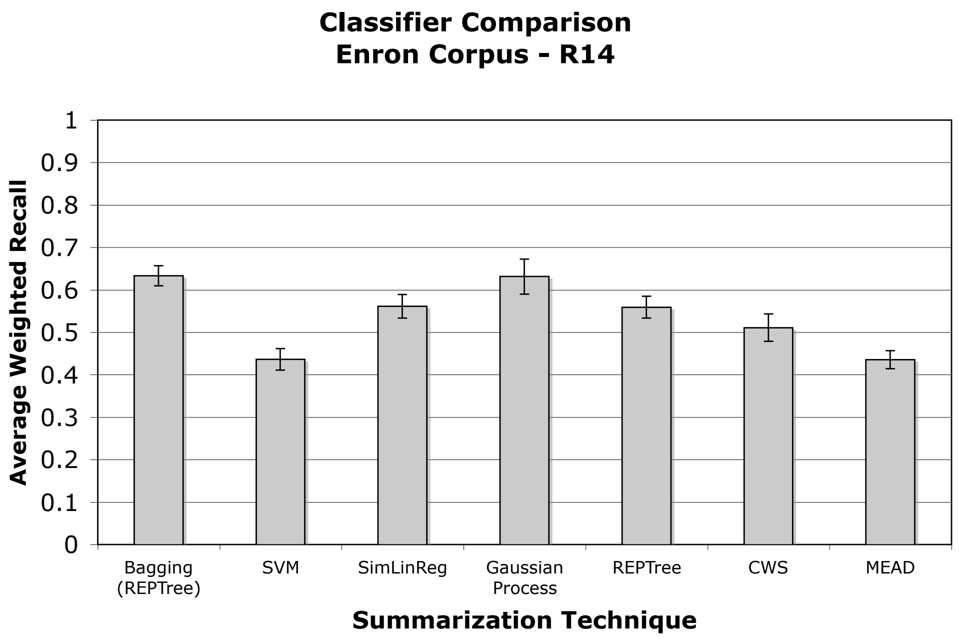


Figure 4.1: Summarization comparison on the Enron dataset. R14 are the original features presented in [42].

For the comparisons, two-tailed t-tests were always performed and if the null hypothesis was rejected with at least 95% confidence, the result was considered significant. A Bonferroni correction was performed due to the repeated testing on the same annotated corpora. Because of the Bonferroni correction the 95% confidence threshold for significance in the Enron corpus was moved from 0.025 to 0.0008 and for the BC3 corpus to 0.0004.

4.4.2 Comparing Different Classifiers

For comparing the different classifiers we have chosen to use the baseline feature set, R14, and normalized continuous sentence labels. The reason for this is that the baseline feature set is available for both corpora and the normalized continuous sentence labels provide the best overall results. In Figure 4.1 we compare the performance of 7 summarization approaches on the Enron dataset. The first 5 are supervised while the last 2 are unsupervised. On the y axis we measure weighted recall as described in Section 4.3.1 and on the x axis we list the different classifiers. It can be seen that the supervised machine learning algorithms outperform the unsupervised versions. Bagging of the REPTree classifier had the highest av-

	Bagging	GP	SimLinReg	REPTree	Clue	SVM	MEAD
Bagging	*	0.90	$\mathbf{8.0 * 10^{-6}}$	$\mathbf{2.8 * 10^{-6}}$	$\mathbf{1.5 * 10^{-8}}$	$\mathbf{5.9 * 10^{-13}}$	$\mathbf{1.3 * 10^{-13}}$
GP		*	**.0003**	**0.0002**	$\mathbf{8.9 * 10^{-7}}$	$\mathbf{1.8 * 10^{-10}}$	$\mathbf{8.4 * 10^{-11}}$
SimLinReg			*	0.85	0.0015	$\mathbf{4.2 * 10^{-9}}$	$\mathbf{1.3 * 10^{-9}}$
REPTree				*	0.0016	$\mathbf{2.7 * 10^{-9}}$	$\mathbf{7.6 * 10^{-10}}$
Clue					*	$\mathbf{1.8 * 10^{-5}}$	$\mathbf{7.5 * 10^{-6}}$
SVM						*	0.92
MEAD							*

Table 4.2: Significance of T-test comparisons between classifiers for the Enron dataset is shown in bold with the Bonferroni correction. $H_A : Y_{axis} - X_{axis} \neq 0$. P-values are shown for the better classifiers in each pairwise comparison.

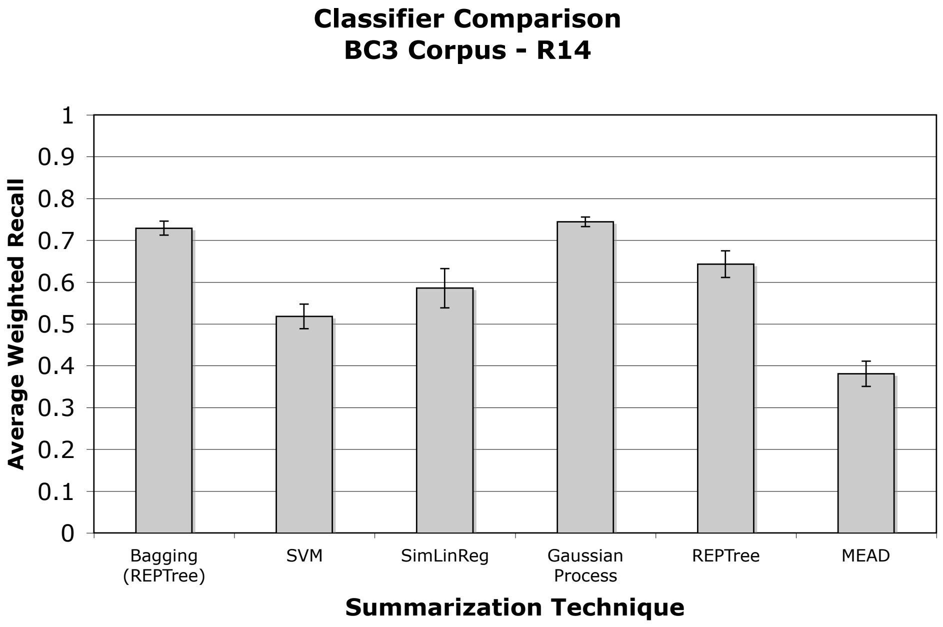


Figure 4.2: Comparison of different classifiers in the BC3 corpus. R14 are the original features presented in [42].

erage weighted recall with a score of 0.63. Bagging performs significantly better than any other approach except for gaussian processes, as can be seen in Figure 4.2. This figure shows which algorithms are significantly better then others and the associated p-values.

