Estimating Effectiveness of Twitter Messages with a Personalized Machine Learning Approach

by

Xunhu Sun

A thesis submitted to the Graduate School of
Florida Institute of Technology
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Computer Science

Melbourne, Florida
July, 2015

We the undersigned committee hereby approve the attached thesis, “Estimating Effectiveness of Twitter Messages with a Personalized Machine Learning Approach” by Xunhu Sun.

Philip Chan, Ph.D.
Associate Professor and Major Advisor
Computer Sciences and Cybersecurity

Ryan Stansifer, Ph.D.
Associate Professor
Computer Sciences and Cybersecurity

Ken Lindeman, Ph.D.
Professor
Education and Interdisciplinary Studies

Richard Newman, Ph.D.
Professor and Department Head
Computer Sciences and Cybersecurity

Abstract

Title: Estimating Effectiveness of Twitter Messages with a Personalized Machine Learning Approach

Author: Xunhu Sun

Advisor: Philip K. Chan, Ph.D.

In Twitter, many aspects of retweeting behavior, which is the most effective indicator of spreading effectiveness of the tweet, have been researched, such as whether a reader will retweet a certain tweet or not. However, the total number of retweets of the tweet, which is the quantitative measure of quality, has not been well addressed by existing work. To estimate the number of retweets and associated factors, this paper proposes a procedure to develop a personalized model for one author. The training data comes from the author's past tweets. We propose 3 types of new features based on the contents of the tweets: Entity, Pair, and Cluster features, and combine them with features used in prior work. The experiments on 7 authors demonstrate that comparing to the previous features only. Pair feature has a statistically significant improvement on the correlation coefficient between the prediction and the actual number of retweets. We studied all combinations of the 3 types of features, and the combination of the Pair and Cluster features has the best performance overall. As an application, this work can be used as a personalized tool for an author to evaluate his/her tweet before posting it, so that he/she can improve the tweet to achieve more attention.

Table of Contents

Table of Contents .... **iv**

List of Figures .... **vi**

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

Acknowledgement .... **viii**

Dedication .... **ix**

Chapter 1 Introduction **1**

Chapter 2 Related Work **3**
- **2.1 Goals of Previous Work** **3**
- **2.2 Features Used in Previous Work** **4**
- **2.3 The differences between This Work and Previous Work** **6**

Chapter 3 System Overview and Feature Extraction **8**
- **3.1 System Overview** **8**
- **3.2 Target Feature** **9**
- **3.3 Base Features** **10**
  - 3.3.1 Basic content features 10
  - 3.3.1 Trends feature 10
  - 3.3.2 Time-related features 11
  - 3.3.3 Sentiment features 11
- **3.4 Overview of Additional Features** **12**
- **3.5 Entity Features** **12**
  - 3.5.1 Selecting entities 13
- **3.6 Pair Features** **14**
  - 3.6.1 Calculating similarity between entities 15
  - 3.6.2 Estimating probability of entity 16
- **3.7 Cluster Feature** **17**
  - 3.7.1 Clustering algorithm: AscendingCut 18

Chapter 4 Experimental Evaluation and Result **22**
- **4.1 Data Collection** **22**
- **4.2 Learning Algorithms** **23**
- **4.3 Evaluation Criteria** **24**
- **4.4 Results on Base Features** **25**
- **4.5 Results on Entity Features** **25**

4.6 Results on Pair Features 28
4.7 Results on Cluster Features...31
4.8 Results on Combination of All Features..35

Chapter 5 Conclusion 39

References 41

List of Figures

Figure 1 — Architecture of Tweet Learning and Evaluation Procedure. 9
Figure 2 — Visualization of Actual Number of Retweets and Predicted Value for Each Tweet. 37

List of Tables

Table 1 — 5 Types of Features Used in Previous Work..4
Table 2 — Features Used in This Work..21
Table 3 — Experimental Data Sets Information..22
Table 4 — Results on All Learners Using Base Features..25
Table 5 — Results on Top 10 Entity Features...25
Table 6 — Results on Top 10, 20, and 30 Entity Features 27
Table 7 — Top 30 Entities of Author Greenpeace 27
Table 8 — Results on Top 10 Pair Features 28
Table 9 — Results on Pair Features of All Entity or Word.....................................29
Table 10 — Results on AEMI Using Tweet, Retweet or Tweet + Web..................30
Table 11 — Top 10 Pairs of Author Greenpeace...31
Table 12 — Results on Cluster Features using AEMI to Measure Similarity.........32
Table 13 — Results on Clustering All Entity or Word..33
Table 14 — Top 10 Clusters of Author Greenpeace..34
Table 15 — Results on Using AEMI or Jaccard to Measure Similarity..................35
Table 16 — Results on Combinations of All Types of Features 36
Table 17 — Results on All Learners Using Additional Feature 37

Acknowledgement

Firstly, I want to sincerely thank my advisor Dr. Philip Chan. For the two years, I learnt so much from both his class and the meeting with him. He guides me as patiently as my father. The time of discussing algorithms with him is inspiring and enjoyable.

My sincere thanks also goes to Dr. Ryan Stansifer. His class and the weekly programming practice squad improved my programming skill so much, and also let me know so much friends. He is really a nice and funny man as both a teacher and a friend.

I also want to say thanks to my friend Josh Lear. I learnt from him that a person can change so many things by hard working. I enjoy the time when we programmed and celebrated together.

Dedication

I dedicate this work to my father and mother, Qinghua Sun and Jun Wang.

Your support to me is so warm and unconditional, but what I repay to you is loneliness. Studying aboard is both what I want and what you want, but when I get more and more mature, I feel more and more sad. If there is a chance going back to two years before, I think our decision would still be the same. I wish our lives could be better in the future.

Chapter 1
Introduction

Twitter as a platform of both the news media and social networks has been the subject of much research as of late. Most of the studies are interested in analyzing retweeting behavior, which is after a tweet posted by the author, some readers are attracted by the content of it and are willing to forward it and spread the information. The more the tweet is retweeted, the wider it spreads, so being retweeted shows how influential the tweet is. Some work addressed questions like “What kind of author is more probably retweeted?” [13]. But the tweet author might be not helped by the answers because they are not constructive advice. It is true but useless to point out that to receive more retweets you need more followers, because the number of followers cannot be changed in a short time. The number of followers is the result of good tweets but not the other way around. Some other work answered the questions like “Which reader will retweet the tweet?” [8]. However, most people like to post tweets in public rather than only sending them to the specified readers.

So far as we know, the question that “How to anticipate the popularity of a tweet?” hasn’t been well addressed. The question is motivated by the observation that some tweets are more popular than the others, even they are from the same author. It’s a difficult problem to solve because it’s much harder than telling why a tweet from a celebrity is more influential than from a regular person, or telling whether a football fan will be interested by the tweet or not. It’s a crucial problem because as an author what he / she really wants to know is “Can I write my tweet in a better way so that more people can see it?” So a procedure that addresses this could have substantial marketing or political value.

This work estimates how effective the tweet is in the aspect of the specified author, by analyzing the features from the author’s historical tweets, and training a personalized model for prediction. In additions to features which have proved to be important previously, we develop the additional Entity, Pair, and Cluster features to extract more abstract information from tweet, then train the machine learning regression models to predict the effectiveness of the tweet, which is the number of retweets.

The contributions of this work are, first we proposed a procedure of evaluating tweets in a quantitative way for the specific author instead of making Boolean (retweeted or not) prediction of the tweet by modeling on mixed tweets from many authors. Second is we introduced new types of features: Entity, Pair and Cluster features. The Pair features significantly outperform previous features in terms of Pearson Correlation Coefficient of the prediction on 7 different authors. Third, the combinations of the new features also statistically significantly improved the performance further, and combining Pair and Cluster features has the prediction most correlated to the actual number of retweets.

This is a case study of the Twitter authors related to climate change. The authors analyzed in this work are famous organizations in the climate change area. Nowadays, climate problem like global warming has become more and more serious. Twitter as a media can effectively help the public aware the situation of the climate. So we hope people can take advantage of the procedure of this work to improve the writing of climate message, then let more and more people pay attention to the climate change.

The remaining parts of the paper are organized as follows: In Chapter 2 reviews problems and features researched in previous work. Chapter 3 presents the structure of the whole system, and the basic and our proposed features. Chapter 4 is the experimental details and the results and analyses. Conclusions and possible improvements are in Chapter 5.

Chapter 2
Related Work

2.1 Goals of Previous Work

Basically, there are 3 types of questions answered by previous work: 1, “Is the tweet a retweet?” or “Does the tweet have retweets?” 2, “For a given reader, which of the received tweets will be retweeted by him/her?” and 3, “Given a tweet, which reader will retweet it?” To achieve different goals, there are mainly 3 types of models in the previous work: Global model, Tweet recommending model, and Reader evaluating model.

A number of studies [5, 9, 10, 13] have proposed algorithms which usually crawl tweets as training data, use whether one tweet is a retweet or has retweets as training target, and identify a model between the retweeting behavior and features of author, tweet, or reader. We call it "Global model”. The model is trained by the data from many authors. The global model can answer the question “Is the tweet a retweet?” or “Does the tweet have retweets?”

Instead of building a global model, some papers [3, 15, 16] propose the “Tweet recommending model”. They pay attention on the reader, study “For a given reader, which of the received tweets will be retweeted by him/her?” The trained model can be a system to recommend tweets for the reader.

