Analysis of Tweet Generation as an Extractive Summarization Problem

First Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
email@domain

Second Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
email@domain

Abstract

Write abstract

1 Introduction

First two paragraphs - order and glue

Summarization techniques developed till now can be broadly classified as extractive or abstractive (Hahn and Mani, 2000). Extractive summarization identifies keywords and phrases from the original text and strings them together to form a summary, while abstractive summarization describes the content of the document in more general terms.

With the rise in popularity of social media, message broadcasting sites have become the new means of communication, voicing opinions, broadcasting news, promotions and so on. Newspapers, channels, movies, government officials, entertainers, have all established themselves on social media and use it regularly for broadcasting news and promoting their products. Twitter is such a public message broadcasting service. Since all posts on the website are a limited length, 140 characters, it has been termed as microblogging. The messages, called tweets, convey precise information about a given topic in a limited length. The nature of this website has made it popular for global discussions on current goings-on in the world, with an estimated 200 million tweets being tweeted per day.

The tweets are often used as a link to a new web page, that has more detailed information about the topic being discussed. This set up suggests that the web page, which might contain videos, images, or articles, blogs, and so on is being promoted with the use of the tweet. In the case of articles, intuitively, the tweet seems to be an indicative, informative, or critical summary of the article being promoted. Hence, it might seem that the problem of tweet generation in this context can be looked at as an extractive summarization problem where the linked article is the source text and the tweet is the generated summary.

Lloret and Palomar (2013) and Lofi and Krestel (2012) both study this same problem of generating tweets using summarization methods. While the former compares various extractive summarization algorithms with Twitter data to generate tweets from documents, the latter suggests a system to generate a tweet using documents from local government records.

To validate modelling of tweet generation as an extractive summarization problem, we extracted such a data set from Twitter, also extracting linked documents through the tweets. We used this data and applied unigram, bigram and LCS (longest common subsequence) matching techniques to show that to generate tweets from related articles we need a more involved approach than blindly using extractive summarization algorithms. We also use stylistic analysis on the articles to explain the results obtained.

2 Background and Related Work

Add papers: * summarization evaluation paper
* add transitions between paragraphs

move to data

Brooke et al. (2012) describe the process of building a formality lexicon by analyzing the stylistics of text. They calculate formality scores for words and sentences by training a model on a large corpus based on the appearance of words in specific documents. Their model represents words as vectors and the formal and informal seeds appear in opposite halves of the graphs, suggesting that we can use these seeds to determine if an article is formal or informal. Brooke and Hirst (2013) used an LDA based model using a similar idea of seed words for getting stylistic rankings for documents. The documents were ranked for styles such as literary, colloquial, subjective, concrete, and so on.

There have also been studies specific to Twitter data, for classifying and summarizing text, intents, etc. Ghosh et al. (2011) classified the retweeting activity of users based on entropy. The study

considered the occurrence of the same URL in a different tweet as a retweet, and was able to separate the tweets as automatic or robotic retweeting, campaigns, news, blogs and so on. The study shows some interesting trends of retweeting activity for each of these cases. In another study, Chen et al. (2012), were able to extract sentiment expressions based on their corpus of tweets, that resulted in extraction of both formal and slang sentiment bearing words.

Mirco-blogging sites, easy access to Internet and the popularity of social media offers an opportunity to analyze data that comprises of statements from a huge number of users. Twitter is such a platform and has gained millions of users by now, and is hugely popular platform now for announcements, voicing opinions, promotions and so on. This data has been used for event summarization studies. O'Connor et al. (2010) uses topic summarizations for a given search for better browsing. Chakrabarti and Punera (2011) generate an event summary by learning the event using a Hidden Markov Model over the tweets describing it. Wang et al. (2014) generate a coherent event summary by treating summarization as an optimization problem for topic cohesion. Inouye and Kalita (2011) compare multiple summarization techniques to generate a summary of multi-post blogs on Twitter.