Figure 4.2 illustrates that the results hold for the BC3 corpus as well. Bagging and gaussian processes therefore seems to be the most effective classifiers for email summarization. They are useful in summarization as they does not overfit the training data. We were somewhat surprised by how low SVM's weighted recall score was, but it was confirmed in both datasets. It would be interesting to see whether kernels other than the polynomial one perform any better with SVM. As will be shown in Section 4.4.3, SVMs are actually better suited for a binary framework.

CWS was developed using the Enron dataset [7] and it does better than MEAD, the other unsupervised summarizer. However the supervised approaches except SVM outperform it. MEAD is used as a baseline since it has not been designed for emails but standard text instead.

4.4.3 Continuous vs Binary Labels

In this section we are focusing on the difference between using a binary label and a continuous label that is normalized by sentence length for training data. The binary label is generated by taking the annotation score and using a threshold to decide whether to include a sentence or not. The threshold used in the Enron corpus was 8 and the threshold used in the BC3 corpus was 2. Eight was also the threshold level that was used for the Enron annotation in [7]. Two was selected as the threshold in the BC3 corpus as it signifies that the majority of the annotators wanted to include the sentence in the summary.

We were not able to compare Simple Linear Regression and Gaussian Processes as these two algorithms cannot be used with binary class labels in WEKA, which we used as algorithm implementations. We have also included Ripper in this evaluation as this was the algorithm used in previous work on extractive email summarization [42].

Figure 4.3 and Figure 4.4 show our results graphically for the Enron and BC3 corpus respectively. It can be seen that the best results are achieved using a continuous normalized framework and the bagging algorithm. The continuous labels using bagging were significantly better with p-values of less than 0.00001 compared to binary bagging, Ripper, and MEAD individually. However not all algorithms are suited for a continuous regression setting. SVM actually performs better in the binary framework. We hypothesis that this is because SVM is margin maximizing algorithm and that this works best when having only two classes.

Feature Set	Classification Algorithm			Enron WR	Std. Dev.	BC3 WR	Std. Dev.
R14	Continuous	Normalized	Bagging	0.63	0.024	0.73	0.017
			GP	0.63	0.041	0.74	0.011
			SimLinReg	0.56	0.028	0.59	0.047
			REPTree	0.56	0.026	0.64	0.032
			SVM	0.44	0.025	0.52	0.030
R14	Continuous	Unnormalized	Bagging	0.43	0.028	0.64	0.020
			GP	0.42	0.028	0.64	0.026
			SimLinReg	0.37	0.025	0.59	0.047
			REPTree	0.40	0.024	0.62	0.016
			SVM	0.39	0.025	0.52	0.032
R14	Binary		Bagging	0.52	0.032	0.57	0.020
			REPTree	0.51	0.030	0.39	0.021
			SVM	0.52	0.024	0.54	0.027
			Ripper	0.52	0.021	0.54	0.023
	Unsupervised		Clue	0.51	0.032	N/A	N/A
			MEAD	0.44	0.021	0.38	0.030

Table 4.3: Comparing different types of classifiers for summarization using weighted recall for evaluation and the baseline R14 feature set.

Feature Set	Classification	Enron WR	Std. Dev.	BC3 WR	Std. Dev.
R14	Bagging	0.63	0.024	0.73	0.017
	GP	0.63	0.041	0.74	0.011
	SimLinReg	0.56	0.028	0.59	0.047
	REPTree	0.56	0.026	0.64	0.032
	SVM	0.44	0.025	0.52	0.030
R14+SA_S	Bagging	0.64	0.027	0.69	0.019
	GP	0.63	0.035	0.72	0.011
	SimLinReg	0.56	0.028	0.59	0.047
	REPTree	0.58	0.035	0.60	0.032
	SVM	0.41	0.027	0.53	0.040
R14+SA_E	Bagging	0.62	0.036	0.70	0.024
	GP	0.59	0.037	0.69	0.013
	SimLinReg	0.56	0.028	0.59	0.047
	REPTree	0.58	0.020	0.58	0.036
	SVM	0.41	0.021	0.53	0.032
R14+CWS	Bagging	0.62	0.031	N/A	N/A
	GP	0.64	0.039	N/A	N/A
	SimLinReg	0.56	0.028	N/A	N/A
	REPTree	0.56	0.025	N/A	N/A
	SVM	0.46	0.027	N/A	N/A
R14+SA_A	Bagging	N/A	N/A	0.76	0.012
	GP	N/A	N/A	0.77	0.013
	SimLinReg	N/A	N/A	0.59	0.047
	REPTree	N/A	N/A	0.67	0.025
	SVM	N/A	N/A	0.73	0.020
R14+Meta	Bagging	N/A	N/A	0.77	0.012
	GP	N/A	N/A	0.76	0.011
	SimLinReg	N/A	N/A	0.59	0.047
	REPTree	N/A	N/A	0.66	0.031
	SVM	N/A	N/A	0.60	0.024
R14+Subj	Bagging	N/A	N/A	0.74	0.015
	GP	N/A	N/A	0.75	0.011
	SimLinReg	N/A	N/A	0.59	0.047
	REPTree	N/A	N/A	0.65	0.027
	SVM	N/A	N/A	0.67	0.019
R14+Meta+ Subj+SA_A	Bagging	N/A	N/A	0.80	0.012
	GP	N/A	N/A	0.79	0.012
	SimLinReg	N/A	N/A	0.59	0.047
	REPTree	N/A	N/A	0.67	0.020
	SVM	N/A	N/A	0.77	0.017

Table 4.4: Comparing feature sets in a continuous and normalized setting.