The “Reader evaluating model” finds the readers who are more likely to retweet. The models in [8, 15] control the author-based and tweet-based attributes, to observe “Given a tweet, which reader will retweet it”, and the result can benefit business promotion and information dissemination.

2.2 Features Used in Previous Work

Generally there are 5 types of features used in previous work (as Table I shows): author-based, tweet-based, reader-based, author-reader-based, and tweet-reader-based. Based on the goal to achieve, the different model uses the different types of features.

Table 1 — 5 Types of Features Used in Previous Work

Feature type	Description
Author-based	The publisher of the tweet
Tweet-based	Tweet content or publishing time
Reader-based	The person who retweets the tweet
Author-reader-based	Relationship between author and reader
Tweet-reader-based	Relationship between tweet and reader

On features related to the author, some researchers [3, 5, 13, 15] find that the number of followers / followees of the author are correlated to the number of retweet. The days since the author registered Twitter and the number of favorite tweets are also checked by [13] and [15] but the result is showing no obvious influence. Uysal et al. [15] also utilize the total tweets count, the tweets count per week, number of times the author has been listed, is author a verified user, does user profile have description or url, is the language English. Feng et al. [3] take advantage of the author's user id and location id which are rarely used in other papers, and also the prior probability of being retweeted, time span since last time being retweeted, and number of times the author mentioned by others.

Lots of tweet-based features have been proved to be quite important for learning, like whether the tweet contains url / hashtag / image, or whether it mentions someone [2, 3, 5, 13, 15, 16]. Uysal et al. [15] find tweet based features outperform others, which include question mark, exclamation mark, quotation mark, emoticons, length of tweet, tf-idf, first person pronoun, and same character consecutively three times. Naveed et al. [10] measure the sentiments in tweet by Affective Norms of English Words (ANEW) dictionary [1], and positive and negative terms by their predefined word list. Quercia et al. [12] take the category of words as features for training, they distinguish positive and negative emotional words from tweet using Linguistic Inquiry Word Count (LIWC) [11], and they also consider person pronouns, tenses of verbs, cognitive words and time words. Macskassy et al. [9] use information from Wikipedia to decide the topics of tweet and user, then measure the similarity between them to predict retweet probability.

Many reader-based features have poor performance, which include prior probability of retweet (retweet willingness) and features the same with author-based [3, 15]. However, Kyumin Lee et al. [8] introduce time-related features called readiness features since user may not have the chance to see the tweet at certain time. They analyze tweeting likelihood of the day and hour of a user by taking a ratio of number of tweets on given day/hour and the total number of tweets, the tweeting steadiness of the user by measuring the standard deviation of elapsed time between consecutive tweets, and the last time user tweet something.

The relationship features perform an important role in the experimental result [3, 15, 16], especially in the work on ranking or recommending tweets to the certain reader. The author-reader features represent the closeness and interaction between the author and the reader, including the reader's mention / retweet / reply count of author, the time span since last interaction, they are friends or in the same location or not, and the similarities between their tweets, recent tweets, self-descriptions, and following lists.

The tweet-reader features describe the relationship between the tweet and the reader [3, 15, 16], such as whether tweet directly mentions the reader or has hashtags used recently by the reader, or the similarities between the tweet and the user's historical tweets or recent tweets.

2.3 The differences between This Work and Previous Work

Our work proposes the “Author personalized model”, which answers the question “For a particular author, which tweet is better”. It’s different from the “Global model”, because our model excludes the author-based features so that we can concentrate on “How does a popular tweet look like?” instead of “How does a famous author look like?” Our work is also different from the “Tweet recommending model”. Although both of the two models check “what’s a good tweet”, the “goodness” of tweet in the prior wok is based on the interest of the certain user, while the “effectiveness” in this paper represents the public interest. It’s obvious that the “Reader evaluating model” is different from this work because we do not post tweets to specified readers.

The training tweets of Global model and Tweet recommending model are both the original tweets from some authors and the tweets retweeted by some authors. But in this work, we only use original tweets as training data, because our model is trying to estimate the effectiveness of the tweets written by the author, not by some others.

The target value of this paper is quite different from previous work as well. The target value in most previous work is a Boolean prediction which is usually “retweeted or not”, however, this work is trying to predict a continuous value correlated to the number of retweets. The difference between these two is like the ways to describe the tweet “Is it good?” and “How good can it be?”

Among 5 types of features, our study requires only tweet-based features, because we fixes the variable of the author to find more effective tweets. So far only the general features on tweet content have been well examined, it’s still possible to mine deeper in the tweet to reveal more information that has not been utilized. For instance, usually a tweet containing “retweet this please” or in short “plz rt” can have a higher chance to be retweeted. That means people are more likely to be persuaded by certain words, while in previous paper only the emotional words in the dictionary have been tested, words like “rt” exist only in Twitter world are still unexploited. Feature like maximum length of the words in the tweet also can be valuable, because if a tweet contains words which are quite long and hard to understand, the public could lose interest in it.

Chapter 3
System Overview and Feature Extraction

3.1 System Overview

Our work is a personalized tool for helping an author evaluate tweet before publishing it, by using author's previous original tweets to train a model predicting how many retweets a tweet could receive. Fig. 1 shows the architecture of the tweet learning and evaluation procedure, which consists of tweet learning procedure and tweet evaluation procedure. In the tweet learning procedure it takes the old tweets of the author as a data set, each original tweet is turned into a training instance by extracting features, and the number of the retweets of the tweet (after logarithm) is the target value of learning. Then the learning algorithm will use the instance set to train a predictive model.

In the tweet evaluation procedure, when the author wants to post a tweet, the system will extract features of it, then the model will provide a score of the quality which is correlated to the number of retweets it could get, so the author can modify the tweet (like adding a photo or using more sensitive words) to make it better. The author can repeat the procedure multiple times until the tweet is optimized for posting.

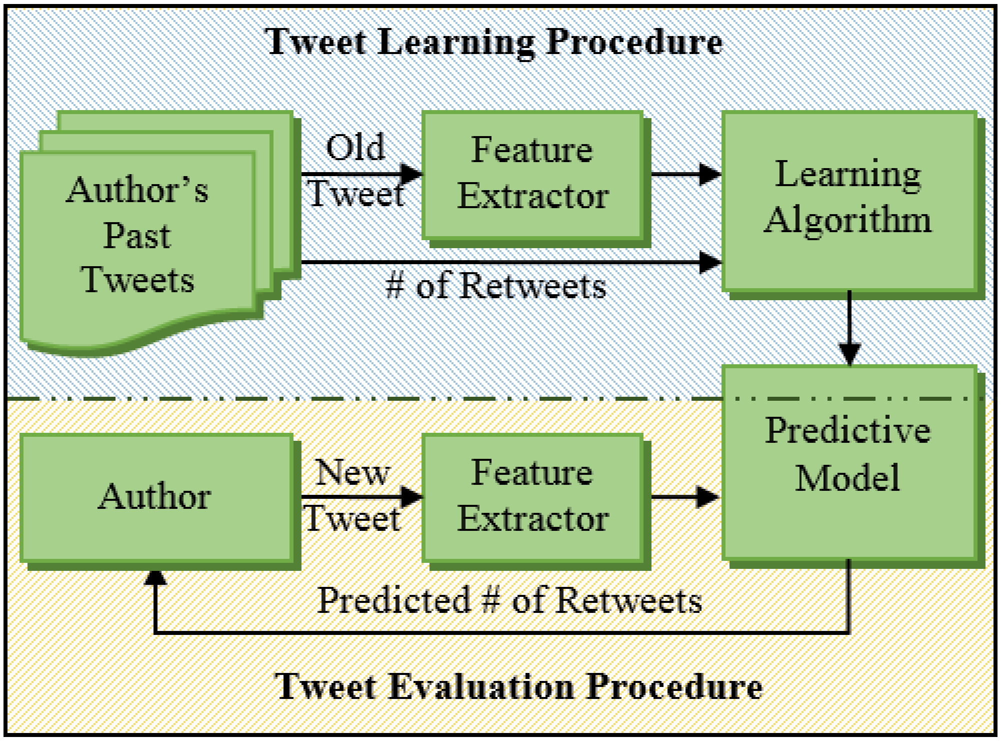

Figure 1 — Architecture of Tweet Learning and Evaluation Procedure.

3.2 Target Feature

The target feature is the value which our learning model is trying to learn and predict. The target value is the logarithm of (number of retweets of the tweet + 1). We use the number of retweets to represent the effectiveness of a Twitter message, as more retweets of a tweet means the more impressive it is and also the wider it spreads. We take the logarithm of the value because the number of retweets varies widely (see Table 3 in Chapter 4). The learning algorithm is easier to handle the value after the logarithm. And also it's a monotone increasing function so we can still tell between two tweets which one is better after the logarithm.

3.3 Base Features

All the training features of this work are tweet-based features (see Table 1 in Chapter 2). The reason is that we are building a personalized tweet evaluator for a particular author, so the author-based features are useless since the attributes of author rarely change, and the features related to reader are not quite relevant as well since the group of the followers doesn't vary too fast either.

The "Base" feature are mostly features in a similar form of in previous work. These features are effective and easy to be extracted, so we want to use them as a base line to compare with additional features which are more complex and abstract.