There has also been an attempt at generating tweets, texts of 140 characters using different text summarization techniques by Lloret and Palomar (2013). Summarization systems were used to summarize texts to sentences and then were compared against each other, evaluated using the ROUGE metric for evaluation. The ROUGE-1, ROUGE-2 and ROUGE-L metrics were used and the tweets were compared against an ideal summary. ROUGE is better when used with multiple reference texts and is not meant to be used at the sentence level. Thus the evaluation is done using the unigram, bigram and longest common subsequence matching techniques used in ROUGE-1, 2 and L.

To the best of our knowledge, only Lofi and Krestel (2012) aim at generating tweets based on data from documents related to the topic. The system proposed uses keyword extraction techniques to generate tweets containing links to the article, hashtags based on the topic from documents and summarized content of the document. The study does not give details of implementation or evaluation of the system. Moreover, after the hashtags and the url, the Twitter constraint of 140 characters leaves room for few words in the generated tweet.

He et al. (2000) study the limits of extractive summarization by comparing user preferences for multiple types of summaries for an audio-visual presentation. They demonstrate that the most preferred method of summarization is highlights and notes provided by the author, rather than transcripts or slides from the presentation. Conroy et al. (2006) have defined an oracle score towards the same aim. The oracle score is based on the maximum likelihood probability of words occuring in model summaries and is in turn used to generate summaries that perform better than any extracted and also human-generated summaries. These studies show that extractive summarization algorithms may not generate good quality summaries even after giving high ROUGE evaluation scores.

3 Data Extraction and Preprocessing

3.1 Using Twitter for Data Extraction

As mentioned earlier, there have been numerous studies that used data from the public Twitter feeds. However, since none of the datasets used in these studies contained tweets and related articles promoted by these tweets separated into categories as required for this study, we extracted data directly from the site.

3.2 Extracting Data

Data was extracted from Twitter using the Twitter REST API using 51 search terms, or hashtags. These hashtags were chosen from a range of topics including pop culture, international summit meetings discussing political issues, lawsuits and trials, social issues and health care issues. All these hashtags were trending (being tweeted about at a high rate) at the time of extraction of the data. To get a broader sample, the data was extracted over the course of 15 days in November, which gave us multiple news stories to choose from for the search terms. A few examples of the search terms are shown in *Table 1* Only English tweets were extracted since the study is limited to English. In the beginning, about 30,000 tweets were extracted, and more than half of these tweets, around 16,000 contained URLs referencing some news articles, photos on photo sharing sites, and videos. The

hashtags were chosen to maximise the number of articles related to the tweets. Hence, a lot of topics that were chosen were being tweeted about by news agencies and other popular news sources.

Can be compressed if required

Politics	Science & Technology
#apec2014	#rosetta
#G20	#lollipop
#oscarpistorius	#mangalayan
Events	Films and Pop culture
#haiyan	#TaylorSwift
#memorialday	#theforceawakens
#ottawashootings	#johnoliver
International	Sports
#berlinwall	#ausvssa
#ebola	#playingitmyway
#erdogan	#nycmarathon

Table 1: Table of Hashtags used for extraction. Table shows some examples of search terms chosen from various different categories.

The data from the tweets was cleaned by removing the tweets that were not in English as well as the ones that were retweeted, which is equivalent to re-publishing the same tweet from a different user.

Unique URLs were first extracted from the 16,000 or so URLs in the data. Next, data from these unique URLs was extracted and then preprocessed. The newspaper package was used to extract article text and the title from the web page. For the articles obtained from URLs, photos and video links for example, from Instagram and Youtube needed to be removed. For this, the data cleaning was achieved by removing articles by limiting word length of the extracted text to about 150 words. This ensured the removal of photos, videos, advertisements, incorrectly extracted articles from the data. After this preprocessing, the number of useful articles reduced from 6003 to 3066.

The final version of the data consists of all tweets along with all the information of the tweet itself, such as the text of the tweet, links to articles if any, hashtags, and so on. The article links from these tweets are stored as a separate file, with information about the articles themselves, along with some preprocessed data. This includes the URL itself and the text extracted from the article, as well as some extracted information such as sentence boundaries, POS tags for tokens, parse trees and dependency trees. This processing of the text was done using the CoreNLP toolkit developed at Stanford Manning et al. (2014).