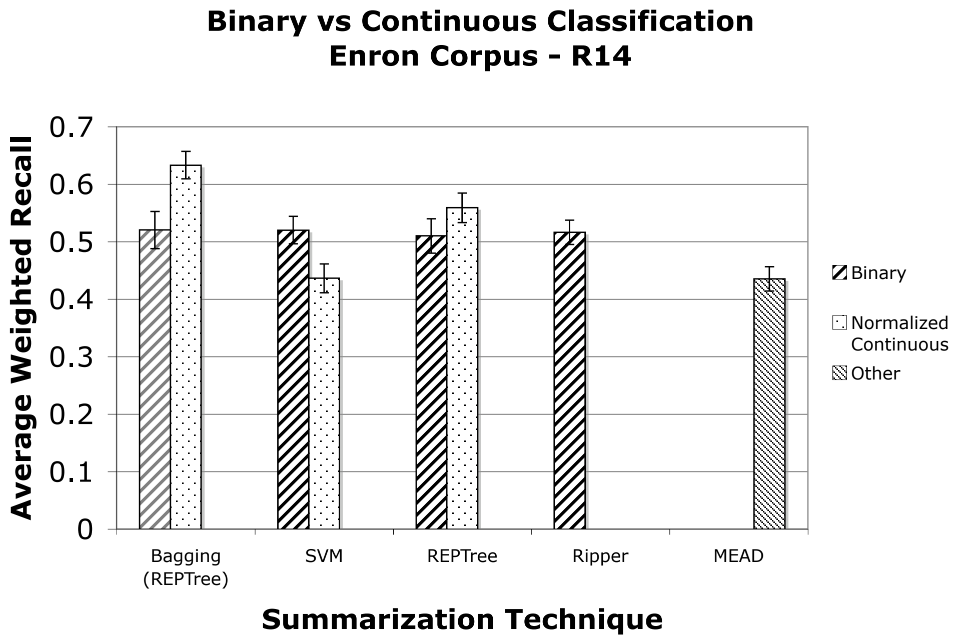


Figure 4.3: Performance differences between binary and continuous classifiers in the Enron corpus.

4.4.4 Normalization by Sentence Length

We compared the classifier performance using annotator score labels that were normalized by sentence length with those that were not. These comparisons were performed using the R14 feature set. As can be seen in Figure 4.5 the average weighted recall using the normalized labels was better than using the unnormalized labels for all algorithms in the Enron dataset. In the BC3 corpus the results are similar with all the algorithms performing better or no worse using the normalized sentence labels. This proves our hypothesis that it is important to normalize the training data by length since our summaries are limited by word count.

4.4.5 Feature Sets

The baseline feature set, R14 (from [42]), was compared to additional speech act features, meta labels, and subjectivity labels. In this comparison we use the normalized continuous sentence labels as they provide the best performance. Automatically labeled speech acts were also compared to manually labeled speech acts. Ciranda was used to automatically label the speech acts at the email level (SA_E)

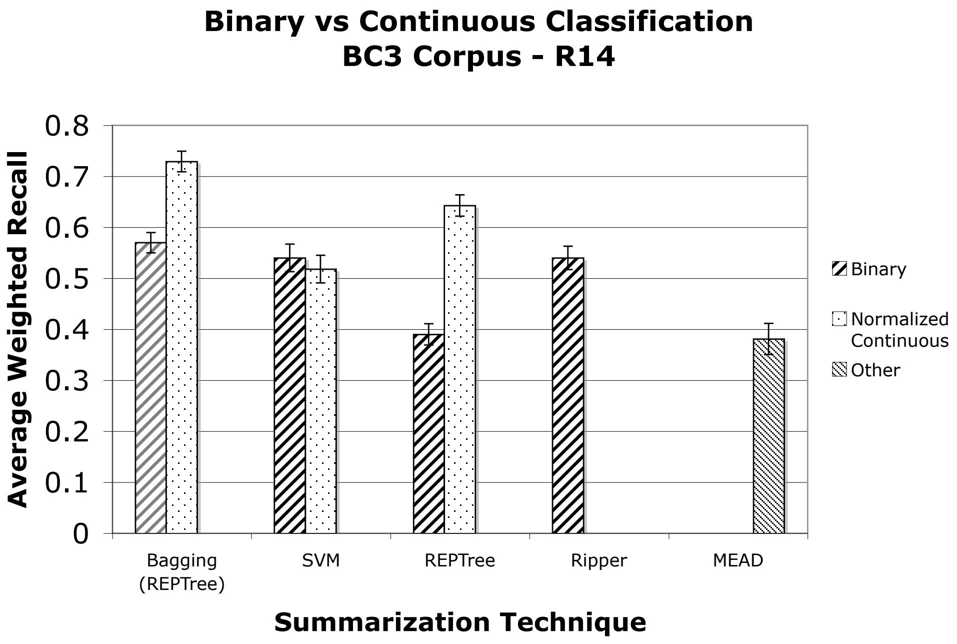


Figure 4.4: Performance differences between binary and continuous classifiers in the BC3 corpus.

and at the sentence level (SA_S). In the BC3 corpus manual speech act annotations (SA_A) were also available. Manual annotations for meta sentences (Meta) and subjective sentences (Subj) were also available in the BC3 corpus.

Analysis of the new features shows that the results depended on the quality of the features as can be seen in Figure 4.6. The generated speech act labels with Ciranda did not help with summarization. This included both email level annotation and sentence level annotation. However the manually labeled features were significantly better than the baseline as can be seen in Table 4.5. The generated features did not perform well for different reasons. The email level annotations were too coarse to help select which sentences should be selected for summarization. The sentence level annotations were too noisy as there was not enough data for Ciranda to select the correct label. Ciranda was trained to label emails, not sentences. The fact that the manually generated labels were useful for summarization shows that speech acts are indeed a useful feature, the generated labels we used are just too noisy to be useful. It would be useful to pursue automatic speech act classification at the sentence level.

Meta sentences had the highest increase in weighted recall of all the new fea-

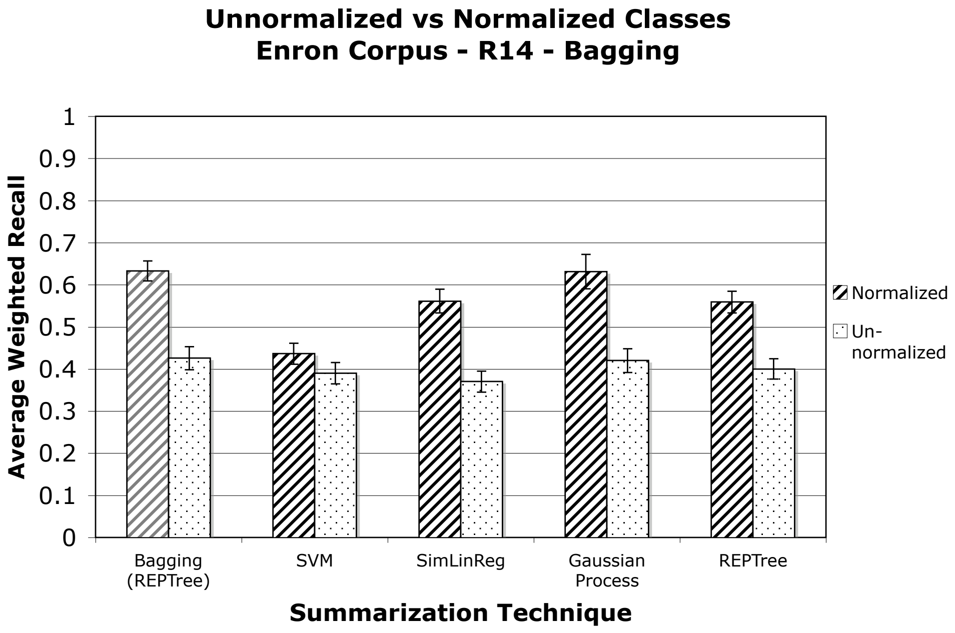


Figure 4.5: A comparison of summarizer performance using annotators scores normalized by sentence length. R14 are the original features presented in [42].

