Summarizing Online Conversations: A Machine Learning Approach

by

Arpit Sood, mohamed.thanvir@students.iiit.ac.in , Vasudeva Varma

in

24th International Conference on Computational Linguistics - (Coling-2012)

Mumbai, India

Report No: IIIT/TR/2012/-1

Centre for Search and Information Extraction Lab
International Institute of Information Technology
Hyderabad - 500 032, INDIA
December 2012

Summarizing Online Conversations: A Machine Learning Approach

Arpit Sood
SIEL, LTRC
IIIT-Hyderabad, India
arpit.sood@research.iiit.ac.in

Thanvir P Mohamed
SIEL, LTRC
IIIT-Hyderabad, India
mohamed.thanvir@students.iiit.ac.in

Vasudeva Varma
SIEL, LTRC
IIIT-Hyderabad, India
vv@iiit.ac.in

ABSTRACT

Summarization has emerged as an increasingly useful approach to tackle the problem of information overload. Extracting information from online conversations can be of very good commercial and educational value. But majority of this information is present as noisy unstructured text making traditional document summarization techniques difficult to apply. In this paper, we propose a novel approach to address the problem of conversation summarization. We develop an automatic text summarizer which extracts sentences from the conversation to form a summary. Our approach consists of three phases. In the first phase, we prepare the dataset for usage by correcting spellings and segmenting the text. In the second phase, we represent each sentence by a set of predefined features. These features capture the statistical, linguistic and sentimental aspects along with the dialogue structure of the conversation. Finally, in the third phase we use a machine learning algorithm to train the summarizer on the set of feature vectors. Experiments performed on conversations taken from the technical domain show that our system significantly outperforms the baselines on ROUGE F-scores.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*text analysis*

General Terms

Algorithms, Experimentation.

Keywords

summarization, internet relay chats, machine learning, text segmentation, ROUGE.

1. INTRODUCTION

In the recent past the communication of users through social media has seen an exponential increase. A substantial chunk of information exchange happens in the form of online conversations like Internet Relay Chats (IRC), Facebook[1] and Twitter[2] streams. Among them, we focus on conversations from the online support forums which aim to discuss and resolve user-related issues. Such forums contain a lot of information which can benefit organizations as well as information-seeking users. However, these forums suffer from the problem of information overload and redundancy, where similar topics get discussed multiple times by different users. Summarization is a proven effective way to tackle these problems [8, 23]. An effective summary provides the main topics of discussion by removing redundant and unwanted information from the conversation. This saves users' time by providing the essence of the document. These summaries can be used to analyze the impact of virtual social interactions and virtual organizational culture on software/product development [31].

Summarization has been applied to different text genres, such as news articles [5], scientific articles [12] and blogs [27]. But very little work has been done on summarizing conversations. Conversation summarization differs from the traditional document summarization in its informational need and structure. Written conversations in the form of emails and chats have features like acronyms, hyperlinks, nicknames and spelling mistakes [6] which make traditional Natural Language Processing (NLP) techniques difficult to apply. Text in the form of news articles and books is a monologue whereas conversations fall in the genre of correspondence and requires dialogue analysis [20]. Real-time conversations comprise a sequence of exchanges between multiple users that may be synchronous or asynchronous, and may span different modalities [26]. This poses more challenges in analyzing conversations.

To deal with these problems, we propose a supervised machine learning approach to conversation summarization. We employ different types of features to capture the statistical, linguistic and sentimental aspects along with the dialogue structure of the conversations. We evaluate the effectiveness of these features with the aid of different machine learning algorithms on a dataset containing email and chat conversations. More specifically, the contributions of this paper are as follows: (1) to summarize online conversations by investigating new features like sentiment information and dialogue structure of the conversations; (2) to provide a generic fea-

[1] `https://www.facebook.com/`
[2] `https://twitter.com/`

AND'12, Mumbai, India.

ture set which can be applied to both email and chat conversations; (3) to develop a trainable summarizer which can learn using a wide range of machine learning techniques. Our approach has statistically significant performance gain over the baselines on ROUGE F-scores.

The rest of this paper is organized as follows. In section 2 we discuss the related work on conversation summarization. Section 3 discusses our proposed approach. The experimental and evaluation measures are discussed in section 4, followed by results and discussions in section 5. Finally, section 6 concludes this paper and gives a perspective for future work.