Tweets are linked to URLs through another file. A URL could have been tweeted through multiple tweets, all the ids of these tweets are linked to the same URL.

4 Analysis

This section details the analyses performed on the data. The analyses mimic the ROUGE-1,2 and L methods of comparing documents, where we compare the tweet and article text using these methods. This is done to determine whether the tweets promoting the articles could be generated from the document text. The results of the comparison show that the tweet is not extracted from the article text.

We calculate the degree of common words - unigrams and bigrams, between the tweet and the text of the document. We also check least common subsequences between the tweet and the document. These are the ROUGE-1,2 and L methods. The hypothesis is that these results give an approximation of the degree to which the tweet is extracted from the document text.

For all these analyses, the stop words have been eliminated from the tweet as well as the document, so that only the significant words are taken into consideration. The hastags, references (@) and URLs from the tweets were also all removed.

4.1 Total match with text in article

To calculate the position of tweet text as a whole in the text, we checked for a complete substring match of the tweet in the text. Out of the 2471 unique instances of tweet text and the article text pairs, where a tweet text was checked against the text in the article, a complete match was found 23 times. 9 times out of these, the tweet text had been matched against the title of the article extracted into the text. The rest of the results are significant, since the text of the tweet appears exactly as is inside the text of the article. For these cases, the user that wrote the tweet went through the article text, and the sentence that either seemed to be the most conclusive contribution of the article, or expressed the opinion of the user was extracted to be tweeted. We also checked to see if the tweet text matched with the article titles, and this was found not to be the case.

Overall, this comparison showed that if the tweet is extracted as a whole from the document, it is either from the title, or actually from inside the document text that was found most appropriate by the user.

4.2 Percentage match for unigrams

Next, we did a percentage match with the text of the article. This was a bag-of-words check using unigrams from the tweet and the document. The order of the words in the tweet or the text did not matter. The results we got seem to suggest that a lot of significant words in the tweet are in fact present in the article. The minimum percentage match obtained was 60%. However, since the order of the words did not matter, this result can be traced back to the fact that tweet is based on the same topic as the document. *Figure 1* shows the percentage of matches in the tweet and the article text as compared to the number of unigrams in the tweet. The mean of the match percentages is 29.53 and standard deviation is 20.2.

Add and words matching

$$unigramMatch = \frac{|unigrams(tweet) \cap unigrams(text)|}{|unigrams(tweet)|} * 100 \quad (1)$$

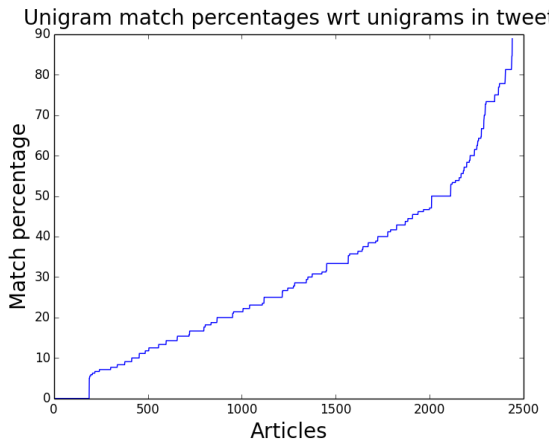

Figure 1: Unigram match percentage.

4.3 Percentage match for bigrams

Similar to the unigram matching techniques, bigram percentage matching was also calculated. The text of the tweet was converted into bigrams and we then looked for those bigrams in the article text. The percentage was calculated similar to the unigram matching done earlier.

Add words matching

$$bigramMatch = \frac{|bigrams(tweet) \cap bigrams(text)|}{|bigrams(tweet)|} * 100 \quad (2)$$

Figure 2 shows the percentages of matches found in every article. Mean is 10.73 with a standard deviation of 18.5.