tures by themselves. This was somewhat surprising as meta sentences had the lowest kappa agreement between all the annotators. This shows that it is not necessarily important for all the annotators to agree.

In Figure 4.7 it can be seen that again the automatically generated speech acts did not improve weighted recall significantly. In the Enron corpus, manually annotated speech acts are not available. An additional feature, CWS, which is the clue word score as generated in previous work [7], surprisingly also did not improve weighted recall significantly. It seems that clue words are a good feature by themselves in the clue word summarizer, but they are not a useful additional feature in a supervised machine learning system.

4.4.6 Classifier Runtime Analysis

When the summarization algorithm is intended for a real application, runtime performance is also an important aspect. Users cannot wait for the summary to be generated or else the purpose of summarization is defeated. In this section we compare the amount of time the different algorithms take to create summaries. The

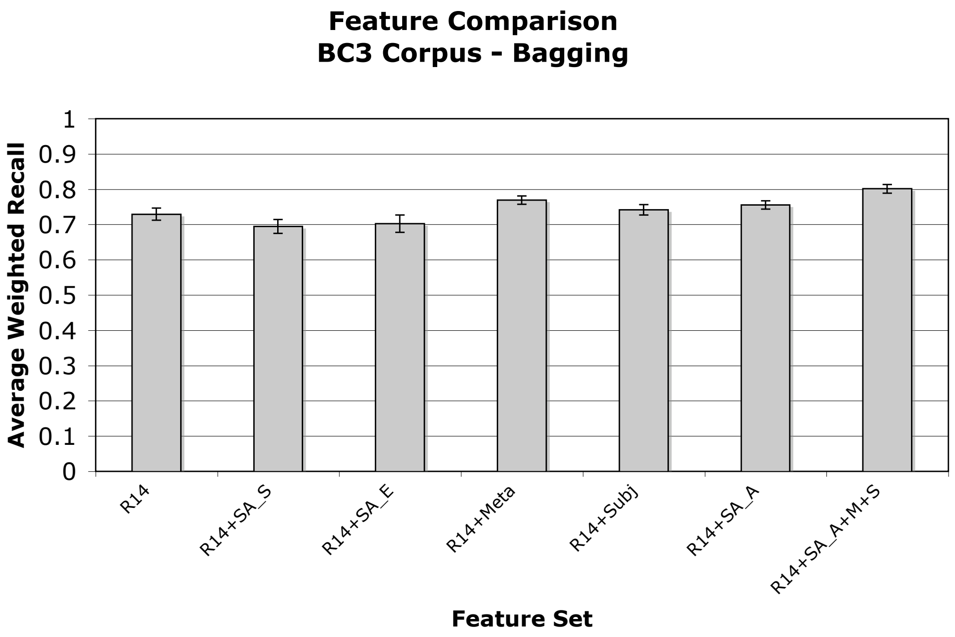


Figure 4.6: Comparison of different features on the BC3 corpus. R14 are the original features presented in [42]. SA_A are the annotated labels, SA_S are the sentence level Ciranda [11] labels and SA_E or the email level Ciranda labels. Meta (M) and Subj (S) are manually annotated labels.

	Recall	P-value
R14	0.73	
R14+SA_S	0.69	**0.0005**
R14+SA_E	0.70	0.01
R14+Meta	0.77	**0.00001**
R14+Subj	0.74	0.095
R14+SA_A	0.76	**0.0009**
R14+SA_A+M+S	0.80	$\mathbf{2.3 * 10^{-9}}$

Table 4.5: Average weighted recall and significance with Bonferroni correction of T-test comparisons against the R14 feature set for the BC3 dataset.

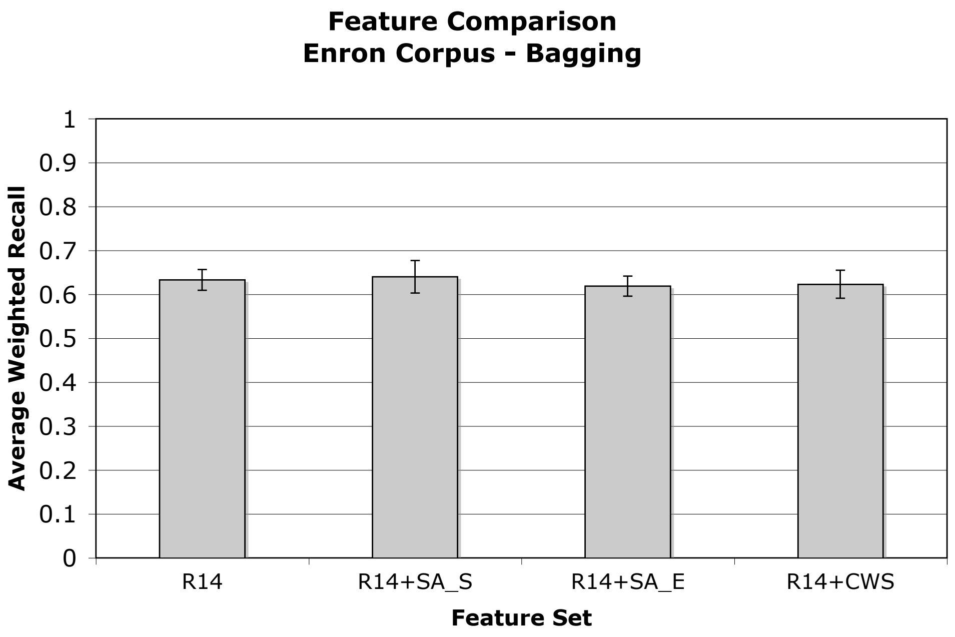


Figure 4.7: Comparison of different features on the Enron corpus. R14 are the original features presented in [42]. SA_S are automatically generated sentence level labels and SA_E are automatically generated email level Ciranda labels. CWS is the clue word score [7].

feature extraction time was not compared as it is the same for each of the machine learning algorithms. The process was broken into two steps: model creation and summary generation. In the model creation step, the algorithm was trained using the features and labels in the training data. In the summary generation step, this model was used to create a text summary.

