ABSTRACT

KAKARADDI, SHISHIR MAHESH. A Comparison of Summarization Techniques for Small Sets of Micro Blogs. (Under the direction of Dr. Robert St. Amant.)

Microblogs are a popular way to keep abreast with important information about different topics. Half a billion people across the world use Twitter, and and 175 million tweets are posted each day. With so much available data, summarization is critical. This thesis examines the problem of summarizing tweets from a single Twitter user over the the course of a few weeks to a month, between 35 and 60 tweets. The summarization takes the form of a subset of those tweets. We have developed and compared algorithms that can be used to summarize tweets in this way. We describe an unsupervised algorithm that clusters tweets by topic, using information retrieval techniques, and then selects a specific tweet from each cluster, relying on sentiment. An experimental evaluation shows that this approach produces results that are not as good as those of human summarizers, but are better than an existing baseline for such summaries.

A Comparison of Summarization Techniques for Small Sets of Micro Blogs

by
Shishir Mahesh Kakaraddi

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Computer Science

Raleigh, North Carolina

2012

APPROVED BY:

Dr. Edward F. Gehringer

Dr. Christopher G. Healey

Dr. Robert St. Amant
Chair of Advisory Committee

DEDICATION

To my family, for their support, love and encouragement

To my advisor, Dr Robert St. Amant, for his guidance in research

To all of my friends, for our friendships

BIOGRAPHY

Shishir Mahesh Kakaraddi was born on 21st July, 1987 in Belgaum, Karnataka, India. He received a Bachelor of Engineering degree in Information Science from M S Ramaiah Institute of Technology, Bangalore, India. He worked at IBM India Pvt Ltd as Software Developer for11 months and then he joined North Carolina State University to pursue his Masters degree in the Department of Computer Science. After completion of his Masters degree he will be joining VMware in Palo Alto, California as a Member of Technical Staff.

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Robert St. Amant who provided his professional advice on research ideas and dissertation writing. I also want to thank other committee members, Dr.Edward Gehringer and Dr. Christopher Healey, for spending their time reading my and contributing valuable comments and suggestions.Thanks to Dr. Tessa Lau, Research Staff Member, IBM Research for motivating me to do research and for introducing me to my advisor. I want to thank my friends Sridevi Venugopal and Thothathri Srinivasan for providing their opinions on several intermediate results. I would like to thank Sina Bahram, Srinath Ravindran and others in the Knowledge Discovery Lab for their suggestions. I would also like to thank everybody who helped me with the evaluation of my research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Traditionally, news and information about events would be available from few sources like television news channels and radios. With the upsurge of use of social networks over this decade, data is available on a real-time basis in humungous amounts on platforms like Twitter, Facebook and so on. To make true the statement that information is wealth, this overwhelming amounts of data has to be processed and converted into useful, precise information that provides important insights into different issues.

1.1 Twitter

Twitter is a social network where a user can post details or comments about a topic of his choice in 140 characters. Each such post is called a tweet. These tweets also include URLs to different articles, hashtags that act as meta data about the tweet itself. These tweets as micro-blogs where users can present any content in a precise manner. Popular tweets are tweeted again by other users and are called as retweets.Twitter has half a billion user accounts and the users send 175 million tweets every day. It is being used massively to post content on the web by a range of users ranging from politicians to pop stars to voice their opinions, support different causes or to just post whats on their mind. Twitter has become the strong social medium where users react to catastrophes like the Haiti earthquake or post about a power cut or express their constant emotions during a football match. A search on a popular event generates tens of hundreds of results per second.

On the other side of the sphere, the incredible amounts of data generated by Twitter act as the source of data for many researchers these days. Twitter provides easy-to-use RESTful APIs to obtain tweets of any public user or to search for tweets and researchers are working on this data to generate useful information from them. They have been used in varied contexts to provide summarized information. Nichols et al have worked on generating a textual summary of soccer using the tweets from Twitter. Chakrabarti et al have presented the set of highlights or important sub events as a summary of structured, time-constrained events. Sakaki et al detect occurrence and time of earthquakes or trajectory of typhoons from set of user tweets as summary.However in the context of summarizing tweets of a single user, we feel that a small set of tweets that is representative of all the topics and events that a user tweets about can be considered as a summary. This provides an idea of the users interests and an insight into the kind of content the user usually tweets on Twitter.