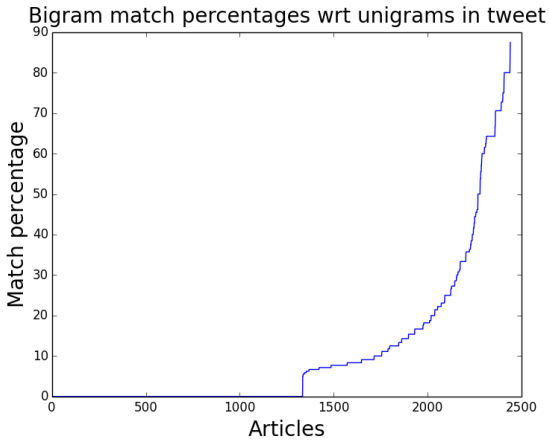

Figure 2: Bigram match percentage.

4.4 Percentage matching inside a window in the article text

The next analysis was to check for a significant word matching inside a two or three sentence window inside the article text. We used a three sentence long window using the sentence boundary information obtained during preprocessing. After the text of the window was extracted, we performed a similar analysis as the last one, except on a smaller set of sentences. Again, the order of the unigrams didn't matter. Next, the matching percentages from all such windows in the articles were compared and the maximum out of these was considered for the highest match percentage and match position for the final results. The result from this experiment is shown in *Figure 3*. Here, the mean of the values is 26.6% and deviation 17%.Add equation for this matching)

(

Add position and words matching

4.5 Longest Common Subsequence match inside a window for the text

The percentage match analyses were a bag-of-words approach disregarding the order of the words inside the texts and tweets. To respect the order of the words in the sentence of the tweet, we also used the least common subsequence algorithm between the tweet text and the document text. This subsequence matching was done inside

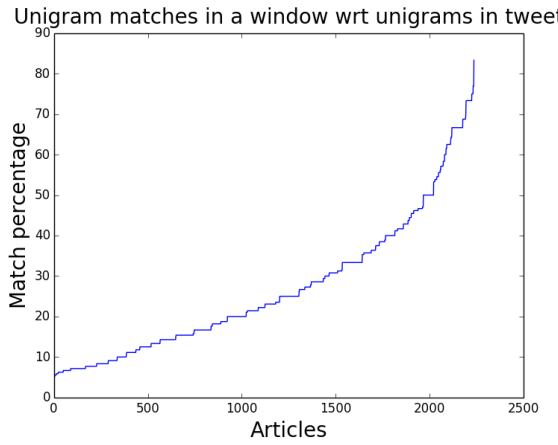

Figure 3: Percentages of common words in tweet and text.

a sentence window of 5 sentences. Again, the final result for the article was the window in which the maximum percentage was recorded among all windows. The percentage match was calculated against the number of words in the tweet, as found in the least common subsequence calculated between the two texts. These numbers are shown in *Figure 4*. The mean here is 44.6% and the standard deviation is 22.7%.

change graph to reflect positions

$$LCSMatch = \frac{|lcs(tweet,text)|}{|unigrams(tweet)|} * 100 \quad (3)$$

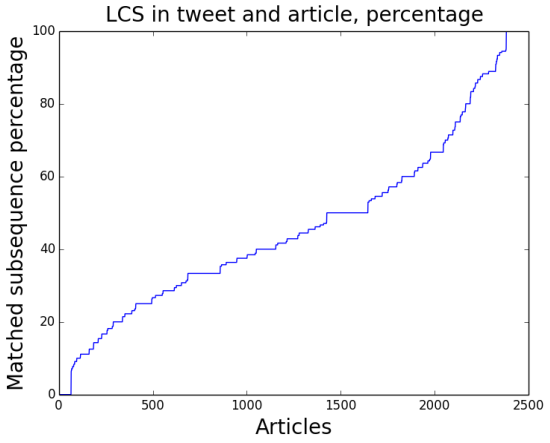

Figure 4: Percentages of words matching in tweet and document text using an LCS algorithm.