2. RELATED WORK

A good amount of research has been done on extractive summarization of documents using supervised learning approach. Edmundson [7] and Luhn [17] have used features such as average term frequency, title words and cue phrases for scoring sentences to create a summary. They used very simple features which can be applied easily and have been considered as the baseline in the area of text summarization, but their approach does not perform comparatively well on social media. Moreover, they have not taken the structure of the text into consideration which can be used to capture sentence importance. Marcu [19] used the structural aspect of the text to discover the relationship between the segments of the text. The relationships were defined by using Rhetorical Structure Theory (RST) [18]. They created a RST tree for the document to be summarized. The nodes of the tree represent the segments from the document, where segments which are more important occupy the upper level of the tree. Summarization is done by performing cuts at different depths of the tree. Although this method makes use of relationships to understand the text in a better manner, the process of constructing tree is costly and the system scalability is an issue.

A significant amount of work has been done in social media, but the majority of it focusses on email summarization. Summarization has been looked as a noun phrase extraction problem by Muresan et al. [21], where noun phrases are taken as the information units, and machine learning techniques are used to learn rules for extraction. Instead of using linguistic information, Rambow et al. [24] have used features based on the representational structure of the emails to perform summarization. Their results show that using email specific features benefit summarization performance significantly. However, the work is very specific to emails and same features are not applicable to other types of text media. Carenini et al. [3] have also considered email summarization as a sentence classification problem but differs from Rambow et al. [24] as they have created an unsupervised system. They have shown that the clue words, subjective words and phrases can be used as an evidence to the importance of the sentences. Ulrich et al. [28] have done regression-based classification compared to the previous work where binary classifier have been used for summarization. They have also shown that usage of speech acts and subjectivity of the sentence as features leads to better performance. However, they have used a manually annotated email dataset, and automating such speech acts to extend it to real world email conversations is a challenging task.

In case of chat summarization, the most relevant work has been done by Zhou and Hovy [31]. They have summarized IRC logs by clustering the discussions and then generating summaries for each of the clusters by extracting adjacent pairs of response by users. Here multiple topics have been considered to be part of the summary. However, they have not taken into account the relative importance of different clusters in a thread. To the best of our knowledge, this is the only concrete work in the area of chat summarization [29].

3. PROPOSED APPROACH

3.1 Preprocessing

Social media text is noisy and unstructured in nature. It requires preprocessing. We describe two modules which constitute the preprocessing phase, namely, Spelling Correction, which operates at word level, and Text Segmentation, which operates at sentence level.

3.1.1 Spelling Correction

To get rid of spelling mistakes we use web as a resource. After removing stopwords from the text, we query the web using Microsoft Bing Search API[3]. For each word we construct a trigram from the text by considering its adjacent left and right context words, and query the web with this trigram. The web results that we get contain related words. Then using Levenshtein distance[4] we pick the three closest candidate words from which we select the one with the highest frequency. We summarize our algorithm for Spelling Correction as:

Algorithm 1 Spelling Correction

```
for each word w in the chat do
  Initialize priorityQueue Q = { }
  bingResponse := QueryBing (w_0 w w_1)
  snippetSet := extractSnippets (bingResponse)
  for each word w' in the snippetSet do
    Q.push(Levenshtein's Distance(w, w'))
  end for
  Initialize closestWordSet CWS := Top3Words(Q)
  Initialize closestWord := MaxFreqWord(CWS)
  if w is not closestWord then
    w = closestWord
  end if
end for
```

where w_0 and w_1 are the left and right adjacent words of the context word w.

3.1.2 Text Segmentation

Since online conversations are not formal in nature, the arguments and points of discussions are not punctuated appropriately which leads to the need for text segmentation. We did text segmentation using two approaches, TextTiling and Longest Contiguous Messages (LCM). In the first approach we used TextTiling [11] algorithm which segments the conversation based on the topic drifts. To assign sentences on per user basis we applied TextTiling on the obtained segments in a hierarchical manner. Whereas in our novel approach of LCM, we concatenate the subsequent messages of

[3] http://msdn.microsoft.com/en-us/library/dd251072.aspx [Online; last accessed May, 2012]

[4] http://en.wikipedia.org/wiki/Levenshtein_distance

Chat Conversation Excerpt	On Applying LCM
jbailey: I've heard that recent kde's are quite nice and fix the usability problems I was having. Hey, I'm setup no gnome, so why try other things? chillywilly: cause kde performs better, imho and chillywilly: that's a big reason why I switched jbailey: *shrug* I've had no problems running gnome on a 500mhz celeron.	jbailey: I've heard that recent kde's are quite nice and fix the usability problems I was having. jbailey: Hey, I'm setup on gnome, so why try other things? chillywilly: cause kde performs better, imho and that's a big reason why I switched. jbailey: *shrug* I've had no problems running gnome on a 500mhz celeron.