3.3.1 Basic content features

The features "Does the tweet include a photo/URL/hashtag /mention" have been proved to be quite important by related works. These entities usually contain more attractive information (like a video from the URL) for a reader, then the reader could be more likely to retweet the tweet and spread the information to others. A tweet which is pretty long or contains some word hard to understand could lose the interest of public, even though Twitter has a limitation of 140 characters. We assume that a word with more characters means it's more sophisticated. So we introduce the length of text and the length of longest word as features for the readability of tweet

3.3.1 Trends feature

Trends[1] are topics information provided by Twitter which refer the top 10 trending subjects at a particular time for a specific location. User can click on the link of topic keyword in the Twitter home page then the related tweets or authors will be shown. The trending information can tell us what's having the public's attention, and if the tweet contains word of the trends and joins the discussion, it would lead to a higher chance to be searched and more feedback from people. We give the feature Trends Boolean value to measure whether

[1] https://mobile.twitter.com/trends

it contains trending word or not, the trending location has been set to global since the authors in the experiment are world-wide organizations.

3.3.2 Time-related features

Even if the content of tweet is attractive enough, it still could have low retweet count if it is published at a bad time (like midnight) when most people are off the Internet so there is less chance of seeing the tweet. The feature Day in week is extracted by the published time of tweet, the value is set from 0 to 6 when the day is Monday through Sunday. The feature Hour in day is set from 0 to 23 to represent 0am to 23pm in the day. We expect that there should be more retweets in weekend and also the evening of the working days.

3.3.3 Sentiment features

The sentiment contained in a tweet can effect readers' emotions and their retweeting behavior. We applied 2 ways to extract sentiment from words in tweets.

The Affective Norms of English Words (ANEW) dictionary [1] measures the emotional ratings for English words in 3 dimensions which are valence (pleasure or displeasure), arousal (excitement or calmness), and dominance (weakness or strength). ANEW scores a word with the value between 1 and 9 in those 3 dimensions separately, so we extract 3 features Valence, Arousal, and Dominance of a tweet by taking average score of all words. The word cannot be found in ANEW dictionary will be ignored, and if a tweet doesn't contain any word in ANEW, it will be a neutral score, which is 5.

SentiStrength algorithm [14] analyses tweet and gives two scores describing positive and negative sentiment of the tweet. The positive sentiment ranges from 1 to 5, but the negative one ranges from -1 to -5. For convenience we makes the features Positive Sentiment and Negative Sentiment vary both from 1 to 5, which 1 means no sentiment at all and 5 stands for strongest sentiment.

3.4 Overview of Additional Features

Additional features are developed based on the content of tweet writing of the author, to reveal the author's habit of language usage and the field of interest. For example, some of the authors like to ask reader to help forward the tweet, but author Greenpeace hardly persuades public in this way, instead, it usually says "stop xxx", like "stop hurting the earth", "stop polluting the arctic". The most intuitive advantage of the features we designed here is that they are personalized feature, they are trying to analyze in the view of the served author. Just like in the last example, if the model is not targeted to Greenpeace, the word "stop" might not be considered as feature, a global model would be busy on examining word "rt" because many people like to use it, like ClimateReality who uses it a lot.

We propose 3 types of additional features: Entity, Pair, and Cluster feature. The Entity feature utilizes the elements in the tweet, mostly the words, but also hashtags, users mentioned, and domains from the links in the tweet. Then we group up the entities into pairs, the Pair feature could possess more underline meaning and less ambiguity than only a word. The higher level abstract information of the tweets is mined by clustering entities into topics interested by the author.

3.5 Entity Features

An entity is a word, hashtag, mention, or domain existing in the tweet content, and the feature is "Does tweet contain this entity". The motivation of listing particular entities out as features is that some special words usually are more welcomed than others, even they do not imply any emotion. The most "magic" term in Twitter is "please retweet this" or a shorten version "rt plz", as readers are more likely to help spread the information if they were asked. Similar to word, a tweet having a hashtag "#ClimateChange" or mentioning "@Obama" or having a link of "YouTube.com" might attract more people as well. We are trying to dig out the most influential entities from which the author can benefit. So we score and rank all entities of a certain author's historical tweets, and take the top 10 (or 20, 30) entities as 10 separate Boolean features, like "Does the tweet has the word 'retweet'?"

The domain is extracted from the expanded URL of the tweet. The URL in the tweet is shorten to the format like this: https://t.co/xfAX1z1mp2, and the API provided by Twitter is able to give the original link of it. But actually most of the time the original link is still a URL shorten by another web site. So here the domain used as entity comes from the URL redirected back to the very original address.

3.5.1 Selecting entities

Here we introduce 3 ways of scoring entities to select the best ones: 1, pick the most frequent entities (DF); 2, take the sum of number of retweets related to the entity (Sum); 3, average the sum by inversed tweet frequency (IDF).

Firstly, the tweet frequency of an entity (DF) measures how many times the entity appears in the tweets. Here we only count the number of tweets, so if word "please" appear 3 times in a tweet, it's still counted as one tweet, as in (1):

$$DF(e) = |T(e)| \qquad (1)$$

The *T(e)* is the set of tweets containing entity e. Usually in some related works, when scoring a word such as tf-idf, a word occurs too many times will be penalized, but in the scenario of tweet, the most frequent non-stopped word usually has around 10% tweet frequency over all tweets of an author. Hence high tweet frequency is a considerable quality for a word, otherwise if we accepted a word hardly appears in the training set, it might never reappear in the test set, then the feature of the word would be useless.

Secondly, to estimate the importance of a word, we take a sum of number of retweets of tweets which includes the specific word, as in:

$$Sum(e) = \sum_{t \in T(e)} nrt(t) \qquad (2)$$

In the above equation, *e* represents a particular entity, *T(e)* means the set of tweets containing entity *e*. The *nrt(t)* is the number of retweets of the tweet *t*. Sum prefers the word which brings more retweets as well as which appears lots of times, that means a word appearing a few times in only popular tweets has an equal chance to be selected than the one appearing in lots of tweets

Finally, because Sum could have a problem that it biases the frequent entity so much that it has similar result with the DF, then a word appearing fewer times but having great contribution cannot be found out. To prevent that, a method taking average of retweets for a word is considerable. However simply taking Sum divided by tweet frequency is highly bias to the entity occurring only once or twice in the mostly retweeted tweets. So the impact of tweet frequency should be reduced by taking logarithm, which is the way of using inversed document frequency (IDF) as in (3):

$$IDF(e) = Sum(e) * \ log\left(\frac{N}{TF(e)}\right) \qquad (3)$$

The *Sum(e)* is the equation (2), the *N* is the total number of tweets, and *DF(e)* is the number of tweets which contain the entity *e*. IDF is making a balance of selecting between a frequent word and an averagely influential word. The test only concerns the entities occurring in at least two different tweets, so an entity appearing only once in all the historical tweets of an author doesn't have a chance.

3.6 Pair Features

Pair feature takes the pair of entities (could be word, hashtag, mention, or domain) which co-occur in the same tweet. The feature is like “Does the tweet have both entity A and entity B?”. The pair could express more accurate meaning than the word, like when we talk about “apple” and “banana” most of the time our discussion is not regarding the mobile phone.

Similar with Entity feature, we want to score the pairs and select some of them as features. There are 2 general ways to score the pair. The first way is using DF, Sum, IDF of Entity feature section, instead of applying the methods to a single entity, we apply them to an entity pair, so the pair can be measured by number of occurrence, total number of retweets, and average number of retweets. The second way we are introducing is measuring the similarity between 2 words, and selecting the pair with the words highly related to each other. The idea behind this is that 2 words could appear together occasionally (like pairs in the first way), but if they are more likely to occur together and rarely show up alone, that means they must share the similar meaning or belong to the same category. So using similarity score to measure the pair can make sure that the pair exists by the traditional usage or the author's habit rather than by chance, as a result, we can expect that it would still exist in the data of the test set later.

3.6.1 Calculating similarity between entities

We use 2 methods to measure the similarity of words: AEMI and Jaccard.

Augmented Expected Mutual Information (AEMI) [6] measures the mutual information between 2 words by considering both the co-occurrence and the sole occurrence, it's defined as:

$$AEMI(a,b) = P(a,b)log\frac{P(a,b)}{P(a)P(b)} - P\left(a,\bar{b}\right)log\frac{P(a,\bar{b})}{P(a)P(\bar{b})} - P(\bar{a},b)log\frac{P(\bar{a},b)}{P(\bar{a})P(b)} \quad (4)$$

The *a, b* represent any 2 entities, *P(a)* means the probability of occurrence of *a*, and *P(a,b)* is the probability of occurrence of *a* and *b* together. *P(ā)* is the probability of *a* not occurring, and *P(ā,b)* is the probability of *b* occurring while *a* not occurring. The first component of (4) is the supporting evidence that *a* and *b* are related, while the remaining parts are the counter-evidence of that. So a high AEMI of 2 words means they must have high probability of co-occurrence, and low probability of occurring without each other.

Jaccard calculates the similarity by the number of occurrence of two entities divided by the number of occurrence of at least one of them, as in:

$$Jaccard(a,b) = \frac{|T(a) \cap T(b)|}{|T(a) \cup T(b)|} \qquad (5)$$

The *T(a)* is the set of tweets containing entity *a*, it’s taking the intersection of 2 sets in the numerator and taking the union of them in the denominator, and dividing the sizes of the two sets.

3.6.2 Estimating probability of entity

For the probabilities in AEMI and the tweets counts in Jaccard, there are 3 ways to calculate it: 1, based on original tweets; 2, based on original tweets and web pages; 3, based on number of retweets. With original tweets, probabilities are estimated by the number of tweets, which means *P(a)* is number of tweets divided by total number of tweets. The problem of probability based on only tweets is that they are too few and contain so few words in each. So the probability of the infrequent word could vary extremely from 0 to 1, even after applying m-estimate. As a result, when the infrequent words group into pairs, it’s hard to tell the high value pair really has more inner similarity than others. To ease the lack of information of original tweets, we could borrow the web pages as complementary documents.