The recorded times have been averaged over several runs, and represent training on 36 email threads and generating one summary. The summaries were generated using the R14 feature set that was used as a baseline in this thesis. The test was performed on a dual-core Pentium 4 3.2Ghz with 2GB of RAM. The summarization system was run on Suse Linux 10.3.

The results are displayed in Figure 4.8. The y axis represents the amount of time the algorithm takes to run in seconds. The axis is broken in two places so that the timing of the faster algorithms is still readable. The timings are stacked with the summary generation time on the bottom and the algorithm training time on top. There are two stacks for each algorithm - the left one displaying performance on the Enron corpus and the right one displaying performance on the BC3 corpus.

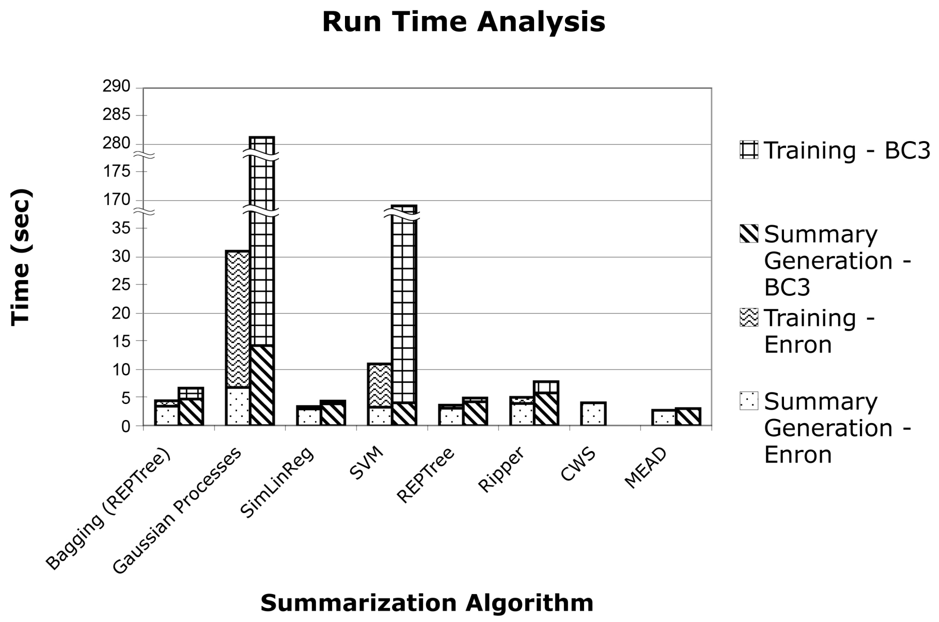


Figure 4.8: A runtime analysis of different summarization algorithms.

It can be seen that Gaussian Processes and SVMs took a much longer time to build the model than the other techniques. This is because these two classifiers are more complicated than rule-based classifiers such as REPTree and simple linear regression.

Comparing the results for the Enron and BC3 corpora, the email threads in the BC3 corpus took longer to summarize for all the methods. This effect was most pronounced in the Gaussian Processes and SVM, which were already the slowest in the Enron corpus. This is most likely the case because in the BC3 corpus the email threads were longer both in the number of emails and the amount of text than in the Enron corpus. We do not know why the difference in runtime is so significant in the BC3 corpus except that the increased thread length might exacerbate the slowness already seen in the Enron corpus.

4.5 Summary

We created a continuous classification framework for summarizing email threads and evaluated it on two corpora, one of which was developed for this study. Our results show that the best regression-based classifiers for email thread summarization perform better than binary classifiers because they preserve more information. In our comparison between different classifiers, we found that Bagging and Gaussian Processes have the highest weighted recall. We confirm the 14 features from [42] provide good results but can be improved with other features. The results on our new dataset show that speech acts are a very useful feature if they can be generated with higher accuracy at the sentence level. Meta sentences and subjectivity were also shown to be useful features for email summarization.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

In this thesis we described how to summarize email threads using machine learning. Our summarization approach was extractive which meant that we selected which sentences in the original text to keep for the summary. Since we used machine learning, we also required a dataset. To our knowledge there were no publicly available annotated datasets for email summarization. We therefore annotated the BC3 corpus and made it publicly available for download and extension. To assist researchers in contributing to the corpus, we are also releasing the annotation framework we used as open source software.

The annotation of the BC3 corpus allowed us to train and evaluate our summarization algorithm on two corpora: the BC3 corpus and the Enron corpus. We compared different machine learning algorithms and found there were significant differences between different classifiers. Bagging with regression trees was found to be the best algorithm as it was as accurate as gaussian processes but much more efficient. We also switched from a binary framework, which was used in previous email summarization by extraction, to a continuous framework that prevents information loss by not requiring a threshold. We also tested new features for email summarization. In addition to 14 features previously used in email summarization [42], we also used speech acts, meta sentences, and subjectivity. I will outline these achievements in more detail below:

- We released the BC3 corpus consisting of 40 annotated email threads. Each email thread was annotated by 3 annotators with extractive summaries and abstractive summaries with links. Each sentence in the email thread was annotated for speech acts, meta sentences, and subjectivity. The corpus is intended for use with extractive or abstractive email summarization (or a combination of the two). The web-based annotation framework is also available for download. The framework was built with Ruby on Rails and allows researchers to manage their email threads and previous annotations. It also allows researchers to schedule and perform annotation studies.

- Our summarization approach used regression-based machine learning. As our class labels for all the original sentences we used the sum of all the annotator's annotations. Therefore we could use all the data the corpus provided for us without having to use a threshold. These annotator scores were then normalized by sentence length, as our target summaries were limited in word length. We trained regression algorithms on our corpus data. This gave us a model that could give a new sentence an annotation importance score. The final summary could be constructed by repetitively adding the highest scoring sentence and reducing redundancy.

- We compared different machine learning algorithms and found that bagging with regression trees and gaussian processes performed significantly better than the other algorithms according to a weighted recall evaluation. An algorithm runtime analysis revealed that Gaussian processes took significantly longer to train and generate summaries than bagging with regression trees. This led to our conclusion that bagging with regression trees is the most suitable algorithm for email thread summarization.