Table 1: Effect of using LCM on a Chat Conversation Excerpt

the same user, and when the user changes or when a period is encountered we consider it as a sentence boundary. We have compared both these approaches in Section 5.1 and found out that LCM outperforms TextTiling. Table 1 shows the application of LCM on a chat conversation excerpt.

3.2 Feature Selection

In extractive text summarization, our main objective is to select the most representative and relevant sentences from the text. To deal with the problem of sentence extraction, we represent each sentence by a feature vector. Each feature vector contains a set of different features which capture the basic information content and the dialogue structure of the conversation. We classify the features into two sets: **basic** and **complete**. The **basic** feature set contains features which are not specific to conversations and consider the conversation as a simple piece of text. These features incorporate the statistical and linguistic aspects of the sentence. Features present in the **basic** feature set are :

- **Mean TF-IDF:** Frequency has been used as a feature for various text processing tasks. We use the tf-idf [25] scheme to characterize the frequency of a word. The value of this feature is calculated as the mean of the tf-idf values of all the words present in the sentence. This value is normalized by dividing it with the largest corresponding value among all the sentences in the conversation. Let $f(t,d)$ be the raw frequency of term t in document d, then the normalized term frequency, $tf(t,d)$ is given by:

$$tf(t,d) = \frac{f(t,d)}{max\{f(w,d) : w \in d\}} \quad (1)$$

  If $DF(t,D)$ represents the document frequency for term t in document collection D, then the inverse document frequency, $idf(t,D)$, is given by:

$$idf(t,D) = log\frac{|D|}{1+DF(t,D)} \quad (2)$$

$$tf-idf(t,d,D) = tf(t,d) \times idf(t,D) \quad (3)$$

- **Mean TF-ISF:** This feature also captures the frequency but takes into consideration only one conversation at a time. The frequency of a word is characterized as term frequency × inverse sentence frequency (tf-isf) [15]. This feature is calculated as the mean value of the tf-isf of all the words present in the sentence. This value is normalized by dividing it with the largest corresponding value among all the sentences. It is different from the mean TF-IDF feature as here we try to capture the importance of words based on their frequency among sentences, whereas in mean TF-IDF we capture the importance of words based on their frequency among documents. If $f_s(t,s)$ represents the frequency of term t in sentence s, then the term frequency $tf_s(t,s)$ is given by:

$$tf_s(t,s) = \frac{f_s(t,s)}{max\{f_s(w,s) : w \in s\}} \quad (4)$$

  If $SF(t,d)$ represents the sentence frequency of a term t in document d, and S_d, the number of sentences in document d, then the inverse sentence frequency, $isf(t,d)$, is given by:

$$isf(t,d) = log\frac{S_d}{1+SF(t,d)} \quad (5)$$

$$tf-isf(t,s,d) = tf_s(t,s) \times isf(t,d) \quad (6)$$

- **Sentence Length:** This feature is included to avoid shorter sentences which can be incomplete and have less probability of contributing to the main summary [14]. Longer sentences tend to contain more information. We use the normalized length of the sentence as the feature. It is the ratio of the number of words in the sentence to the number of words in the longest sentence of the conversation.

- **Sentence Position:** Sentence position plays a major role in determining the presence of the sentence in the conversation summary [15]. There can be several ways of defining sentence position: it can be position of the sentence in the user utterance, section or absolute position in the complete conversation. We consider the absolute position of the sentence in the text as the feature. It is normalized by dividing with the number of sentences present in the conversation.

- **Similarity to Title:** The title is a short and precise representation of the primary topic in the conversation. We represent the title and the sentences in vectorial form using tf-isf scheme as given by equation 6. This feature is calculated as the cosine similarity [25] between the title and the sentence vector.

- **Centroid Coherence:** All sentences are represented in vectorial form using the tf-isf scheme(Equation 6). The average of all the sentence vectors represent the centroid. This feature is defined as the cosine similarity between the sentence vector and the centroid vector. Sentences with feature values closer to 1 can

Relation	Definition	Example
Greet	Greetings and pleasantries.	Hi Sir, Respected Sir/Maam, Thanking you, Bests
Background	Setting context for the discussion. Mainly informative.	My Ubuntu's version is 12.04 on 32 bit computer.
Query	Issue being discussed or a request. Main topic of discussion.	I was trying to install a new editor, but i am not able to do it.... can you suggest anything ?
Elaboration	Content and idea behind query, discussed by the participant who issues the query.	When i used apt-get command, it threw some error. The log is very big, and the terminal closes abruptly.
Solution	Replies from the participant regarding the problem. Authoritative, affirmative and strong opinions.	You need to use sudo to install it. This may be because of false interpretation.