Chapter 2

Related Work

There has been considerable amount of work in the area of micro blog minning. The different techniques and results are discussed in this chapter.

2.1 Twitter Classification

Twitter classification is used as the first step in many different twitter mining techniques. It is used as an intelligent data cleaning method to remove the unnecessary tweets from the initial twitter dataset. Sakaki et al.(2010) use classification as an intelligent filter in their technique of detecting the occurrence, time and location of earthquakes and typhoons by mining tweets. They provide a multi step process to obtain this information. Their first step is to classify the tweets into two classes -tweets pertaining to real earthquakes and tweets not pertaining to real earthquakes. They use a Support Vector Machine based classifier with different types of features like,

A. Statistical Features (Number of words, Position of query word)

B. Keyword Features (Word vector of the Tweet)

C. Word Context Features (Words before and after the query word)

Their classifier does really well at detecting earthquake events when they use only statistical features on the features. It has a F-value of 73.69%. Its F-value is lower when they used other features 53.85% and 57.14% for B and C respectively. This is further emphasized by the work of Sriram et al. (2010). They describe a technique for organizing tweets into different top level classes using a Naive Bayes classifier. The goal of their work is to classify sets of tweets into generic classes, such as News, Opinions, Deals, Events, or Private Messages. The best performing classifier relies on eight features, one nominal and the remaining binary, collectively called 8F:

1. author (nominal)
2. resence of shortening of words and slang terms,
3. time/event phrases,
4. opinion words,
5. emphasis words,
6. currency and percentage signs,
7. @username at the beginning of the tweet, and
8. @username within the tweet.

The overall accuracy of 8F is around 95%. Because the model uses author information, however, it will tend to be tuned to each user's tweeting habits. This is the main drawback of this approach; as a trained model cannot be easily applied to tweets from new users. The authors write, "The author feature is found to be very discriminative in our dataset."

They also test their work against a Bag-of-Words (BOW) feature set, and find that 8F improves performance by 32.1% over BOW. So even they found that statistically computed features seem to perform significantly better than Bag of Words. Performance of the combined 7F+BOW (7F is 8F with the author feature removed) falls between 8F and BOW.

2.2 Twitter Summarization

A substantial amount of work has been done in exploring different techniques to summarize tweets. The word summary has been used as a common term for any information that is concise and extracted from a larger body of tweets. It takes different structures based on the context. The techniques of extracting summary too differ significantly based on the context and application. Chakrabarti et al.(2011) have described different approaches to summarize an american football game. They aim to recognise all the sub events in a game like touchdowns, field goals etc using the tweets pertaining to the game. They have evaluated three different algorithms for this task. The first algorithm is called SUMMALLTEXT based on calculating euclidean distance between word vectors of every pair of tweets and choosing the one with the least distance from all other tweets and highest distance from already chosen tweets. This process is repeated for a specified number of times. This approach has many drawbacks, It gives too much importance to the one major subevent, It cannot recognise two separate events of the same type like two touchdowns at different times. They take this as the baseline for evaluating their other algorithms. To differentiate the events better temporally they use an important parameter in summarizing a football game. They divide the tweets into different time slots and apply the SUMMALLTEXT. This results in temporally segregated set of sub events. This is called

SUMMTIMEINT. It suffers mostly with the same problems as SUMMALLTEXT. Finally they describe a variant of Hidden Markov Model(HMM) which recognises the changes in the language model over time better than the SUMMALLTEXT and SUMMTIMEINT. They call it SUMMHMM. They train the model for american football game before running the evaluation. The models generated are very specific to the type of event hence the model needs to be trained for every new type of event. The model mainly depends on bursts in tweet volume and changes in language model over time. This makes the model usable only for events which are like sports, where there are defined important moments which correlate to bursts in tweet volume. The evaluation shows that on an average recall is just 0.5 when top 30 tweets are extracted after the SUMMHMM computation. This shows that recognizing sub events is really a hard problem even when you consider trained models.