A second method of estimating probabilities is to utilize web pages mentioned in author’s tweets as additional materials. When a web page link is written in a tweet, usually it’s a page of news of video, which is helping demonstrate the idea of the tweet, in other words, the page is showing what the author want to say, so web pages can be regarded as the extension of the author’s historical tweets and they are on the same area interested by the author. A web page has much more words than a tweet, so it’s much easier to mine out how much the words are related to each other naturally. The drawback of web pages is that unlike tweet or retweet, it’s not using “Twitter language”, such like word “rt” (retweet) and “mt” (modified tweet) are related in Twitter world, but it doesn’t often happen in web pages, and hashtag and mentioned user cannot be found in most of web pages either. The other problem of web pages is that in practice the crawled page has lots of noisy information but the main article, such like the words in the web side title, menu, and advertisement which could be anywhere. We apply domain stop words to discard the noise as much as we can. The domain stop words list is built from all the crawled pages of the same domain, if a word appears in more than 80% pages, it’s considered as a noise from the web site rather than a word from the article, and then it will be added to the list, and later when we use the page of the domain to calculate probability, this word will be ignored.

A third method, instead of probability based on sum of tweets, the probability of an entity based on the sum of retweets number of the entity as calculated in:

$$P(a) = \frac{\sum_{t \in T(a)} nrt(t)}{\sum_{t \in T} nrt(t)} \qquad (6)$$

T is all the tweets of the author, *T(a)* is the tweets including entity *a*, *nrt(t)* is the number of retweets of tweet *t*, and the *P(a,b)* is similar with this. Using the number of retweets to calculate the occurring probability of the entity looks a little strange, but actually it still makes sense. The reason is that the retweet is not only the number belong to the original tweet, but also the copy of tweet republished by the reader, for example if tweet X has 1000 retweets, those will be 1000 tweets in the readers account, so the words appearing in X also appear in other 1000 tweets. In this aspect, probability based on number of retweets is still describing the occurring chance of the entity.

Retweets is another view of examining pair which inclines pairs that occur a lot in the retweets. The target of the system is the number of retweets, but the original tweets and web pages has nothing related with the target. So the method based on retweets is able to take advantage of the additional information than tweets and web, and moreover, it is one of the solutions to the problem of differentiating the strength of the link between infrequent words. Even though a retweet has as few words as a tweet does, the number of retweets is much larger than number of original tweets, so it brings more information of words and makes the probability calculation smooth.

3.7 Cluster Feature

The Cluster feature is grouping all the entities used by the author into clusters, then the entities within one cluster express a topic. So given a tweet, we can tell which topic it’s talking about by checking the entities of it. Since a tweet could related to more than one topic, the feature is designed as proportional value. For example, if a tweet has 10 words, 4 of them belong to cluster A, 1 of them belong to cluster B, and others do not belong to any one, then feature A is 4 / (4+1) that's 0.8, and feature B is 0.2, that means the tweet is related to topic A, but also mentions a little on topic B. The Pair feature mentioned in last section is not a mini version of Cluster feature. The Pair feature requires both of the entities appearing in the tweet, so one feature only covers a few tweets mentioning the limited situation, it's more specific, one Cluster feature, however, works even if there is only one word in the tweet referring to the topic.

A Cluster feature has 2 main advantages comparing to an Entity or a Pair feature. Firstly, the cluster represents the higher level of abstract information expressed by tweet than a single entity or pair. Like some tweets of Greenpeace contain word "wind", some have "solar", and some include "nuclear", actually they are all talking about the topic related to renewable energy. So after extracting topics from tweets, it's possible for learning algorithm to relate them to the number of retweets and tell what kind of topic is interesting or annoying for readers. Secondly, cluster can condense large amount of information of tweets into limited number of features. Unlike top 10 entities or pairs can only have 10 or 20 entities involved, top 10 clusters can include hundreds entities, as a result, the tweet writing in infrequent words (like "solar") can take advantages of the Cluster feature as well.

3.7.1 Clustering algorithm: AscendingCut

To cluster words into groups, we use the words similarity mentioned in Pair feature. But the classic clustering algorithms can hardly work well on this problem. Firstly, some of the clustering algorithms like K-means have to calculate the mean point among instances, but here we cannot create an average word out of many words. Secondly, some algorithms not requiring mean point, such as hierarchical clustering algorithm, need distance between instances instead of similarity. 1/AEMI and 1/Jaccard could be used as distance function, however, they do not make sense because AEMI can be 0 or negative and Jaccard can be 0 too, the inverse of them as distance is undefined value. And even we use m-estimate or manually assign a big value for the undefined distance, it still has a problem that there are great amount of words having no relationship with some other words, so if hierarchical clustering tries to merge clusters, the distance between clusters will be overwhelmed by the big values, so the merging operation might become random.

This problem is more like a graph problem, where some words are not connected to each other. But some graph clustering algorithms, like EdgeBetweenness or BridgeCut [4], are too slow when number of edges is large and not so suitable for weighted graph. Moreover, we require the algorithm to allow a word not belonging to any clusters if the word is irrelevant to any main topic of the author. So we propose AscendingCut to find the clusters depending on the similarity of words. A cluster is defined as a group of words which have strong similarity between each other and weak relationship with the words outside the group. To find out this kind of cluster, we can remove the edges between words from weakest to strongest, then finally the cluster will be totally disconnected from all other words. We need to specify the maximum size of the cluster, otherwise all the words can be considered as a whole cluster as well. Here is the pseudo code of the algorithm:

Algorithm 1: AscendingCut

```
Input: Similarities of all words, MaxClusterSize, StepNum
Output: ClusterList
Build graph G where node is word and edge is similarity
Find the Min, Max value of edges
StepSize = (Max - Min) / (StepNum – 1)
Threshold T = Min
While T <= Max and G is not empty
    Remove all the edges of G lower than T
    T = T + StepSize
    Find all the isolated subgraphs in G
    For each subgraph S
        If size of S <= MaxClusterSize
            Add new cluster S to ClusterList
            Remove S from G
Return ClusterList
```

In the algorithm, the input similarities of words is mentioned in Pair feature section, and firstly we convert the similarities into a whole graph by using the word as the node and the similarity between words as the edge. Then we set up a threshold and remove the edges which the value is lower than the threshold, and by keeping on increasing the threshold, all the edges are possible to be visited. Every time after discarding some edges, we could have some subgraphs totally isolated from the whole graph, and if the subgraph is smaller or equals to the maximum cluster size, it can be accepted as a cluster and removed from the whole graph. The algorithm could be disconnect edges one by one from lowest to highest, and looking for the new clusters every time after that. But instead, we set a step number and remove many edges in one step, because the former way is too slow in practice and the current way hardly affects the result if the step number is big enough. The first method's time complexity is $O(E^2)$ in which E is the number of edges, because removing edges one by one is E operations and inside the loop finding isolated subgraphs at line 10 needs to visited E edges in worse case. While in Alg. 1 the time complexity is $O(StepNum * E)$, because the outer loop at line 7 only runs at most step number times. In this work we set *StepNum* to be 1000, which is still saving plenty of time, when there are thousands of nodes so the number of edges could be millions. To further speed up the algorithm, the edges with 0 or negative AEMI value can be discarded in advance because it means the similarity between the 2 words is even lower than it between random words.

In this work, both AEMI and Jaccard are studied as the similarity function, and also as Pair feature, the probability of AEMI can come from original tweet, web page, or retweets. The max size of cluster is set to be 20 words because we don't want the cluster to be quite large then the topic would be too general. After clustering, we pick up top 10 largest clusters as 10 features. The reason of using size as criteria is first we definitely don't want to use singletons, which are a great many, secondly, since the maximum size is already relatively small, we prefer the topic described by a variety of words, it has more effort of the author and also higher chance to get matched by the tweets later.

A summary of the features is shown in Table 2. All the features with continuous value will be normalized into range 0 and 1. The words used in the features are lower-cased, and discarded by the stop word list, but not stemmed, since with or without stemming do not make a lot of difference in the result of the test, so we keep the original form of word which is easier to understand.

Table 2 — Features Used in This Work

Type	Num	Name
Base	14	Include photo, Include URL, Include hashtag, Include mention, Length of tweet, Length of longest word, Trends, Day in week, Hour in day, Valence, Arousal, Dominance, Positive Sentiment, Negative Sentiment
Additional	30	Top 10 entities, Top 10 pairs, Top 10 clusters

Chapter 4
Experimental Evaluation and Result

4.1 Data Collection

This is a case study of authors related to climate change. We selected 7 related organization accounts (Table 3) recommended by Twitter Popular Account[2]. We called Twitter API through Twitter4J[3] to crawl 7 accounts' tweets from January 27 to June 15, 2015. The Twitter Streaming API[4] kept pushing to us the new tweets of the authors as soon as they posted them, then a timer for each tweet was set, and after 24 hours we used Twitter REST API[5] to crawl the tweet again to get the number of retweets it achieved at that time. We set the time threshold to be 24 hours because [7] shows that 75% of retweeting occurs within one day.