Table 2: Brief Description of RST Relations

be considered to better represent the basic ideas of the document [14].

- **Special Terms:** Online conversations like technical IRCs, forums are often flooded with numbers and proper nouns which represent the important facts [2]. We consider the numbers and proper nouns as special terms. This feature is the count of the number of special terms present in the sentence. We normalize this feature by dividing it with the total count of unique special terms present in the conversation. We use Stanford Parser[5] for this purpose.

In addition to the **basic** feature set, we consider more features which capture the dialogue structure and the sentiment information of the conversations. We call this *larger* feature set as **complete** feature set. New features introduced are:

- **Is Question:** The technical forums, IRCs frequently deal with resolving issues and addressing concerns raised by the users [24]. Usually, issues and concerns are raised as questions. This feature is represented by two values, 0 or 1, where 1 represents that the sentence is a question, otherwise not.
- **Sentiment Score:** This feature captures the sentiment present in the sentence. Opinions with strong sentiment carry lot of information and have more chance of contributing to the summary [9]. Unresolved issues are often marked by negative response whereas user satisfaction is generally denoted by positive response. To get the sentiment score of a word we use SentiWordNet[6], where each word is given a positive and a negative score between [0, 1]. Sentiment score for the sentence is obtained by accumulating the scores for all words present in the sentence. We have normalized the scores for this feature by dividing the sentence score by the number of words present in the sentence. Here we focus on correctly detecting the presence of sentiment rather than classifying the sentiment present as positive or negative. Instead of dealing with all words, we have restricted ourselves to verbs, adjectives and adverbs because majority of the left out words will have positive and negative score of 0, and objectivity of 1. We calculate the score of a word w, $S(w)$, as:

$$S(w) = positiveScore(w) - negativeScore(w) \quad (7)$$

- **Discourse Marker:** This feature defines the purpose of existence of a sentence in text. According to Rhetorical Structure Theory (RST), each sentence in a discourse has some meaning with respect to the complete text. RST explains the coherence of the text by assigning relationships between various units (sentences in our case) of the text. RST makes use of two kinds of building blocks to describe text, namely, *Nuclearity* and *Relations*. Mann and Thompson [18] showed that a sentence can be decomposed into segments, usually clauses, and then the relationship between these segments is established using the concept of Nuclearity. If a sentence has two clauses, then the main segment is called the nucleus, and the subordinate segment is called as satellite. Chuang and Yang [4] incorporated this information as a feature for summarization, considering the nucleus as more important than satellite. However, we do not consider the nuclearity aspect of RST. In our case the text belongs to technical domain which has different information need. For example, consider a sentence like "We want to use XYZ software, but it is not getting installed using ABC command" which has the first part of the sentence as nucleus followed by a satellite. Here both segments are equally important with respect to the summary. Therefore, we have chosen sentences as the basic units for segmentation instead of clauses.

  To describe our data, which are technical chats and emails, we have used only the Relation aspect of RST. There are many relations proposed in RST to define the text, however we use only 5 relations to define our data. These are *Greet*, *Background*, *Query*, *Elaboration* and *Solution*. These relations are signaled by using cue phrases like *because*, *if*, *since* etc. Using Marcu's discourse-marker-based hypothesizing algorithm [19], we discover these relations at the level of sentences. Since the same sentence can be marked with two different relations, we consider the following preference order: Query > Solution > Elaboration > Background > Greet. Each sentence is marked by one of these 5 relations as Greet, Background, Query, Elab-

[5] `http://nlp.stanford.edu/software/lex-parser.shtml`
[6] `http://sentiwordnet.isti.cnr.it/`

oration or Solution. These relations define the way text is structured in email and chat conversations. Table 2 explains the meaning of these relations in the context of conversations.

3.3 Summarizer Training

After selecting the features for summarization, each sentence is converted into a feature vector. These set of feature vectors are then given as input to learn the trainable summarizer using different machine learning algorithms. Various machine learning algorithms have been proposed in the literature. We chose the Decision Tree, the Naive Bayes classifier, Mutlilayer Perceptron and Support Vector Machines for our experiments.

3.3.1 Decision Trees (C4.5)

From the family of decision tree algorithms, we chose C4.5 algorithm [22] to train our summarizer. A decision tree is created by representing each node as a feature. Each feature is associated with a set of rules which are learned by the algorithm. We used J48 which is an open source Java implementation of the C4.5 algorithm in Weka[7] toolkit.