- Selecting the right features to use in machine learning is critical to a successful algorithm. In this thesis we also compared different feature sets. As a baseline we used the 14 features previously used by Rambow et al. [42]. The new features we used were speech acts, meta sentences, and subjectivity. Speech acts consisted of several binary features that were not mutually exclusive including: Request, Commit, Proposal, and Meeting. All these features were manually annotated in the BC3 corpus at the sentence level. They were shown to significantly improve the weighted recall of generated summaries when added to the baseline feature set. We also wanted to see if these features could be generated automatically and still be useful. We used Ciranda [11] to automatically classify emails and sentences as speech acts. These features proved not to improve weighted recall of the generated summaries significantly.

5.2 Future Work

Summarization of emails is a relatively new field. There are many areas that have yet to be explored. Research can also be done on similar but newer forms of communication such as online chats or blog discussions. The following are some topics that would be interesting to explore.

- Since we have two corpora available, it would be interesting to train on the BC3 corpus and test on the Enron corpus and vice versa. This would provide external validity to our summarization approach. Similarly we could also train and test on the Enron and BC3 corpora combined.

- Computing the human ceiling would be useful in helping interpret the weighted recall results. This would be done by calculating the weighted recall for one of the annotators summaries while building the gold standard from the other two.

- The BC3 corpus was built to be used for both abstractive and extractive email summarization. This thesis only focuses on extractive email summarization, which means there is a great opportunity to continue this work into abstractive email summarization. It would be reasonable to build a system that combines extractive and abstractive summarization. Sentences could be extracted and then combined to make the summaries more concise.

- Another idea for future research would be to try boosting for sentence extraction. Our results show that Bagging was the most successful algorithms out of the ones we compared. Boosting also has potential since it is a similar meta-algorithm. We did not compare a boosting algorithm because we used WEKA, which did not contain an implementation for using boosting for regression.

- Subjectivity has been shown to be a very useful feature for email summarization. However we only used a binary variable to represent whether a sentence was subjective or not. More detailed labeling, with polarity and the author's intensity would most likely be even more useful for email summarization.

- Extending this research into other forms of communication would also be interesting. Although online chats are less structured than emails, it would be interesting to see if the same techniques can be applied in this field.

- Finally, it is important to consider how to present the summary information to the user. In order for the generated summaries to have the highest impact on the user the need to presented in a unobtrusive method that fits into the user's work flow.

Bibliography

[1] J. Anderson and G. Hinton. *Parallel models of associative memory*. Lawrence Erlbaum Assoc., Hillsdale, NJ, 1981. → pages 8

[2] P. Bailey, N. Craswell, I. Soboroff, and A. de Vries. The CSIRO enterprise search test collection. 2007. → pages 19

[3] R. Bekkerman, A. McCallum, and G. Huang. Automatic categorization of email into folders: Benchmark experiments on Enron and SRI corpora. Technical Report IR-418, Center of Intelligent Information Retrieval, UMass Amherst, 2004. → pages 18

[4] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996. → pages 9

[5] J. Carbonell and J. Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336, 1998. → pages 7

[6] G. Carenini, R. Ng, X. Zhou, and E. Zwart. Discovery and regeneration of hidden emails. *Proceedings of the 2005 ACM symposium on Applied computing*, pages 503–510, 2005. → pages 19

[7] G. Carenini, R. T. Ng, and X. Zhou. Summarizing email conversations with clue words. *16th International World Wide Web Conference (ACM WWW'07)*, 2007. → pages ix, 10, 12, 19, 20, 33, 34, 41, 46, 48

[8] G. Carenini, R. Ng, and X. Zhou. Summarizing emails with conversational cohesion and subjectivity. *The 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2008)*, June 2008. → pages 4, 22, 38

[9] J. Carletta. Unleashing the killer corpus: experiences in creating the multi-everything ami meeting corpus. In *Proc. of LREC 2006, Genoa, Italy*, pages 181–190, 2006. → pages 13, 21

[10] J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, W. Kraaij, M. Kronenthal, G. Lathoud, M. Lincoln, A. Lisowska, I. McCowan, W. Post, D. Reidsma, and P. Wellner. *The AMI Meeting Corpus: A Pre-announcement*, pages 28–39. Springer Berlin, 2006. → pages xi

[11] V. R. Carvalho and W. W. Cohen. On the collective classification of email ”speech acts”. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 345–352, New York, NY, USA, 2005. ACM. ISBN 1-59593-034-5. doi:http://doi.acm.org/10.1145/1076034.1076094. → pages ix, 12, 21, 22, 36, 47, 52

[12] W. Cohen. Learning Trees and Rules with Set-Valued Features. *AAAI/IAAI*, 1:709–716, 1996. → pages 11

[13] W. Cohen, V. Carvalho, and T. Mitchell. Learning to classify email into “speech acts”. In D. Lin and D. Wu, editors, *Proc. of EMNLP 2004*, pages 309–316, Barcelona, Spain, July 2004. Association for Computational Linguistics. → pages 19

[14] S. Corston-Oliver, E. Ringger, M. Gamon, and R. Campbell. Task-focused summarization of email. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, 2004. → pages 11

[15] C. Cortes and V. Vapnik. Support-Vector Networks. *Machine Learning*, 20 (3):273–297, 1995. → pages 8, 9

[16] G. DeJong. An overview of the FRUMP system. *Strategies for Natural Language Processing*, pages 149–176, 1982. → pages 6

[17] A. Dempster, N. Laird, D. Rubin, et al. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977. → pages 8

[18] P. DOMINGOS and M. PAZZANI. On the Optimality of the Simple Bayesian Classifier under Zero-One Loss. *Machine Learning*, 29:103–130, 1997. → pages 8

[19] R. Duda, P. Hart, et al. *Pattern classification and scene analysis.* → pages 8

[20] H. Edmundson. New Methods in Automatic Extracting. *Journal of the Association for Computing: Machinery*, 16(2):264–285, 1969. → pages 5

[21] J. Fleiss et al. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382, 1971. → pages 27

[22] Y. Freund and R. Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997. → pages 9

[23] V. Hatzivassiloglou, J. Klavans, M. Holcombe, R. Barzilay, M. Kan, and K. McKeown. Simfinder: A flexible clustering tool for summarization. *Proceedings of the NAACL Workshop on Automatic Summarization*, pages 41–49, 2001. → pages 14

[24] J. Heer. Exploring enron: A sketch of visual data mining of e-mail archives. *Proceedings of Email Archive Visualization Workshop*, 2005. → pages 18

