Relational Term Weighing Metric For Event Detection

A B.Tech Dissertation report submitted in fulfilment of the requirements for the degree of Bachelor of Technology

by

BHAWIKA AGARWAL(2015UCP1559),
SAKSHI YADAV (2015UCP1683),
AMIT KUMAR YADAV (2015UCP1549)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
MALAVIYA NATIONAL INSTITUTE OF TECHNOLOGY JAIPUR

May 2019

Certificate

We,

BHAWIKA AGARWAL(2015UCP1559),

SAKSHI YADAV (2015UCP1683),

AMIT KUMAR YADAV (2015UCP1549),

Declare that this thesis titled, "Relational Term Weighing Metric For Event Detection" and the work presented in it are our own. I confirm that:

- This project work was done wholly or mainly while in candidature for a B.Tech. degree in the department of computer science and engineering at Malaviya National Institute of Technology Jaipur (MNIT).
- Where any part of this thesis has previously been submitted for a degree or any other qualification at MNIT or any other institution, this has been clearly stated. Where we have consulted the published work of others, this is always clearly attributed.Where we have quoted from the work of others, the source is always given. With the exception of such quotations, this Dissertation is entirely our own work.
- We have acknowledged all main sources of help.

Signed:

Date:

Dr. Arka Prokash Mazumdar

Assistant Professor

Department of Computer Science & Engineering

Malaviya National Institute of Technology Jaipur

May 2019

Abstract

Name of the student: **BHAWIKA AGARWAL(2015UCP1559),**
SAKSHI YADAV (2015UCP1683),
AMIT KUMAR YADAV (2015UCP1549)
Degree for which submitted: **B.Tech.** Department: **Computer Science and Engineering**
Thesis title: **Relational Term Weighing Metric For Event Detection**
Thesis supervisor: **Dr. Arka Prokash Mazumdar**
Month and year of thesis submission: **May 2019**

Microblogs have become the new communication medium between users. It allows millions of users to post and share content of their own activities, their opinions about different topics. As a result, there is an urgent need to detect events from microblogs so that users can identify events quickly, also and more importantly to aid higher authorities to respond faster to occurring events by taking proper actions. To compute similarity between microblog posts, most existing approaches consider each term independently.

In this paper, we propose a co-occurrence based term-weighing algorithm making use of bloom filters, which exploits the relationship between co-occurring words and results are compared against tf-idf and tf-igm based approaches. These weighing metrics are used in an online incremental clustering framework to cluster the posts based on similarity. We used precision, recall and f-measure as evaluation metrics. Our results were significantly better compared to the popular tf-idf approach and tf-igm approach applied to the same dataset.

Acknowledgements

We are profoundly grateful to **Dr. Arka Prokash Mazumdar** for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion.

We would like to express deepest appreciation towards **Dr. Girdhari Singh**, Head of Department of Computer Science and Engineering and **Dr. Ramesh B Battula**, Project Coordinator whose invaluable guidance supported us in completing this project.

At last we must express our sincere heartfelt gratitude to all the staff members of Computer Science and Engineering Department who helped me directly or indirectly during this course of work.

BHAWIKA AGARWAL
SAKSHI YADAV
AMIT KUMAR YADAV

Contents

List of Figures

List of Tables

Abbreviations

TF	**T**erm **F**requency
IDF	**I**nverse **D**ocument **F**requency
IGM	**I**nverse **G**ravity **M**oment
ACR	**A**verage **C**o-occurrence **R**epetition

Chapter 1

Introduction

This chapter is an introduction to the thesis, first it gives a brief description of event detection and then a brief description about approaches of event detection. In addition, it states the problem the scope and limitation of the thesis work, and the research methodology.

1.1 Overview

Nowadays, Microblogs have become a new platform for connecting people and sharing digital content by allowing users to post and share short text, images, short length videos. This content is delivered to a network of followers with very less delay. Content type depends on the interests of the users and the situation.