3.3.2 Naive Bayes Classifier (NB)

Naive Bayes Classifier is one of the most practical methods of bayesian learning. In naive bayes classifier, each instance is described by a conjunction of attribute values and the target function needs to be predicted. In our case, instances are the sentences from the conversation text and the target function is binary, with values 0 or 1. Therefore, the naive bayesian classifier is applied as:

$$P(c \in C | f_1, f_2 \cdots f_n)$$

$$= \frac{P(f_1, f_2 \cdots f_n | c \in C) P(c \in C)}{P(f_1, f_2 \cdots f_n)} \quad (8)$$

where C is the set of target classes (i.e. in the summary, 0, or not in the summary, 1) and $f_1, f_2 \cdots f_n$ are the set of features. While training, it tries to maximize the probability of occurrence of events given any class. We use the Naive Bayes implementation available in Weka toolkit.

3.3.3 Multilayer Perceptron (MLP)

Multilayer perceptron is a feedforward neural network with one or more layers between input and output layers. It uses a supervised learning technique called backpropagation for training the network. We use the MLP implementation from the Weka toolkit.

3.3.4 Support Vector Machine (SVM)

Support Vector Machines are supervised machine learning models which construct an N-dimensional hyperplane to separate the training data (positive and negative examples) into two categories with maximum margin. We use the LibSVM implementation from the Weka toolkit.

4. EVALUATION

4.1 Dataset

In conversation summarization there is no standard dataset available to validate our approach. Most corpora do not have any model summaries associated with them to use as gold standards, which makes evaluation difficult in nature. We are dealing with conversations which are in the form of chats and emails. As guided by the work of others, Zhou and Hovy [31] and Uthus and Aha [29], we are aware of only one publicly available chat corpus with associated summaries, which is the GNUe Traffic archives. For emails, we have used BC3 corpus.

4.1.1 GNUe Archives

GNUe Archives is a collection of summary digests of discussions on GNUe development. Each digest contains human created summaries in the form of a newsletter based on IRC logs. The summaries are both extractive and abstractive in nature. However, most of the summaries are composed by directly quoting the most informative and relevant sentences from the chat logs. These manual summaries form the gold-standard data for evaluation. We have a collection of 325 chat conversations and their corresponding summaries. There are multiple participants in each conversation with various topics being discussed throughout the conversation. This data can be downloaded from here[8].

4.1.2 The BC3 Corpus

The BC3 (British Columbia Conversation Corpus) is a collection of multimodal conversational data developed by Ulrich et al. [28]. The corpus contains 40 email conversations from the World Wide Web Consortium (W3C) mailing list. The conversations are in the form of email threads. It has 3222 sentences and an average of 6 emails per thread. The corpus provides both abstractive and extractive summaries. For extractive summaries, the email threads have been manually annotated at sentence level. For our purpose, we have considered manually created extractive summaries from the corpus as the gold-standard data for evaluation. The BC3 corpus is publicly available for use[9]. This data is free from errors such as spelling mistakes, abbreviations and emoticons, and the email is also segmented into sentences.

4.2 Experimental Setup

To evaluate our approach, we used ROUGE [16] as a metric for summarization performance. ROUGE F-scores were used, as they represent both precision and recall aspects, for different matches: unigram (ROUGE-1), bigram (ROUGE-2) and longest subsequence (ROUGE-L). In our experiments we performed 5-fold cross-validation, where we divided the pre-annotated data into two parts, namely, training set and testing set. In other words, we used 80% of the data for training the summarizer and the remaining 20% for testing, where similar process was repeated 4 more times. While generating summaries we have limited the number of words to that in the model summaries.

We compare our methods with following most widely used baseline systems. (1) MaxLength: Selects the longest sentences of every participant. This is a proven strong baseline for conversation summarization [10]. (2) HAL: A sentence

[7] http://www.cs.waikato.ac.nz/ml/weka/

[8] http://kt.earth.li/kernel-traffic/index.html

[9] http://www.cs.ubc.ca/labs/lci/bc3.html

extraction based system[10], where the sentences are selected based on the words score in the semantic space built using a lexical co-occurrence model [13]. (3) FirstSent: Selects first sentence from the longest messages of each user in the sequence (*Position Hypothesis*). (4) DiaSumm: This system creates summary by extracting inter-connected structure of segments that quoted and responded to each other. (3) and (4) are hard-to-beat baselines and were used by Zhou and Hovy [31] while summarizing IRC logs. We also perform extractive summarization at different compression rates(from 10% to 25%) to see its effect on precision and recall. In addition to this, we evaluate the spelling correction and text segmentation modules. To create dataset for this evaluation, we randomly selected 88 chats, where spelling mistakes were manually corrected and the chat was segmented wherever required such that each segment is a user utterance. We compare our Spelling Correction module's performance to a baseline of standard Aspell [1], always picking the first correction suggestion.