Table 3 — Experimental Data Sets Information

Author Name	TrainSet	TestSet	AvgRt	MinRt	MaxRt
ClimateDesk	320	37	14.80	1	175
Climateprogress	1553	426	33.25	1	348
ClimateReality	3816	997	29.41	1	624
EarthVitalSigns	126	25	42.82	6	239
Greenpeace	1000	383	132.28	3	1110
UNEP	737	477	42.73	3	499
UNFCCC	559	357	36.07	1	712

[2] https://twitter.com/who_to_follow/interests/social-good

[3] http://twitter4j.org/en/index.html

[4] https://dev.twitter.com/streaming/overview

[5] https://dev.twitter.com/rest/public

Table 3 shows the details of data from each author. The columns are the number of instances in training and test set, and the average, minimum, and maximum number of retweets of each author. Some authors have relatively large amount of data for learning, like Greenpeace which has the highest average number of retweets and ClimateReality which posts the most tweets. While some other authors, like EarthVitalSigns and ClimateDesk, have much fewer instances than others, so it's harder to get good performance on them for learning algorithms because of the lack of data. So later when we evaluate the performance of learning models, we concern with the average result of all authors as well as the results of 2 important authors: ClimateReality and Greenpeace.

For each author, only the original tweet will be an instance, namely the tweet retweeting from others won't be used, because the retweeted one doesn't reflect the writing custom of the author. And the tweet having no retweets is excluded too, because the zero retweeting count could be caused by network problem of the crawling program, so the tweet could be noise in some sense. Then instances from all the tweets created before May 15, 2015 are in the training set, and the tweets after that are in the test set. We test the model in a similar way of tweet evaluation procedure in Figure.1, we test the model by putting the tweets of test set into the predictor.

4.2 Learning Algorithms

We trained both Linear Regression (LR) algorithm and Artificial Neural Network (ANN) algorithm in WEKA toolkit[6] and Support Vector Regression (SVR) in LIBSVM[7] to estimate the retweeting number. The Linear Regression is using the default setting which is Akaike criterion for model selection, M5 method for the attribute selection. The ANN has one hidden layer and the number of hidden nodes equals to a half of the number of attributes plus 1. The hidden unit has sigmoid threshold, and the only one output unit has no threshold. The learning rate and momentum rate of ANN are both 0.1. For the SVR we

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

[7] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

use both epsilon-SVR (EpSVR) and nu-SVR (NuSVR), the kernel function is radial basis function.

4.3 Evaluation Criteria

We use Pearson Correlation Coefficient (PCC) to evaluate the learned model. PCC measures the linear dependence between the predicted values (number of retweets estimated by algorithm) and the actual values (number of retweets of the tweet), giving a value between +1 and -1, where 1 means total positive correlation, 0 is no correlation, and -1 indicates total negative correlation, as in:

$$PCC(P,T) = \frac{E[(P-\mu_P)(T-\mu_T)]}{\sigma_P \sigma_T} \qquad (7)$$

Here P and T are predicted values and actual target values, μP is the mean of P, and σP is standard deviation of P, and E is the expectation. We expect the prediction can be positively correlated to the actual number of retweets so that it can be used to evaluate the quality of the tweet. PCC is the main criteria of this work because we want to determine whether a tweet is better than the other, so the ideal model should give a score to a tweet so that the tweet with more retweets has higher score and the one with less retweets has lower score, in this case, PCC is an efficient way to measure the correlation between the prediction and the number of retweets. On the contrary, we do not expect the learner to predict exactly the correct number. For example, if 3 tweets have 100, 200, 300 retweets separately, and the model scores them as 10, 20, 30, they are still effective estimates.

In the test results, we compare the PCC of using only Base features with the PCC of using Base features + additional feature set. We consider whether the average PCC on 7 authors get improved, and also we perform the paired t-test to determine whether the improvement is statistically significant. The paired t-test is two-tailed and the confidence level is 95%. If the feature set has a significantly effective result, we will mark it in the table.

4.4 Results on Base Features

Table 4 shows the correlation result on the test set using base features, the rows are 4 different algorithms, and each column is the correlation for the corresponding author except the last column is the average performance of the algorithm, the bold cell means the best algorithm for the author. Correlations show the EpSVR has the best result on 3 authors out of 7. Since EpSVR works pretty well in the test on base features, we will use EpSVR as the learning algorithm in the later comparison of other features.

Table 4 — Results on All Learners Using Base Features

Algorithms	**PCC of each author**							
	ClimateDesk	**Climateprogress**	**ClimateReality**	**EarthVital Signs**	**Green peace**	**UNEP**	**UNFCCC**	**Average**
LR	**0.355**	0.343	0.585	0.319	0.409	0.430	**0.503**	0.420
ANN	0.264	**0.356**	0.571	0.202	0.355	0.459	0.453	0.380
EpSVR	0.312	0.353	0.599	**0.445**	**0.431**	**0.459**	0.495	**0.442**
NuSVR	0.292	0.354	**0.601**	0.416	0.407	0.458	0.499	0.432

4.5 Results on Entity Features

Table 5 — Results on Top 10 Entity Features

Feature Set	**PCC of each author**							
	ClimateDesk	**Climateprogress**	**ClimateReality**	**EarthVital Signs**	**Green peace**	**UNEP**	**UNFCCC**	**Average**
Base	0.312	0.353	0.599	0.445	0.431	0.459	0.495	0.442
All DF	0.277	**0.358**	0.606	0.342	0.408	0.482	**0.538**	0.430
All IDF	0.360	0.351	**0.609**	0.491	0.419	0.484	0.514	0.461
All Sum	0.354	0.338	0.608	0.465	0.413	**0.492**	0.527	0.457
Word DF	0.275	0.334	0.601	0.478	0.436	0.469	0.510	0.443
Word IDF	**0.366**	0.351	0.608	**0.540**	**0.438**	0.475	0.499	**0.468**
Word Sum	0.295	0.346	0.608	0.514	0.435	0.470	0.515	0.455

a. All means selecting all types of entities, Word means selecting only words.

Table 5 shows the comparison of all types of entity feature and only word entity feature with different ways to pick up them. The learning algorithm here is EpSVR, and the first row Base is the result using only Base feature as a base line for comparison, other rows are results of Base feature + different type of top 10 entity feature. Generally IDF is better than DF and Sum for both All and Word, and outperforms Base in average. DF has a performance even worse than Base, which indicates that the words author most likes to use can mislead the learning algorithm. Regarding the difference between All and Word, it seems that introducing more types of entity helps some authors but for author Greenpeace and EarthVitalSigns, it performs worse than Word, and even worse than Base on Greenpeace. That means certain introduced hashtag, mention, or domain could have the effect of misguiding learning algorithm.

A feature could be either helpful, useless, or harmful, and the problem here is caused by the harmful feature which plays a significant role in the training set but not really much in the test set, so the learning algorithm is misled by it while training. As an example of harmful entity, #thecrossing is the 5th most frequent entity of author Greenpeace (Table 7) which is mentioned 41 times in the training set but never appears again in test set. The machine learning algorithms require the data distribution of training set is the same with the test set, but this kind of word that is only hot for a while could let the learner credit it with the reason of high (or low) retweets and overlook other components of the tweet, then the learner would have a tough time on test set. And this type of hot topic word is hard to be caught by the Trends feature of Base, because like #thecrossing, it’s an activity related to the author instead of a popular topic for global Twitter. On the contrary, picking up useless feature, like word rarely appears in neither training nor test set, usually doesn’t hurt PCC so much, because in most case learning algorithm could ignore the feature then its performance would be similar to Base. And that might be the reason of why IDF, which prefers the entity with less occurrence, outperforms DF and Sum, which have higher chance to pick up harmful entity as feature and yield a lower PCC than Base.

Table 6 shows the result of all types of entity with IDF, to compare picking up top 10, 20, and 30 entities. It shows that increasing the number of entities doesn’t necessarily improve the correlation. The PCC of Greenpeace gets worse and worse when the number of entities getting larger and larger.

Table 6 — Results on Top 10, 20, and 30 Entity Features

Feature Set	PCC of each author							
	ClimateDesk	Climateprogress	ClimateReality	EarthVitalSigns	Greenpeace	UNEP	UNFCCC	Average
Base	0.312	0.353	0.599	0.445	**0.431**	0.459	0.495	0.442
Top 10	0.360	0.351	0.609	**0.491**	0.419	0.484	0.514	0.461
Top 20	**0.374**	0.352	**0.625**	0.483	0.396	**0.490**	0.543	**0.466**
Top 30	0.331	**0.360**	0.621	0.449	0.369	0.484	**0.556**	0.453

a. Number 10, 20, 30 in the feature set names mean selecting top 10, 20, 30 entities.

Table 7 — Top 30 Entities of Author Greenpeace

Method	Entities ordered by rank
DF	theguardian.com, greenpeace.org, #climatechange, oil, world, #thecrossing, shell, arctic, climate, change, savethearctic.org, years, stop, bbc.com, people, year, power, coal, #divest, air, energy, rig, #solar, make, #arctic, @shell, time, china, nationalgeographic.com, global
IDF	greenpeace.org, theguardian.com, oil, stop, #climatechange, shell, world, costa, rica, independent.co.uk, arctic, sea, years, electricity, powered, energy, year, #thecrossing, people, savethearctic.org, #arctic, change, time, #savethearctic, climate, plastic, theplaidzebra.com, generate, china, days
Sum	theguardian.com, greenpeace.org, oil, #climatechange, world, shell, stop, arctic, years, #thecrossing, year, savethearctic.org, sea, people, energy, change, climate, independent.co.uk, powered, costa, rica, electricity, #arctic, time, power, #savethearctic, china, bbc.com, make, air

Table 7 shows the top 30 entities of author Greenpeace, ranked by the DF, IDF, and Sum. The tweets analyzed are from the training set. As an example, the word “stop” is out of top 10 as frequency of being mentioned (DF), but it gets up to 7th in Sum and even 4th in IDF, which means it’s a word bringing more retweets in average, even Greenpeace doesn’t use it so many times. An advantage of the Entity feature is that it's a personalized feature set rather than feature benefiting the majority of authors, just like words in the table, all the features come from the Greenpeace's habit and its reader's feedback. For example, "rt" is the 2nd favorite word of author ClimateReality, and if we build a global learning model, "rt" is probably one of the features since ClimateReality posts more than 3 times tweets of Greenpeace, but Greenpeace doesn't like to impress readers in that way, so there's no need to use it as a feature for its own model.