Jeffrey et al.(2012) try to address the shortcomings of a trained model to recognize subevents and summarize them. To demonstrate their technique for summarizing sporting events they have used a running example of soccer. Apart from recognizing the subevents they aim to provide a set of sentences as summary of a soccer game. They use the burst in tweet volume information to detect events like a goal, yellow card and penalty shoot etc. Each burst in tweet is considered a sub event they detect the exact burst of tweets and the collect all the tweets from start of the burst to the end of the tweet burst. This is considered as a cluster. They remove the noise and choose the top n sentences from each cluster ranking them based on the phrase graph approach discussed in Sharifi et al.(2010). This model is unsupervised unlike other approaches used for summarizing tweets of an event. However this model relies on the fact that each burst of tweets is one event. This may not hold true if two important events occurred in quick succession. Quick succession of events in common in some sports like american football. Chakrabarti et al.(2011) have reported that in their dataset of american football games 45% of the interception-plays were followed by field-goals or touchdown-plays. They also cite this as a reason for some performance drop in there SUMMHMM approach. Jeffrey et al.(2012) have evaluated their model only on a soccer game which has relatively less number of events and generally the events for a soccer game are well spaced out in time compared to other games like american football or basketball. However their results show a recall ranging from 0.6 to 0.9 and a precision of around 0.9 for the soccer games they have evaluated this is promising result considering the difficulty of the problem.

Searching twitter for important real time information is an important application of twitter. Query phrase based summarization techniques have been explored for providing good search results. OConnor et al.(2010) have a built a system to summarize the search results for a given term from twitter. A user can enter a keyword, the output will be a clustered sets of tweets identifying the different subtopics about the keyword. They use a series of steps to identify these subtopics. They can be either unigrams, bigrams or trigrams. In the first step they tokenize and filter the words. In the next step the calculate the importance of each token using probability of occurrence in the result subset and general tweet corpus. Then they merge similar topics using jaccard coefficient of similarity between the n-grams. They remove near duplicate tweets by looking at trigrams. This approach is interesting and good to extract the sub topics from a given topic. However it relies heavily on the tokenizer and probability of occurrence of words. This can lead to non important but frequently used words being chosen as the topics in case of users tweets. It is common for a particular user to use similar words over and over again which do not represent a sub topic. Example words like president, POTUS which stands for President of The United States etc are used extensively in President Barack Obamas twitter feed. Using this algorithm they would get recognized as the main topics of clusters thus resulting in substandard results.

A very well known approach to summarizing tweets based on query word is the technique described by Sharifi et al.(2010). They describe an algorithm called the Phrase Reinforcement(PR) algorithm which can be used summarize a set of tweets. The have described the algorithm as a method to filter search results for a query term on twitter. The PR algorithm begins by creating a phrase graph of all the tweets obtained with the query term as the root node. Then each phrase which occurs in the context of the root node is augmented to the phrase graph. Then the graph is weighted based on a scheme which weighs heavily on repetition of words. Then the sentence with the highest weight is taken out first and so on to generate a partial summary. This technique could be used to summarize cluster of tweets which belong to a related event. However this technique is not suitable for summarizing cluster formed by tweets of a user because, we found that same sets of words are not used adjacent to the topic phrase when we look at tweets of a single user. For example for a topic like health care which is seen as a topic in President Obamas tweets shows that the context is sometimes womens health and sometimes it is related to health insurance.

Chapter 3

Preliminary Development

There are different approaches to summarizing micro blogs. Broadly we can divide the approaches into supervised and unsupervised learning approaches. The different approaches we explored are explained in this chapter. We describe the data collection and processing phases. Then we discuss the supervised classification technique. We try to classify tweets into representative and non representative tweets using generic statistical features. We discuss the result of these explorations below. Then we look at unsupervised techniques like clustering using different algorithms and different features.