5. RESULTS AND DISCUSSIONS

5.1 Analysis of Spelling Corrector and Text Segmenter

As mentioned in Section 3.1.1, our spelling corrector makes use of web. We have used the same evaluation procedure as used by Whitelaw et al. [30]. It focuses on metrics that reflect the quality of end-to-end behavior, and account for the combined effects of flagging and automatic correction. Precisely, there are three states: a word could be unchanged, flagged or corrected to a suggested word. We would like to state that all words which are flagged for spell correction will undergo changes. Following are the possible scenarios:

1. Type 1: A well spelled word is not flagged (Final State: Correct).
2. Type 2: A well spelled word is wrongly corrected (Final State: Incorrect).
3. Type 3: A misspelled word is not flagged or wrongly corrected (Final State: Incorrect).
4. Type 4: A misspelled word is corrected (Final State: Correct).

Spell Checker	Type 1	Type 2	Type 3	Type 4
Aspell	240690	4521	503	686
WebSpellCorrect	244789	422	230	959

Table 3: Performance Analysis of Spelling Correction Module for 88 Chat Conversations

In Type 3, we have combined the scenario when a misspelled word is not flagged or gets flagged but wrongly corrected, because both the situations finally lead to an incorrect word. We compare our algorithm for spelling correction with the baseline on all the above mentioned four categories. Table 3 shows that our approach (WebSpellCorrect) outperforms the baseline Aspell by a significant amount. We can see that the baseline introduces lot of Type 2 errors, which is more than the number of misspelled words corrected. On further investigation, we noticed that the technical terms are wrongly flagged by Aspell because of its limited vocabulary in case of such domains. It adds more noise to the documents in the form of misspelled words than initially present. On misspelled words, our system WebSpellCorrect achieves an accuracy of 80.65% compared to Aspell, which has an accuracy of 57.69%.

Text Segmenter	FP	FN
LCM	112	875
TextTiling	832	1657

Table 4: Performance of Text Segmentation Modules on False Positives and False Negatives

Text segmenter is very relevant to summary generation as we treat it as a sentence extraction problem. For text segmentation, we evaluated both the approaches described in Section 3.1.2 for false positives (FP) and false negatives (FN) and found that our method LCM outperformed TextTiling. This is attributed to the fact that in the case of TextTiling algorithm, segmentation of text into discourse units requires subtopic structure, whereas the chats are spontaneous and unstructured in nature. Therefore, we have used LCM over TextTiling. Table 4 compares LCM with TextTiling on 88 chat conversations having 12514 segments.

5.2 Comparison of the Feature Sets:

We experimented with two feature sets, namely, **basic** and **complete**, as mentioned in Section 3.2. The **basic** feature set contains the standard features that can be used with other kinds of text genre as well. Some of these features have been used in the past by researchers for different information retrieval tasks. The **complete** feature set contains additional features which are specific to technical chats and email conversations. This bigger feature set gives more detailed information about the importance of each sentence with respect to the summary. Table 5 shows that using **complete** feature set gives better results compared to **basic** feature set. This is because the new features added help us to better capture the important characteristics of the conversation such as dialogue structure and sentiment information. We have shown the results for both the feature sets on different corpora for Naive Bayes classifier. We got similar improvements when **complete** feature set was used over **basic** with other classifiers.

5.3 Effect of Compression Ratio:

As we increase the compression ratio, the ROUGE recall scores increase for all algorithms, since with larger number of sentences in the summary the probability of the right sentences getting selected increases. The ROUGE precision scores decrease with increase in compression ratio but not in a monotonous manner. Figures 1 and 2 plot the ROUGE-1 Recall and Precision scores against compression ratio for different training algorithms. Similar patterns are observed for other ROUGE scores such as ROUGE-2 and ROUGE-L, and therefore they are not shown here. All the methods improve significantly over the baseline approach[11]. There

[10]Secured 1st position in Document Summarization task in DUC 2007

[11]Only FirstSent and DiaSumm are shown, as they performed better than HAL and MaxLength for BC3 corpus

Configuration	GNUe Archives			BC3 Corpus	
	basic	**complete**		**basic**	**complete**
ROUGE-1	0.42370	0.57179		0.50901	0.58857
ROUGE-2	0.20375	0.36512		0.27477	0.45714
ROUGE-L	0.41024	0.56608		0.49549	0.57191

Table 5: Feature Set Comparison for Naive Bayes Classifier on 325 Chat Conversations and 40 Email Threads on ROUGE-F scores