4.6 Results on Pair Features

Table 8 — Results on Top 10 Pair Features

Feature Set	**PCC of each author**							
	ClimateDesk	**Climateprogress**	**ClimateReality**	**EarthVital Signs**	**Greenpeace**	**UNEP**	**UNFCCC**	**Average**
Base	0.312	0.353	0.599	0.445	0.431	0.459	0.495	0.442
AEMI*	0.332	**0.374**	0.612	**0.511**	0.459	0.465	0.494	**0.464**
DF	0.281	0.366	0.611	0.467	0.451	**0.465**	0.492	0.448
IDF	0.334	0.351	0.616	0.493	0.433	0.460	0.498	0.455
Jaccard	0.293	0.357	0.595	0.452	0.453	0.461	**0.500**	0.444
Sum	**0.344**	0.338	**0.617**	0.510	**0.464**	0.457	0.495	0.461

a. * indicates the feature set is significantly better than Base set based on a paired t-test with 95% confidence.

Table 8 is the experiment result on Base + top 10 Pair features, and all types of entities are used instead of only words. For AEMI and Jaccard, the probability is estimated with original tweets rather than retweets or web pages. AEMI has the highest average PCC among all other feature set, and AEMI's improvement on Base features is statistically significant. It could because when two entities have a high AEMI value, they share mutual information with each other, instead of just appear together by change. AEMI outperforms Jaccard on 6 out of 7 authors, and that means AEMI picks up more helpful pairs, so it might prove the words in AEMI pair are more related to each other than words in Jaccard pair.

The Pair features could have less chance of misleading the learner than Entity features. It's interesting that DF in Pair feature test doesn't have a terrible performance as in Entity feature test. A possible explanation could be that a pair has much fewer times of occurrence than a word, so it has less chance of being contained by most welcomed tweets at the same time, then the pair wouldn't be the only reason of high retweets, and the importance of other features could be deliberated by the learning algorithm.

Table 9 — Results on Pair Features of All Entity or Word

Feature Set	PCC of each author							
	ClimateDesk	Climateprogress	ClimateReality	EarthVitalSigns	Greenpeace	UNEP	UNFCCC	Average
Base	0.312	0.353	0.599	0.445	0.431	0.459	0.495	0.442
10 All*	0.332	**0.374**	0.612	**0.511**	0.459	0.465	0.494	0.464
10 Word*	0.335	0.363	0.605	0.463	**0.466**	**0.468**	**0.511**	0.459
20 All*	0.341	0.353	**0.615**	0.511	0.460	0.463	0.501	0.463
20 Word*	0.336	0.357	0.603	0.452	0.464	0.462	0.508	0.455
30 All*	**0.350**	0.361	0.612	0.507	0.461	0.465	0.500	**0.465**
30 Word	0.349	0.364	0.603	0.408	0.463	0.461	0.505	0.450

a. * indicates the feature set is significantly better than Base set based on a paired t-test with 95% confidence.

b. Number 10, 20, 30 in the feature set names mean selecting top 10, 20, 30 pairs.

c. All means selecting all types of entities, Word means selecting only words.

Table 9 shows the comparison between all entity and word only, using AEMI with probability estimated with only original tweets. The number in the row name is the number of pairs selected. Generally All is better than Word on average in every level of top pairs, and most of them have a statistically significant improvement comparing with Base features. It seems performance of All with AEMI is quite stable in different numbers of pairs, which means the additional pairs, which have less mutual similarity than the first 10 pairs, are sort of useless, they are neither helping the training nor causing the bad performance. Word has an obvious decreasing trend on author EarthVitalSigns, the one who has fewest tweets and of cause the fewer number of words, so after using up the valid word pairs, the words of additional pairs might exist by chance, as a result, learning algorithm could easily get misdirected by these “lucky” pairs.

Table 10 — Results on AEMI Using Tweet, Retweet or Tweet + Web

Feature Set	PCC of each author							
	ClimateDesk	Climateprogress	ClimateReality	EarthVitalSigns	Greenpeace	UNEP	UNFCCC	Average
Base	0.312	0.353	0.599	0.445	0.431	0.459	0.495	0.442
10 OT*	0.332	**0.374**	0.612	0.511	0.459	0.465	0.494	0.464
10 RT	0.358	0.352	**0.615**	**0.514**	0.458	0.463	0.497	**0.465**
10 Web	0.368	0.369	0.592	0.473	0.433	0.461	0.497	0.456
20 OT*	0.341	0.353	**0.615**	0.511	0.460	0.463	**0.501**	0.463
20 RT	0.312	0.364	0.610	0.486	0.458	0.459	0.495	0.455
20 Web	0.381	0.369	0.588	0.479	0.434	0.458	0.498	0.458
30 OT*	0.350	0.361	0.612	0.507	0.461	**0.465**	0.500	**0.465**
30 RT*	0.339	0.364	0.610	0.480	**0.461**	0.457	0.495	0.458
30 Web	**0.383**	0.364	0.587	0.485	0.430	0.458	0.498	0.458

a. * indicates the feature set is significantly better than Base set based on a paired t-test with 95% confidence.

b. Number 10, 20, 30 in the feature set names mean selecting top 10, 20, 30 pairs.

c. OT, RT, and Web mean using Tweet, Retweet and Tweet + Web to estimate probability of entities.

Table 10 is the comparison on calculating probability of AEMI by tweet (OT), retweet (RT), and tweet + web (Web). All the OT improvements are statistically significant. Web has a tough time on two key authors ClimateReality and Greenpeace, which could because top pairs picked up by Web hardly reveal the writing custom of the author and rarely include the types of entity other than word either.

Table 11 shows the top 10 pairs of entities for Greenpeace selected by the different methods. In the AEMI Web row, the pairs have no other entities but only words, so the other types of entity are overwhelmed by the web pages, the pairs in Web are mostly English phrases.

Table 11 — Top 10 Pairs of Author Greenpeace

Method	Pairs ordered by rank (word1-word2)
AEMI OT	#thecrossing-savethearctic.org, change-climate, barrier-reef, air-pollution, great-reef, oil-rig, barrier-great, arctic-savethearctic.org, fossil-fuels, arctic-shell
AEMI RT	costa-rica, independent.co.uk-rica, costa-independent.co.uk, #thecrossing-savethearctic.org, great-reef, barrier-reef, barrier-great, electricity-generate, change-climate, fossil-fuels
AEMI Web	change-climate, fossil-fuel, fossil-fuels, global-warming, make-time, future-make, people-time, part-time, report-year, year-years
DF	#thecrossing-savethearctic.org, change-climate, oil-rig, #divest-theguardian.com, barrier-reef, arctic-shell, arctic-savethearctic.org, air-pollution, #thecrossing-shell, savethearctic.org-shell
IDF	costa-rica, independent.co.uk-rica, costa-independent.co.uk, electricity-generate, generate-rica, generate-independent.co.uk, electricity-rica, electricity-independent.co.uk, december-rica, december-independent.co.uk,
Jaccard RT	emma-thompson, costa-rica, klein-naomi, barrier-reef, #ospar2015-ospar, barrier-great, rapidly-recalls, positive-prefab, lifetime-recalls, lifetime-rapidly,
Sum	costa-rica, independent.co.uk-rica, costa-independent.co.uk, #thecrossing-savethearctic.org, change-climate, barrier-reef, electricity-generate, great-reef, barrier-great, arctic-shell

4.7 Results on Cluster Features

Table 12 shows the performance of the Cluster features by selecting top 10, 20, and 30 clusters. Only words are clustered instead of entities. The similarity is calculated by AEMI, using 3 different material: only tweets (OT), retweets (RT), and tweets + web pages (Web). The result shows RT and Web are quite competitive and they both much better than OT. OT performs similar or even better than RT in the Pair feature test, but here, OT is much worse than RT. The potential reason could be that the amount of words considered by Pairs and Clusters are different. Pair features need only top 10 strongest links between words, so at most 20 words, and such a small amount of information can be provided by tweets which are a few and each of them contains a few words. While talk about Cluster features, they need much more mutual information between words to identify how closed they are among hundreds or thousands of words. So the lack of information of original tweets could lead to a problem, as mentioned in feature section, the similarity between the infrequent words varies in a wide range, which is difficult to compare with other similarities.