3.1 Data Collection

Twitter provides a very easy to access REST API for accessing a users tweets. It also provides REST API client bindings in many different languages. This makes collecting data from twitter easier compared to other micro blog services like Facebook etc. We collected data to conduct a study in order to understand more about what people think are the representative tweets among a given set of tweets from a user. We built a website where users could enter twitter username of a person they follow. They will be shown the last 30 to 50 tweets of that person. Then the users can go through these tweets and choose what they think are the representative tweets. The website was a Ruby on Rails application which used Redis key value store as the data store. The data collection application could make calls to twitter API and get the latest tweets by a particular twitter user. We did this study on a total of 13 users. This initial study helps us understand how users choose summaries from a given set of tweets. When we put together the numbers in all 541 tweets were seen amongst the 13 users. They have chosen a total of 63 tweets from these as representative tweets.

3.2 Data Preparation

The tweets collected in the data collection phase have to be pre-processed and cleaned before they can be utilized to compute features. In this phase, the tweets are pre-processed using the two techniques of Stemming and Stopword Removal. Stemming is a process where different forms of the word are reduced to its original root form. In our study, a standard porter stemmer has been used on all the words of the tweets for stemming them . They are then converted to lowercase words. Stopword Removal is a technique where commonly used english words which donot distinguish the sentence from others are removed. In this study, a stop word list in English is used to remove commonly used words in tweets like prepositions, conjunction. This process ensures that all our tweets-data now have only important words eliminating the superfluous words that are seen in all tweets but act as noisy data in our study. The stop word list used in our study is provided as an attachment in the appendix section of this document. The tweet data is now pre-processed and cleaned of noisy and irrelevant words and is ready to be processed.

3.3 Data Processing

We use the dataset of 541 tweets as the dataset to conduct our exploration. We have seen that statistical features have been really good at classifying tweets. Using only statistical features would make the model very generic. This will allow us to use it on different datasets. We calculated the different statistical features which can be used to classify tweets as representative tweets and non representative tweets. These statistical features include Inverse Document Frequency(IDF) for each word of each tweet. It is a commonly used statistical metric in information retrieval. It is helpful in indicating whether a word is unique to a small set of documents or it is common across all documents.We also compute Spelling correctness for each word of each tweet. Then we use aggregations of these values as the features for the dataset.

1. The set of features we chose for classifying tweets include
2. Number of Words in the tweet
3. Average length of words in a tweet
4. Maximum Inverse Document Frequency (IDF)
5. Position of the word with the Maximum IDF
6. Average IDF of the tweets
7. Sum of IDF of all the words in the tweet
8. Spelling correctness percentage of a tweet
9. Number of URLs in the tweet
10. Number of Hashtags in the tweet

Table 3.1: Classification Performance

Algorithm	Precision	Recall	F-Measure
Support Vector Machine	1	0.036	0.07
Naive Bayes	0.25	0.055	0.09

3.4 Classification

We created a comma separated values(CSV) file with all the tweets and the computed features. We did not any features like Username or Bag of words etc which would make the model very specific to the dataset under consideration. We used Weka to run different classification algorithms and noted down the F-Measure. The results are tabulated in table 3.1

The dataset had around 10% of the tweets classified as representative and the rest as not representative. This created an imbalance in the dataset many other algorithms like Bayes Net and J48 decision trees had a F-Measure of 0 for Representative tweets. We tried to balance the dataset by sampling only 10% of the tweets from the non representative set of tweets. This too didn't change the F-Measure much.

3.5 Clustering

We could not achieve good performance using supervised learning techniques. Unsupervised techniques are very flexible. For an unsupervised technique we can use the words and properties of the words in a tweet as parameters, we tagged each word of each tweet with a parts of speech tag. We also assigned a real number indicating the sentiment value of each word.

Our first approach was to use bag of words as the feature set for clustering. We applied Expectation Maximization(EM) and XMeans algorithm on the feature set. Euclidean distance was used as the distance metric between two tweets. We found that clustering was broad and few clusters were created using these algorithms. Unrelated tweets also were clustered together. In order to solve these problems we realized we cannot use all the words in a tweet as features. We applied a filter on the features only nouns and verbs were chosen to be part of the word vector. This helped is reducing the unrelated tweets being clustered together. However only few clusters were generated. Since the number of clusters generated in both XMeans and EM depends on maximum log likelihood estimation. The log likelihood of more clusters would not be maximum but still only few clusters were not enough. Examples of summaries formed using XMeans and EM clusters are provided in the Appendix. For example when we applied these algorithms on 50 tweets of President Barack Obama we got only 3 clusters.