5 Results

Discuss * Implication of results - tweet can't be extracted
* Correlation of formality and lcs scores

As seen from the results of the analyses performed earlier, the tweets have little in common with the articles they are related to. The analyses are based on the ROUGE-1,2 and L. This shows that extractive summarization algorithms cannot be directly applied to articles to generate tweets.

expand motive behind calculating formality scores

We also calculated the formality of the articles to correlate it with the longest common subsequence. To achieve this, the degree of formality of the text was calculated with the help of some other studies. The formality lexicon was generated by Brooke and Hirst (2013) and can be used to measure formality of a given text. The lexicon consists of words and phrases and the degree of formality for their occurrence. Thus, more formal words are marked on a positive scale and informal words like those occurring in colloquial language are marked on a negative scale. The degree of formality was calculated using this lexicon.

$$formalityScore = \frac{|unigramsArticle \cap formalitySet|}{|unigramsArticle|} * 10 \quad (4)$$

The formality lexicon gave positive weights for formal expressions and negative for informal expressions. After calculating the formality weights for all articles, it was observed that they all had a total negative normalized weight, meaning a lot more informal expressions were getting matched. Hence, we used just the formal word occurrences for calculating the weight. Thus, above a certain cut-off weight, the article could be considered formal, else would be considered informal. To make sure these formality scores intuitively made sense, we calculated the average formality score for each hashtag used in the search during data extraction and ordered them, shown in *Table* **??**

Lowest formality scores	Highest formality scores
#theforceawakens	#KevinVickers
#TaylorSwift	#erdogan
#winteriscoming	#apec

This formality score for each article was then correlated with the percentage of match obtained using the longest common subsequence algorithm. The Pearson correlation value was 0.41, with a p-value of 7.08e-66. The p-value justifies that we can reject the null hypothesis, and say with confidence that there is a correlation between the formality scores and the ROUGE-L scores of the tweets and articles. Hence, we can say that the more formal the subject or the article, there are

explain this better

higher chances of the tweet being extracted directly from the article.

6 Conclusion

Write

However, after running analyses on the data we discovered that it does not make sense to model tweet generation from articles as an extractive summarization problem. The size of the tweet is too small to be able to form a coherent sentence for the tweet.

Acknowledgments

References

Julian Brooke and Graeme Hirst. 2013. A multi-dimensional bayesian approach to lexical style. In *HLT-NAACL*, pages 673–679.

Julian Brooke, Vivian Tsang, David Jacob, Fraser Shein, and Graeme Hirst. 2012. Building readability lexicons with unannotated corpora. In *Proceedings of the First Workshop on Predicting and Improving Text Readability for target reader populations*, pages 33–39. Association for Computational Linguistics.

Deepayan Chakrabarti and Kunal Punera. 2011. Event summarization using tweets. *ICWSM*, 11:66–73.

Lu Chen, Wenbo Wang, Meenakshi Nagarajan, Shaojun Wang, and Amit P Sheth. 2012. Extracting diverse sentiment expressions with target-dependent polarity from twitter. In *ICWSM*.

John M Conroy, Judith D Schlesinger, and Dianne P O'Leary. 2006. Topic-focused multi-document summarization using an approximate oracle score. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 152–159. Association for Computational Linguistics.

Rumi Ghosh, Tawan Surachawala, and Kristina Lerman. 2011. Entropy-based classification of'retweeting'activity on twitter. *arXiv preprint arXiv:1106.0346*.

Udo Hahn and Inderjeet Mani. 2000. The challenges of automatic summarization. *Computer*, 33(11):29–36.

Liwei He, Elizabeth Sanocki, Anoop Gupta, and Jonathan Grudin. 2000. Comparing presentation summaries: slides vs. reading vs. listening. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 177–184. ACM.

David Inouye and Jugal K Kalita. 2011. Comparing twitter summarization algorithms for multiple post summaries. In *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on*, pages 298–306. IEEE.

Elena Lloret and Manuel Palomar. 2013. Towards automatic tweet generation: A comparative study from the text summarization perspective in the journalism genre. *Expert Systems with Applications*, 40(16):6624–6630.

Christoph Lofi and Ralf Krestel. 2012. iparticipate: Automatic tweet generation from local government data. In *Database Systems for Advanced Applications*, pages 295–298. Springer.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Brendan O'Connor, Michel Krieger, and David Ahn. 2010. Tweetmotif: Exploratory search and topic summarization for twitter. In *ICWSM*.

Lu Wang, Claire Cardie, and Galen Marchetti. 2014. Socially-informed timeline generation for complex events. *constitution*.