[25] A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, and C. Wooters. The ICSI meeting corpus. In *Proc. of IEEE ICASSP 2003, Hong Kong, China*, pages 364–367, 2003. → pages 21

[26] M. Kan, K. McKeown, and J. Klavans. Domain-specific informative and indicative summarization for information retrieval. *Proc. of the Document Understanding Conference (DUC)*, pages 19–26, 2001. → pages 6

[27] B. Klimt and Y.Yang. *The enron corpus: A new dataset for email classification research.*, pages 217–226. 2004. → pages 13, 18

[28] K. Knight and D. Marcu. Statistics-based summarizationstep one: Sentence compression. *Proceeding of The 17th National Conference of the American Association for Artificial Intelligence (AAAI-2000)*, pages 703–710, 2000. → pages 6

[29] A. Lampert, R. Dale, and C. Paris. The nature of requests and commitments in email messages. In *Proceedings of the AAAI Workshop on Enhanced Messaging*, number Tech Report WS-08-04, pages 42–47. AAAI Press, 2008. → pages 27

[30] C.-Y. Lin and E. Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. *Proceedings of Language Technology Conference (HLT-NAACL)*, 2003. → pages 14, 15

[31] F. Liu and Y. Liu. Correlation between rouge and human evaluation of extractive meeting summaries. *The 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2008)*, June 2008. → pages 15, 38

[32] H. Luhn. The automatic creation of literature abstracts. *IBM Journal Res. Develop.*, 2:159–165, April 1958. → pages 5

[33] D. J. Mackay. Introduction to gaussian processes. 168:133–165, 1998. → pages 33

[34] N. Madnani, D. Zajic, B. Dorr, N. Ayan, and J. Lin. Multiple Alternative Sentence Compressions for Automatic Text Summarization. *Proceedings of the 2007 Document Understanding Conference (DUC-2007) at NLT/NAACL*, 2007. → pages 24

[35] K. McKeown and D. Radev. Generating summaries of multiple news articles. *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 74–82, 1995. → pages 7

[36] K. McKeown, R. Passonneau, D. Elson, A. Nenkova, and J. Hirschberg. Do Summaries Help? A Task-Based Evaluation of Multi-Document Summarization. *Proceedings of the 28th Annual ACM SIGIR Conference on Research and Development in Information Retrieval, Salvador, Brazil*, 2005. → pages 6

[37] K. McKeown, L. Shrestha, and O. Rambow. *Using Question-Answer Pairs in Extractive Summarization of Email Conversations*, pages 542–550. Springer Berlin, 2007. → pages 11

[38] G. Murray and S. Renals. Meta comments for summarizing meeting speech. In *Proc. of MLMI 2008, Utrecht, Netherlands*, 2008. → pages 4, 22

[39] A. Nenkova, R. Passonneau, and K. McKeown. The pyramid method: Incorporating human content selection variation in summarization evaluation. *ACM Trans. on Speech and Language Processing (TSLP)*, 2007. → pages 15, 27, 32

[40] D. Radev, E. Hovy, and K. McKeown. Introduction to the Special Issue on Summarization. *Computational Linguistics*, 28(4):399–408, 2002. → pages 12

[41] D. Radev, T. Allison, S. Blair-Goldensohn, J. Blitzer, A. Çelebi, S. Dimitrov, E. Drabek, A. Hakim, W. Lam, D. Liu, J. Otterbacher, H. Qi, H. Saggion, S. Teufel, M. Topper, A. Winkel, and Z. Zhu. MEAD - a platform for multidocument multilingual text summarization. In *Proceedings of LREC 2004*, Lisbon, Portugal, May 2004. → pages 7, 33, 37

[42] O. Rambow, L. Shrestha, J. Chen, and C. Lauridsen. Summarizing email threads. *Proceedings of HLT-NAACL 2004*, 2004. → pages ix, 4, 11, 12, 13, 14, 34, 36, 39, 40, 41, 44, 46, 47, 48, 50, 51, 52

[43] R. Schapire, Y. Freund, P. Bartlett, and W. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651–1686, 1998. → pages 10

[44] J. Shetty and J. Adibi. Discovering important nodes through graph entropy the case of Enron email database. *Proceedings of the 3rd international workshop on Link discovery*, pages 74–81, 2005. → pages 18

[45] L. Shrestha and K. McKeown. Detection of question-answer pairs in email conversations. pages 889–895, August 2004. → pages 11

[46] C. Tappert, C. Suen, and T. Wakahara. The state of the art in on-line handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(8):787–808, 1990. → pages 8

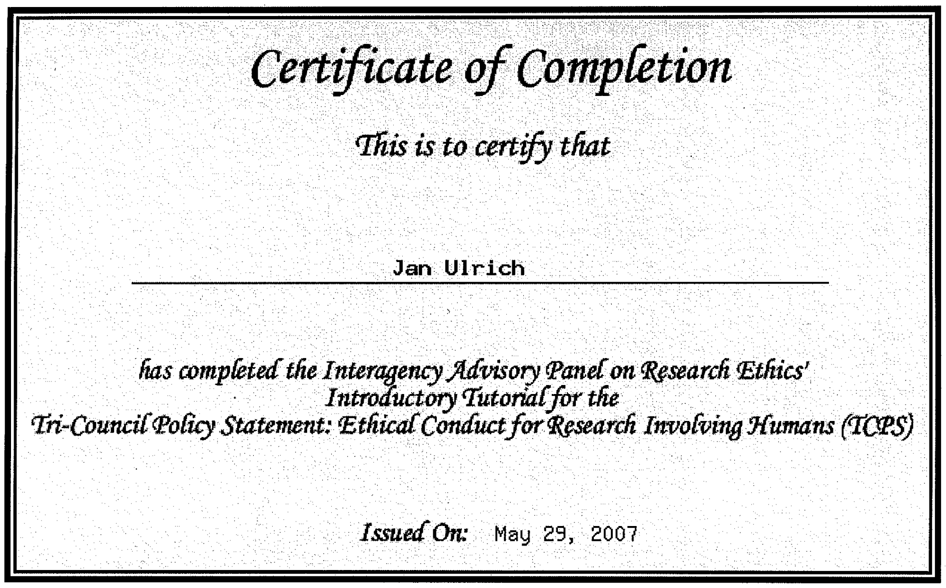
[47] S. Wan and K. McKeown. Generating overview summaries of ongoing email thread discussions. *Proceedings of COLING 2004*, 2004. → pages 10

[48] I. Witten, E. Frank, L. Trigg, M. Hall, G. Holmes, and S. Cunningham. Weka: Practical machine learning tools and techniques with java implementations. *ICONIP/ANZIIS/ANNES*, 1999. → pages 9, 33, 37

[49] D. Zajic, B. Dorr, and J. Lin. Single-document and multi-document summarization techniques for email threads using sentence compression. *Information Precessing and Management*, pages 1600–1610, 2008. → pages 11

Appendix A

Ethical Conduct for Research Involving Humans Certification



Certificate of Completion

This is to certify that

Jan Ulrich

has completed the Interagency Advisory Panel on Research Ethics'
Introductory Tutorial for the
Tri-Council Policy Statement: Ethical Conduct for Research Involving Humans (TCPS)

Issued On: May 29, 2007

Appendix B

Annotator Instructions

General Task Overview:

In this task, your goal is to summarize email threads and to annotate individual sentences for certain phenomena described below.