Chapter 4

System Development

The process of extracting summary tweets from a small set of tweets involves three steps.

1. Access the micro blogs and compute different features of the data.
2. Group the tweets into clusters.
3. Choose a summary tweet from each cluster

In this chapter we discuss the approaches we have taken to complete all the three steps.

4.1 Computing Features

In this step all tweets are converted to word vectors of a fixed size. First we convert each tweet to lower case. Then stemming and stop word removal is done. This new form of the tweet is stored as a processed tweet. The processed tweet is used to compute the Inverse Document Frequency (IDF) of each word in the tweet. Each word of the tweet is also tagged with a parts of speech (POS) tagger. At the end of this step we have the processed form of a tweet, we have the IDF values and the POS tags for each word. The next step is to filter the words from the processed tweet, we filter out all the words which are not tagged as nouns or verbs. This helps in making sure only important words are used as candidates for topics. Then we sort them based on IDF. Lower IDF means the word has been used many times in set of tweets hence is a good candidate for clustering. The top n words from the sorted array are considered as the feature vector. The number n is specified by the algorithm.

4.2 Clustering Algorithm

Each tweet is a feature vector also known as a point for the purposes of explaining the algorithm. The set of feature vectors is an input to the clustering algorithm. The clustering algorithm is a modified version of KMeans clustering algorithm. The KMeans clustering algorithm takes two inputs, number of clusters to be formed and the input points. It returns the clusters. We use KMeans as a part of modified KMeans. In the first iteration of the algorithm input set of feature vectors is split into two clusters. If clusters obtained have a size less than a threshold value then they are added to the final clusters list. If the size of the cluster is bigger than the threshold value then they are added back to the pool of points to be clustered. This way the threshold number for size of a cluster to be accepted as a final cluster acts as a way to control granularity of the cluster. If i clusters are output in a certain iteration and all have a size greater than the threshold value then in the next iteration we try to split the points into i+1 clusters. This breaks a deadlock in the algorithm and keeps the iterations going till we split the points into granular clusters. An other way to break deadlock when too many words are similar between tweets is to add more detail by increasing the number of words considered for clustering in the feature vector.

Modified Kmeans (MKmeans) has many advantages. The most important advantage in the context of summarizing tweets is that it gives the user a easy way to understand and control the granularity of the summary returned. The threshold size for a cluster can be controlled by the user as it easy to understand its function. If the threshold size of a cluster is inversely proportional to the number of tweets in the summary. This is because if the threshold is higher then there will be less clusters output by the algorithm there by reducing the number of clusters in the summary.

Listing 4.1: Distance calculation between tweets

```
def distance_measure(fv1,fv2,params_length)
  fv1 = fv1[0..params_length-1]
  fv2 = fv2[0..params_length-1]
  temp = Hash.new
  for word in fv1
    temp[word] = true
  end
  matches = 0
  for word in fv2
    if temp[word] == true
      matches += 1
    end
  end
  return params_length - matches
end
```

The KMeans algorithm used in the Listing 4.2 is not randomly initialized but chooses equally spaced points as its centers for the first iteration. This helps in getting consistent results. The KMeans algorithm uses a custom defined distance measure in our implementation. The distance between two feature vectors is calculated in a simple way. The distance function takes three inputs. The two feature vectors and parameter length. The third argument is used to decide how many words from the feature vector are considered for calculation of the distance. The output is the distance between the two feature vectors. This type of distance function ensures we can find distance between two feature vectors which have only nominal values easily.