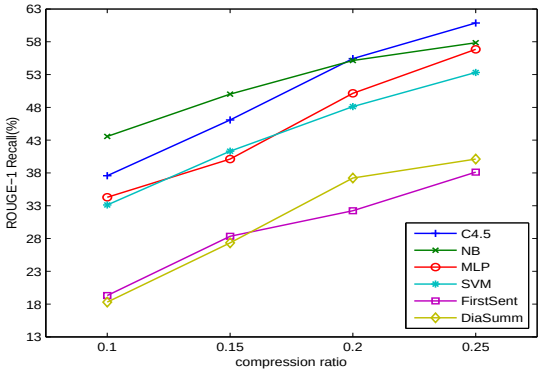

Figure 1: Recall vs Compression Ratio

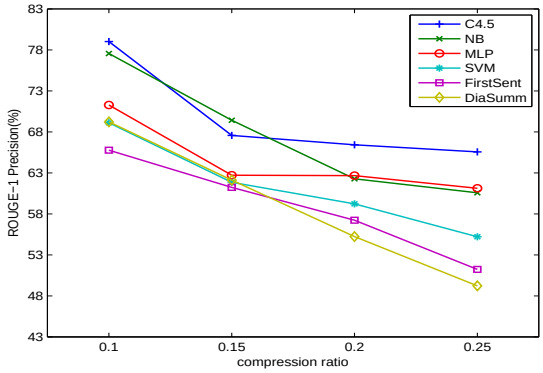

Figure 2: Precision vs Compression Ratio

is relatively less improvement using a higher compression ratio(20% to 25%), compared to a lower one.

Configuration	ROUGE-1	ROUGE-2	ROUGE-L
MaxLength	0.44199	0.27713	0.42209
HAL	0.38874	0.20221	0.37989
FirstSent	0.27651	0.14916	0.26072
DiaSumm	0.42123	0.24495	0.41028
C4.5	0.48192	0.30132	0.45521
NB	**0.57179**	**0.36512**	**0.56608**
MLP	0.55894	0.36318	0.53019
SVM	0.51341	0.30765	0.50282

Table 6: Average ROUGE F-Scores for Chat Conversations

5.4 Effect of Learning Algorithms:

The learning algorithm used strongly influences the quality of results obtained. As mentioned in Section 3.3, we experimented with four different learning algorithms. In the case of GNUe archives, the best results were obtained by using **Naive bayes** classifier. MLP learning algorithm performed almost as well. There was slight decrease in the performance for decision tree and SVM algorithms, but they were still able to outperform the baselines on the measures of ROUGE F-scores. Whereas in the case of BC3 corpus, **C4.5 decision tree** performed the best. Naive Bayes and MLP algorithms also performed relatively good.

All the machine learning algorithms we considered successfully completed the task of generating a reasonable summary, as all of them performed better than the baselines. We have also shown that our feature set is generic enough to learn from any of these algorithms. Table 6 and 7 shows the ROUGE-F scores for various classifiers on chat conversations and BC3 email conversations. All the algorithms performed better than the baselines indicating that the **complete** feature set is a correct way to represent the sentences. All the results presented are statistically significant at 99% significance level. We used paired t-test for testing statistical significance. In our summarizer we have explored a large variety of features, where some of them have been previously applied to different type of information retrieval tasks. We have done a fine-grained analysis of the text and have employed features at word level, sentence level and discourse level to best represent the information present in the sentence.

Configuration	ROUGE-1	ROUGE-2	ROUGE-L
MaxLength	0.27755	0.14909	0.26856
HAL	0.37527	0.24820	0.36104
FirstSent	0.41933	0.26926	0.40487
DiaSumm	0.43815	0.30736	0.41986
C4.5	**0.63724**	**0.52285**	**0.63005**
NB	0.58857	0.45714	0.57191
MLP	0.56292	0.43143	0.55823
SVM	0.48118	0.34172	0.46122

Table 7: Average ROUGE F-Scores for BC3 Corpus

6. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a supervised machine learning approach to summarize online conversations. In our approach, we used a set of different features to incorporate statistical, linguistic and sentimental aspects along with the dialogue structure of the online conversations. We showed that newly introduced features significantly improved the summarization performance. Furthermore, we experimented with various trainable machine learning algorithms and saw that our feature set is generic enough to learn from any of

the algorithms. We successfully summarized real-time chat and email conversations. In future, we would like to extend this work to analyze blogs and twitter streams. We would also like to investigate the language independence of our methods.

7. REFERENCES

[1] K. Atkinson. Gnu aspell. `http://aspell.net/`, 2011.