Table 12 — Results on Cluster Features using AEMI to Measure Similarity

Feature Set	PCC of each author							
	ClimateDesk	Climateprogress	ClimateReality	EarthVitalSigns	Greenpeace	UNEP	UNFCCC	Average
Base	0.312	0.353	0.599	0.445	0.431	0.459	0.495	0.442
10 OT	0.313	**0.389**	0.590	0.454	0.447	0.424	0.507	0.446
10 RT	0.319	0.385	0.596	0.561	**0.471**	0.441	**0.513**	0.469
10 Web	0.368	0.360	**0.613**	0.579	0.427	0.486	0.510	0.477
20 OT	0.369	0.377	0.586	0.476	0.447	0.410	0.509	0.453
20 RT	0.387	0.380	0.593	0.582	0.458	0.456	0.506	0.480
20 Web	0.373	0.362	0.605	0.590	0.418	**0.491**	0.511	0.478
30 OT	0.373	0.377	0.584	0.476	0.448	0.423	0.506	0.455
30 RT	**0.388**	0.375	0.589	0.577	0.449	0.467	0.499	0.478
30 Web	0.374	0.359	0.604	**0.598**	0.435	0.488	0.510	**0.481**

a. Number 10, 20, 30 in the feature set names mean selecting top 10, 20, 30 clusters.

b. OT, RT, and Web mean using Tweet, Retweet and Tweet + Web to estimate probability of entities.

It seems that Cluster features have better improvement while selecting more number of clusters. Comparing to Pair features, Cluster features have much higher result on the small author EarthVitalSigns, but doesn’t benefit a lot the largest author ClimateReality. The reason could be the author with fewer tweets has insufficient data in Base features for learning, and clusters as additional information can fill the gap; but for the authors with much information for Base features (the low performance caused by misleading words not by lack of data), it’s hard to get an improvement.

Table 13 — Results on Clustering All Entity or Word

Feature Set	PCC of each author							
	ClimateDesk	Climateprogress	ClimateReality	EarthVitalSigns	Greenpeace	UNEP	UNFCCC	Average
Base	0.312	0.353	0.599	0.445	0.431	0.459	0.495	0.442
10 All	0.344	**0.385**	**0.605**	0.412	0.438	0.441	0.511	0.448
10 Word	0.319	**0.385**	0.596	0.561	**0.471**	0.441	**0.513**	0.469
20 All	0.388	0.380	0.598	0.404	0.443	0.421	0.503	0.448
20 Word	0.387	0.380	0.593	**0.582**	0.458	0.456	0.506	**0.480**
30 All	0.381	0.375	0.600	0.404	0.433	0.446	0.493	0.447
30 Word	**0.388**	0.375	0.589	0.577	0.449	**0.467**	0.499	0.478

a. Number 10, 20, 30 in the feature set names mean selecting top 10, 20, 30 clusters.

b. All means selecting all types of entities, Word means selecting only words.

Table 13 is the comparison of using all types of entities or words to cluster, the similarity is AEMI of RT. It shows clustering all types of entities doesn't have better PCC on average. So it seems not a good idea to cluster hashtag, mention, and domain. It's interesting that in Pair features All outperforms Word, but it's the opposite in Cluster features. Other than EpSVR, not all the learning algorithms have obviously higher PCC on Cluster features than on Pair features, but all of them show the All is better than Word in Cluster features. If we check the results, the main improvement comes from EarthVitalSign, Greenpeace, and UNFCCC. A possible explanation could be some of the entities of these authors are only hot in a short time, so it's harmful, but in the Pair features only a few entities get selected, while in the Cluster features a great many of entities get enrolled, so the chance of getting hurt is much higher. There is an example to demonstrate it. Table 14 is the top 10 clusters of Greenpeace on AEMI, we can discover that the No. 1 cluster correctly represents the topic of #thecrossing, which is the project of Greenpeace to save the arctic and related to the company Shell. Unfortunately, as mentioned in result on Entity features section, the project has not been adverted any more in the test set, so if Greenpeace tweets the word "arctic" again, it could be only a photo of a white bear instead of the abstracted topic #thecrossing. But it's comforting to know that the AscendingCut algorithm does a great job on clustering, like in the table, the second cluster is obviously a topic of renewable energy, and most of words in the third cluster indicate a project of protecting the turtle about to extinct.

Table 14 — Top 10 Clusters of Author Greenpeace

No	Entities in each cluster
1	#arctic, #savethearctic, #thecrossing, arctic, drill, drilling, emma, follow, greenpeace.org, lifetime, melting, oil, rapidly, recalls, rig, savethearctic.org, shell, thompson, trip, youtube.com
2	burnt, costa, days, december, electricity, energy, farms, generate, homes, independent.co.uk, nuclear, plants, powered, produce, renewable, rica, solar, wind
3	#turtletortoisetuesday, #worldbookday, baby, century, changed, earth, extinction, galapagos, hatched, mass, nasa, nasa.gov, st, thedodo.com, time, tortoises, vox.com
4	#climate, alleviating, climatenewsnetwork.net, eco, ends, hunger, improving, landfill, moderating, nutrition, role, swedish, theplaidzebra.com, trash, tree, trees
5	#fracking, california, drought, hundreds, lion, lions, nytimes.com, pups, reuters.com, rising, sea, seas, starving, warming, water
6	amazonian, columbia, deforestation, dw.de, efeverde.com, end, global, join, km, movement, roads
7	australia, carbon, consumes, emissions, explore, generates, house, inhabitat.com, positive, prefab, shipping
8	barrier, coal, cyclones, fish, great, greenpeace.org.au, mine, newscientist.com, protect, protects, reef
9	#standforforests, boreal, defend, destruction, e-activist.com, forest, incredible, man, planted, stop
10	ably, animal, bear, china, cute, greenpeace.org.uk, nationalgeographic.com, polar, rediscovered, teddy

Table 15 shows the difference between calculating similarity with AEMI and Jaccard, the probability is using Web here. Basically AEMI is better than Jaccard on average for each number of clusters.

Table 15 — Results on Using AEMI or Jaccard to Measure Similarity

Feature Set	PCC of each author							
	ClimateDesk	Climateprogress	ClimateReality	EarthVitalSigns	Greenpeace	UNEP	UNFCCC	Average
Base	0.312	0.353	0.599	0.445	0.431	0.459	0.495	0.442
10 AEMI	0.368	0.360	**0.613**	0.579	0.427	0.486	0.510	0.477
10 Jaccard	0.328	0.367	0.592	0.572	0.401	0.449	0.494	0.458
20 AEMI	0.373	0.362	0.605	0.590	0.418	**0.491**	**0.511**	0.478
20 Jaccard	0.374	0.366	0.582	0.576	0.394	0.455	0.495	0.463
30 AEMI	0.374	0.359	0.604	**0.598**	**0.435**	0.488	0.510	**0.481**
30 Jaccard	**0.377**	**0.369**	0.588	0.561	0.392	0.449	0.499	0.462

a. Number 10, 20, 30 in the feature set names mean selecting top 10, 20, 30 clusters.

4.8 Results on Combination of All Features

The Table 16 is result on the combinations of Entity, Pair, and Cluster features. The EPC in the table is the model using all types of features together, and others are the combinations of 2 types of features. The number is for the number of top items, for example, EPC 10 means selecting top 10 entities, top 10 pairs, and top 10 clusters as additional features. The experimental setting is the best configurations of all previous tests, which is: Entity, all types of entities and IDF; Pair, all types of entities and AEMI with OT; Cluster, only words and AEMI with Web. Every feature set has a statistically significant improvement except EC. The first observation is that the all combination EPC is better than any type of feature alone in the previous results. The second point we can find is that, surprisingly, combining only Pair and Cluster feature is even better than combining all, that means the Entity feature can be easily effected by the problem of the distribution difference between the training and test sets, mainly for the author Greenpeace and EarthVitalSigns. The third observation of the result is the combination of Entity and Pair has the lowest average (except the Base), which implies that Cluster feature is the most significant one out of 3 types. And the fourth conclusion is there is not a setting to have the best performance of all the authors, different author prefers different features. For instance, Climateprogress has high performance when there is no Entity feature; Greenpeace is quite misled by the Entity and Cluster, and UNEP can benefit from the combination of all features.

Table 16 — Results on Combinations of All Types of Features

Feature Set	PCC of each author							
	ClimateDesk	Climateprogress	ClimateReality	EarthVitalSigns	Greenpeace	UNEP	UNFCCC	Average
Base	0.312	0.353	0.599	0.445	0.431	0.459	0.495	0.442
EPC 10*	0.377	0.356	0.629	0.516	0.442	0.503	0.524	0.478
EPC 20*	0.412	0.355	**0.632**	0.524	0.425	**0.515**	0.549	0.487
EPC 30*	0.375	0.370	0.627	0.507	0.423	0.507	0.557	0.481
PC 10*	0.375	**0.378**	0.630	0.586	**0.460**	0.489	0.510	**0.490**
PC 20*	0.381	0.363	0.624	0.580	0.451	0.492	0.515	0.487
PC 30*	0.390	0.365	0.622	**0.591**	0.451	0.491	0.511	0.489
EC 10	0.399	0.352	0.622	0.520	0.396	0.497	0.527	0.473
EC 20	**0.415**	0.354	0.628	0.517	0.381	0.509	0.550	0.479
EC 30	0.367	0.369	0.625	0.510	0.384	0.507	**0.563**	0.475
EP 10*	0.346	0.357	0.616	0.471	0.448	0.487	0.507	0.461
EP 20*	0.369	0.351	0.627	0.491	0.432	0.496	0.544	0.473
EP 30*	0.343	0.368	0.621	0.474	0.421	0.492	0.558	0.468

a. * indicates the feature set is significantly better than Base set based on a paired t-test with 95% confidence.

b. E stands for Entity feature, P stands for Pair feature, and C stands for Cluster feature. EPC means combining Entity, Pair and Cluster features, PC is combining Pair and Cluster features, and so on.

c. Number 10 in EPC 10 means selecting top 10 entities, top 10 pairs, and top 10 clusters, other numbers are similar.