Listing 4.2: Modified KMeans Algorithm in Ruby

```
def MKmeans(all_points ,first_split ,params_length ,threshold_cluster_size)
    i = first_split #Number of Clusters to split into
    while !all_points.empty? #While All Points is not Empty
      clusters = KMeans(all_points ,i ,params_length)
      flag = 0
      all_points = Array.new
      for cluster in clusters
        if cluster.size <= threshold_cluster_size
          final_clusters << cluster
          flag = 1
        else
          all_points += cluster.points
        end
      end
      if flag == 0
        i += 1
      else
        params_length += 1
      end
    end
end
```

4.3 Cluster Summary

Each cluster from the clustering process represents a topic. It is meaningful to choose one tweet from each cluster as the summary tweet. Collectively they will be representing the different topics from the input. We explored a few ways to choose one sentence from each cluster. We tried three approaches for this, The first approach was to choose the cluster center as the most representative tweet. This ensured that the words that describe the topic are included in the tweet. When we tried this out on few users we saw that clusters centers often tend to be tweets with small number of words. They dint provide much information apart from the topic itself. To solve this problem we chose to pick the tweet with the highest number of words. This was our second approach. We compared the results with users choices we saw that users often dint choose them as summary tweets. This led us to the final approach based on sentiment. We calculated sentiment of a tweet as the sum of absolute value of sentiment of each word in the tweet. Then we choose the tweet with the highest sentiment in the cluster. This correlated better with user choices of summary.

This completes the system description. To summarize we calculate feature vector for each tweet based on the IDF and sentiment values of each word of the tweet. The feature vectors are clustered using a modified KMeans algorithm. Each cluster represents a granular topic. We choose a tweet from each cluster based on sentiment to form the summary tweets of the micro blog.

Chapter 5

Evaluation

5.1 Experiment Setup

5.2 Experiment Description

We treat the three phases of the experiment as being independent. Phase 1 involves the summarization of #BarackObama tweets, Phase 2 the summarization of #MittRomney tweets, and Phase 3 the summarization of a participant-chosen set of tweets from different Twitter users.

User ratings are discrete-valued data, which means that some common types of summary and inferential statistics used for continuous data are inappropriate. We adopt the conventions of HCI research in handling Likert scale data, using the median rather than the means as a measure of central tendency, and using non-parametric tests for comparisons.

Bar charts of the MKM and SAT summaries for each of the three experiments are shown below. The median of each distribution is shown in Table 5.1.

	MKM/G	SAT/G	MKM/T	SAT/T
Phase 1: #BarackObama	4	3.75	4	3.75
Phase 2: #MittRomney	4	3	4	3
Phase 3: (participant-chosen)	3.5	3	3.5	3

Table 5.1: Medians

5.3 Results

The summaries in Table 5.1 suggest that MKM outperforms SAT in each of the three phases. We can go further by analyzing the differences between the ratings of two summaries of a specific Twitter user, per experiment participant.

We use a Wilcoxon signed-rank test for comparing medians: MKM versus SAT for the General condition, and MKM versus SAT for the Topic condition. The non-parametric Wilcoxon test is designed for continuous data but is applied in practice to discrete ordinal data as well. The hypothesis we are testing is whether there is a significant effect of Algorithm (MKM or SAT) on the medians of the distributions in each condition, specifically whether MKM > SAT.

For Phase 1, the #BarackObama dataset, a Wilcoxon test shows no significant effect of Algorithm in the General condition ($W = 12.0, p = 0.12$), and similarly no significant effect in the General condition ($W = 11.0, p = 0.18$).

For Phase 2, the #MittRomney dataset, a Wilcoxon test shows that there is a significant effect of Algorithm in both the General condition ($W = 27.5, p = 0.009$) and the Topic condition ($W = 25.5, p = 0.01$).

For Phase 3, the dataset including participant-chosen Twitter users, a Wilcoxon test shows that there is a significant effect of Algorithm in both the General condition ($W = 20.0, p = 0.024$) and the Topic condition ($W = 17.0, p = 0.008$).

Alternative tests, for which the user ratings are encoded in categorical form, show comparable results.

For example, we can reduce the rankings into three categories, in which MKM $<$ SAT, MKM $=$ SAT, or MKM $>$ SAT. A χ^2 test can then be used to test the hypothesis that the probability of MKM $<$ SAT is equal to the probability of MKM $>$ SAT. This hypothesis is rejected in all the same cases as above.

Chapter 6

Conclusion