[2] J. Bengel, S. Gauch, E. Mittur, and R. Vijayaraghavan. Chattrack: Chat room topic detection using classification. In *Intelligence and Security Informatics*, pages 266–277, 2004.

[3] G. Carenini, R. T. Ng, and X. Zhou. Summarizing emails with conversational cohesion and subjectivity. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 353–361, 2008.

[4] W. T. Chuang and J. Yang. Extracting sentence segments for text summarization: a machine learning approach. In *Research and Development in Information Retrieval*, pages 152–159, 2000.

[5] G. Demartini, M. M. S. Missen, R. Blanco, and H. Zaragoza. Entity summarization of news articles. In *Research and Development in Information Retrieval*, pages 795–796, 2010.

[6] H. Dong, S. C. Hui, and Y. He. Structural analysis of chat messages for topic detection. *Online Information Review*, pages 496–516, 2006.

[7] H. Edmundson. New methods in automatic extracting. *Journal of the Association for Computing Machinery*, 16(2):264–285, 1969.

[8] N. Elhadad and K. McKeown. Towards generating patient specific summaries of medical articles. *Proceedings of the NAACL 2001 Workshop on Automatic Summarization*, pages 31–39, 2001.

[9] M. A. Fattah and F. Ren. Automatic text summarization. *International Journal of Electrical and Computer Engineering*, pages 25–28, 2008.

[10] D. Gillick, K. Riedhammer, B. Favre, and D. Z. Hakkani-Tur. A global optimization framework for meeting summarization. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 4769–4772, 2009.

[11] M. A. Hearst. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23:33–64, 1997.

[12] M. Hu, A. Sun, and E.-P. Lim. Comments-oriented blog summarization by sentence extraction. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 901–904, 2007.

[13] J. Jagadeesh, P. Pingali, and V. Varma. A relevance-based language modeling approach to duc 2005. In *Document Understanding Conference*, 2005.

[14] J. Kupiec, J. O. Pedersen, and F. Chen. A trainable document summarizer. In *Research and Development in Information Retrieval*, pages 68–73, 1995.

[15] J. Larocca, N. Alexandre, D. Santos, C. A. A. Kaestner, A. A. Freitas, and C. do Parana. Document clustering and text summarization. *Proc. of 4th Int. Conf. Practical Applications of Knowledge Discovery and Data Mining*, pages 41–55, 2000.

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

[17] H. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165, 1958.

[18] W. C. Mann and S. A. Thompson. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281, 1988.

[19] D. Marcu. *The Rhetorical Parsing, Summarization, and Generation of Natural Language Texts*. PhD thesis, Department of Computer Science, University of Toronto, Toronto, Canada, 1997.

[20] J. D. Moore and F. Keller. Priming of syntactic rules in task-oriented dialogue and spontaneous conversation. *Cognitive Science*, pages 685–690, 2006.

[21] S. Muresan, E. Tzoukermann, and J. L. Klavans. Combining linguistic and machine learning techniques for email summarization. In *Proceedings of the 2001 Workshop on CoNLL*, volume 7, pages 19:1–19:8, 2001.

[22] J. Quinlan. *C4.5: programs for machine learning*, volume 1. Morgan kaufmann, 1993.

[23] D. R. Radev and K. R. McKeown. Generating natural language summaries from multiple on-line sources. *Comput. Linguist.*, 24(3):470–500, 1998.

[24] O. Rambow, L. Shrestha, J. Chen, and C. Lauridsen. Summarizing email threads. In *Proceedings of HLT-NAACL 2004: Short Papers*, pages 105–108, 2004.

[25] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24:513–523, 1988.

[26] O. Sandu. *Domain Adaptation for Summarizing Conversations*. PhD thesis, Department of Computer Science, The University Of British Columbia, Vancouver, Canada, 2011.

[27] S. Teufel and M. Moens. Summarizing scientific articles: Experiments with relevance and rhetorical status. *Computational Linguistics*, 28:409–445, 2002.

[28] J. Ulrich, G. Murray, and G. Carenini. A publicly available annotated corpus for supervised email summarization. In *AAAI08 EMAIL Workshop*, Chicago, USA, 2008. AAAI.

[29] D. C. Uthus and D. W. Aha. Plans toward automated chat summarization. In *Meeting of the Association for Computational Linguistics*, pages 1–7, 2011.

[30] C. Whitelaw, B. Hutchinson, G. Chung, and G. Ellis. Using the web for language independent spellchecking and autocorrection. In *Empirical Methods in Natural Language Processing*, pages 890–899, 2009.

[31] L. Zhou and E. H. Hovy. Digesting virtual geek culture: The summarization of technical internet relay chats. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 298–305, 2005.