The Data:

The emails you will be summarizing and annotating are taken from the mailing list of the World Wide Web Consortium (W3C, an organization for developing Web standards). Some of the topics discussed in the email threads are slightly technical, relating to web development, but you are not required to have expertise in these topics in order to complete the task.

Annotation Details:

You will be presented with an email thread consisting of several emails on a particular topic. These emails represent a conversational exchange between members of the W3C. Before carrying out any annotation, read through the thread in its entirety to get a feel for the discussion. Once you have made a preliminary read-through of the email thread, you will proceed with annotation in three steps.

1. In the first step we ask you to write a summary of the entire email thread, using the text box at the top of the browser. You will author a single summary for the thread, not a summary of each individual email. The summary must be no longer than 250 words. It is therefore important to capture the important information of the email thread in a concise manner.

For each sentence in your written summary, we ask you to provide example sentence(s) from the actual email thread that are related to the summary sentence. You can do this by indicating the thread sentence IDs in brackets at the end of each summary sentence. Here we provide an example: ”John proposed that the next meeting should be at the end of May in Miami.[1.1,1.2] Mary disagreed and suggested San Francisco as an alternate location.[2.2,2.3]”

Here the numbers 1.1, 1.2, 2.2, and 2.3 are sentence IDs from the email thread itself. Each sentence in your summary should have at least one such linked sentence at the end of it. There is no limit to how many sentences you may link in this fashion. In exceptional cases we will allow you to provide no links after a summary sentence, but this should be a rare occurrence.

2. You will next have to produce another summary by selecting the important sentences in the original text; that is, the sentences from the email thread that you think are most informative or important. You can do this by clicking on the important sentences to highlight them. You can select as many sentences as you feel are necessary. There may be overlap with the sentences you linked in step 1, while you may also select sentences that were not linked in step 1.

3. In the next step, you will mark individual sentences related to certain speech acts and other phenomena, which we will now define. Speech acts are sentences that perform a particular function. There are 6 types of sentences to annotate.

   (a) **COMMIT(Cmt)**: The first type of speech act consists of sentences where someone is committing to doing something, for example committing to taking on a certain task. In other words, they are assigning themselves an ”action item”. The following sentences would both be labeled ’commit’:

      ”I’ll research that and get back to you.”
      ”I can get you a rough draft by Friday.”

   (b) **REQUEST(Req)**: The second type of speech act consists of sentences where somebody is making a request, for example asking somebody to perform a task. In other words, they are seeking to assign an action item to another individual or multiple individuals. For example,

      ”Sherry, can you send me the submitted version of that paper?”

”Prior to our next meeting, Johan and Sylvia will research the topic and get back to us.”

(c) **PROPOSE(Prop)**: The third type of speech act is when someone is proposing a joint activity. Proposals seek to assign an action item to all members of the group. For example,

”I suggest that we amend our guidelines.”
”We should take this discussion offline.”

(d) **MEETINGS(Meet)**: The fourth sentence type consists of sentences where someone is discussing a meeting. For example, ”Should we schedule the next face-to-face meeting for June?” and ”I know a good meeting venue downtown” are both meeting related sentences. The relevant meeting could be a face-to-face meeting or video conference, i.e. a multi-party synchronous conversation via some medium other than email.

(e) **SUBJECTIVE(Subj)**: Subjective sentences are sentences where the writer or speaker is expressing an opinion or strong sentiment about something. The sentence reveals the so-called ”private state” of the writer. In more technical terms, a subjective sentence is when an experiencer holds an attitude towards a target. The attitude can be positive, negative or neutral. For example,

”The large buttons are a really nice design feature.”
”I'm not sure if that's a good idea.”
”Finding them is really a pain, you know.”

In example 1, the experiencer has a positive attitude about the target, which are the large buttons. Example 2 shows a neutral attitude, and example 3 shows a negative attitude about an unspecified target.

A sentence should only be labeled as subjective if it reveals the private state of the writer and NOT when the writer is reporting someone else's private state, e.g. ”The folks in marketing think the large buttons are a really nice design feature.” is not subjective.

(f) META: Finally, we also ask you to annotate meta sentences (Meta). These are sentences that refer explicitly to this email exchange for example, ”I'm just reading your last email and will respond ASAP” and ”This thread seems to be getting off-topic.”

These 6 phenomena can be annotated simply by clicking the relevant buttons next to each sentence. Note that a given sentence can receive multiple annotations for example, it can be both a meeting proposal and a meta sentence. When a button is clicked, it will change color to indicate your selection.

Quotations:

You will notice that some emails contain both new text and quoted text. Quoted text is text from a preceding email and is often indicated by the '¿' character. For example:

> ”Hi John.
> I disagree with your proposal.
> I think San Francisco would be a more suitable location.
> Best, Mary
> > I propose that we meet in Miami.
> > Does the end of May work for everyone?
> > Regards, John
> ”

Because of the presence of quoted text, some sentences will occur in several different emails in the thread. When linking sentences in your summary, it is not necessary to link every occurrence of a given sentence in the thread. Also, be aware that sometimes new text occurs in the middle of quoted text. For example:

> ”
> > I disagree with your proposal
> Ok
> > I think San Francisco would be a more suitable location.
> Fine by me
> > Best, Mary
> Regards, John
> ”

Recap:

Write your summary of 250 words or less. Create links between your summary sentences and sentences from the email thread. Then create another summary by selecting the most important sentences in the original email text.

Then annotate sentences for the following 6 phenomena:

1. Commit
2. Request
3. Propose

4. Meetings
5. Subjective
6. Meta

We’ll now start with an example thread for you to practice annotating.