Table 17 is the comparison of all learning algorithms based on additional features. The number 1 and 2 in the algorithm name separately represent combining all features together and combining Pair and Cluster features. The setting of the test is the same with Table 16, and additional features select top 10 items. Firstly, comparing to using only Base features (Table IV), basically all learning algorithms have significantly better result on the additional features except ANN. Secondly, only combining Pair and Cluster feature seem to be a good choice for all learners. Thirdly, EpSVR is still the best learner, while NuSVR performs pretty well too.

Table 17 — Results on All Learners Using Additional Feature

Algorithms	PCC of each author							
	ClimateDesk	Climateprogress	ClimateReality	EarthVital Signs	Green peace	UNEP	UNFCCC	Average
LR1	0.363	0.337	0.623	0.548	0.434	0.474	0.481	0.466
ANN1	0.245	0.330	0.595	0.014	0.396	0.480	0.456	0.359
EpSVR1*	0.377	0.356	0.629	0.516	0.442	0.503	**0.524**	0.478
NuSVR1*	0.410	0.343	0.631	0.533	0.418	**0.506**	0.523	0.480
LR2*	**0.418**	0.375	0.625	0.477	0.442	0.464	0.491	0.470
ANN2	0.327	**0.392**	**0.635**	0.336	0.397	0.450	0.421	0.422
EpSVR2*	0.375	0.378	0.630	**0.586**	**0.460**	0.489	0.510	**0.490**
NuSVR2*	0.382	0.379	0.630	0.556	0.437	0.493	0.508	0.483

a. * indicates the result of the learner is significantly better than the corresponding result using only Base features (Table 4) based on a paired t-test with 95% confidence.

b. Number in the algorithm name: 1 stands for combining Entity, Pair and Cluster features, 2 stands for combining Pair and Cluster features.

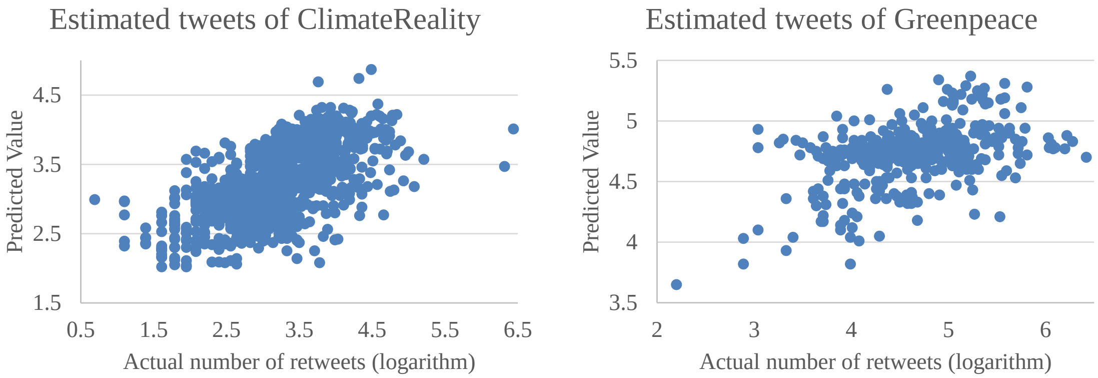

Figure 2 — Visualization of Actual Number of Retweets and Predicted Value for Each Tweet.

Figure 2 shows the correlation between actual number of retweets and the predicted value of each tweet from author ClimateReality and Greenpeace. Each point in the figure is a tweet in the test set, the x value is the number of retweets of it after logarithm, and the y value of the point is the effectiveness of the tweet estimated by the learning model. The figure shows the result of the PC 10 of Table 16, in which the correlation of ClimateReality is 0.63 and Greenpeace is 0.46. A perfect figure of PCC (value of 1.0) should look like an increasing straight line. Although the result of ClimateReality is not perfect, there is still an obvious trend in the figure which is the more number of retweets is, the higher estimated value is. Some of the tweets get clearly over-estimated at the top left corner in the Greenpeace's graph, which could be why Greenpeace has a lower PCC than ClimateReality.

Chapter 5
Conclusion

We propose a process to estimate the effectiveness of the tweet of the author by using machine learning algorithm to build model based on features extracted from historical tweets. We extract base features and 3 types of additional features from the tweets. In the additional features, the Entity features are the effective words, hashtags, mentions, and domains, and we presented 3 methods: DF, Sum and IDF to select them. The IDF has the best performance, and the reason could be that entities selected by IDF are supposed to lead to the high average of number of retweets. The Pair features have entity pairs appearing in the tweets, and beside the above 3 ways, there are 2 further methods AEMI and Jaccard to select them. The experiment shows AEMI performs better than others, and it could because that AEMI selects entities related to each other. The Cluster features extract topics from tweets, and we proposed the AscendingCut algorithm to cluster words based on the similarity between them. The AscendingCut has meaningful clusters result, and the Cluster features benefit the learning algorithm the most among the 3 types of features.

From the experimental results on 7 Twitter authors, the Pearson Correlation Coefficient shows that by using Pair features with AEMI to select pairs, the prediction is significantly more correlated to the actual number of retweets than using only base features. Combining all types of features together helps the learning even more than applying single type. Many combinations have the statistically significant improvement comparing with Base features. Combining only Pair and Cluster features has the best performance because the Entity feature is more likely to suffer from the problem that the author discusses a hot topic only for a short time period, which leads to a tough job for learning algorithm because the data distribution of the training set is different from the test set. Although the Twitter accounts researched in this paper are all organizations related to climate change advocacy, the process of feature extraction and learner training is not limited to the climate change authors, so this work should be used to analyze any author in Twitter.

There could be a further improvement if the temporary hot words can be isolated or even utilized as features. The possible solution might be checking recent hot words or distribution of words. Secondly, the way of selecting top clusters can also be different from just selecting the largest ones, the cluster has more inner similarity could be more representative. Finally the features now only utilizes the original tweets from the author, while the tweets retweeted by the author probably have a great amount of help as well, because those are the tweets interested by the author so they should related to the same topics of the original tweets.

References

[1] M. M. Bradley and P. J. Lang. Affective norms for English words (ANEW): Instruction manual and affective ratings. Technical report, The Center for Research in Psychophysiology, University of Florida, 1999.

[2] K. El-Arini, U. Paquet, R. Herbrich, J. V. Gael, and B. A. Arcas. "Transparent user models for personalization." In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 678-686. ACM, 2012.

[3] W. Feng, and J. Wang. "Retweet or not?: personalized tweet re-ranking." In Proceedings of the sixth ACM international conference on Web search and data mining, pp. 577-586. ACM, 2013.

[4] W. Hwang, T. Kim, M. Ramanathan, and A. Zhang. "Bridging centrality: graph mining from element level to group level." In Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 336-344. ACM, 2008.

[5] M. Jenders, G. Kasneci, and F. Naumann. "Analyzing and predicting viral tweets." In Proceedings of the 22nd international conference on World Wide Web companion, pp. 657-664. International World Wide Web Conferences Steering Committee, 2013.

[6] H. R. Kim, and P. K. Chan. "Learning implicit user interest hierarchy for context in personalization." In Proceedings of the 8th international conference on Intelligent user interfaces, pp. 101-108. ACM, 2003.

[7] H. Kwak, C. Lee, H. Park, and S. Moon. "What is Twitter, a social network or a news media?." In Proceedings of the 19th international conference on World wide web, pp. 591-600. ACM, 2010.

[8] K. Lee, J. Mahmud, J. Chen, M. Zhou, and J. Nichols. "Who will retweet thisfi." In Proceedings of the 19th international conference on Intelligent User Interfaces, pp. 247-256. ACM, 2014.

[9] S. A. Macskassy, and M. Michelson. "Why do people retweet? anti-homophily wins the day!." In ICWSM. 2011.

[10] N. Naveed, T. Gottron, J. Kunegis, and A. C. Alhadi, "Bad news travel fast: A content-based analysis of interestingness on twitter." In Proceedings of the 3rd International Web Science Conference, p. 8. ACM, 2011.

[11] J. W. Pennebaker, M. E. Francis, and R. J. Booth. "Linguistic inquiry and word count: LIWC 2001." Mahway: Lawrence Erlbaum Associates 71 (2001): 2001.

[12] D. Quercia, J. Ellis, L. Capra, and J. Crowcroft. "In the mood for being influential on twitter." In Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on, pp. 307-314. IEEE, 2011.

[13] B. Suh, H. Lichan, P. Pirolli, and E. H. Chi, "Want to be Retweeted? Large Scale Analytics on Factors Impacting Retweet in Twitter Network," Social Computing (SocialCom), 2010 IEEE Second International Conference on , vol., no., pp.177,184, 20-22 Aug. 2010

[14] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and A. Kappas. "Sentiment strength detection in short informal text." Journal of the American Society for Information Science and Technology 61, no. 12 (2010): 2544-2558.

[15] I. Uysal, and W. B. Croft. "User oriented tweet ranking: a filtering approach to microblogs." In Proceedings of the 20th ACM international conference on Information and knowledge management, pp. 2261-2264. ACM, 2011.

[16] Z. Xu, and Q. Yang. "Analyzing user retweet behavior on twitter." In Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012), pp. 46-50. IEEE Computer Society, 2012.