With the instant delivery, events news usually spread faster and reach wider audience in microblogs compared to mainstream media. Events are real-world occurrences that take place in a certain geographical location over a certain time period (Allan, 2002). Capturing information about an event can help in many aspects. For example, it can help in accelerating the crisis response when the information about disastrous events are retrieved at the time of its occurrence. Also it can help people to easily track occurring events. Other applications of event detection is the

Facebook Safety Check in which users confirm they are safe when a natural or a man-made disaster occurs near them. Capturing event details from microblogs is not an easy task, because events information are covered with a huge amount of unnecessary data such as random topics, users daily activities, spam or any kind of data that are not related to an event.

Event detection problem has been addressed by many researchers; they employed different techniques from many fields . The first event detection research program is the Topic Detection and Tracking (TDT) conducted by (Allan, 2002). The techniques introduced are meant to monitor different newswire sources so that users can be aware about occurring events. The approach was applied on full text of well written news articles, however, with the emergence of microblogs, new challenges are introduced when using these techniques. For example, content generated by microblogs users is constrained to be very small, thus if the traditional Term Frequency-Inverse Document Frequency (TF-IDF) has been used in such short documents it will result in a sparse vector issue. Also microblogs posts are noisy. According to Hurlock and Wilson (2011) posts in microblogs do not always refer to an actual event or a subject of importance, but most of the time the posts contain meaningless or uninteresting daily life activity.

To be able to identify these events in real time on microblogs is difficult, because of the immense scale of data, and heterogenity of data. Microblog users post updates with a wide variety of content types, including personal information and other worldly happenings. Also, microblog posts, are very short and have large number of grammatical errors and informal language, making the task more difficult. Figure 1.1 shows an example of such problems. An efficient event detection algorithm should be able to tackle these problems efficiently and effectively.

Many existing approaches of calculating term weight metrics for calculating similarity between posts, to cluster them and eventually use these clusters for further classification and analysis for the task of event detection. But these approaches assume that the words provide independent contribution to the similarity, and no

FIGURE 1.1: Example tweet that contains informal words.

association between co-occurring words is considered. Our proposed approach exploits this co-occurrence of words, and describes a weighing scheme using the same, and results are evaluated against some of the other existing term-weighing metrics.

1.2 Problem Statement

To propose a new weight metric that can be used to compute similarity amongst microblog posts, in an online incremental clustering framework for real-time results and evaluating its performance (precision, recall, F-Measure) against existing such metrics.

1.3 Scope and limitation

We aim to formulate a term weighing metric to cluster similar posts from microblogs that adhere to the following limitations and assumptions:

1. We will use a Twitter based dataset.

2. Our approach will rely on the contextual information only, and the structural information is not used.

1.4 Research Methodology

The following is the approach we used in this thesis:
Research and survey: We reviewing the recent literature related to the thesis problem statement. We analyze the existing methods and approaches, and identify their drawbacks and the lack of existing approaches. We then propose a solution to overcome these drawbacks.
Building the Approach: Our proposed approach include the following general steps:

1. **Data Collection:** extract tweets from the internet using Twitter API
2. **Data Preprocessing:** prepare the data and improve its quality by following steps:
    - Tokenization
    - Stopword removal
    - URL removal, and cleaning
    - Part-of-Speech (POS) tagging and Lemmatization
3. **Defining the metric** we define the weighing metric for calculating similarity, and use it for clustering.
4. **Evaluation:** we evaluate our approach using precision, recall, and F-measure, and compare with other existing approaches.
5. **Results and discussions:** we analyze the results and justify them.

1.5 Thesis Outline

The rest of the thesis is organised as follows : Chapter 2 presents the literature review about event detection in general and various weight metrics and techniques employed. Chapter 3 discusses two of the the popular existing approaches against which our proposed approach is compared. Chapter 4 describes the proposed co-occurrence based approach and its implementation. Chapter 5 presents the experiments and evaluations of the results. At last, chapter 6 concludes the thesis and suggests the future work.

Chapter 2

Literature Survey

In this chapter we introduce the literature review related to our work. We review various similarity metric used for event detection, also we review approaches that are appropriate for event detection on micro-blogging sites like Twitter.

2.1 Overview

Many approaches of event detection in micro-blogs uses platform specific features which make the approach accuracy dependent on these features and cannot be ported to other platforms. Also, other approaches focus on finding a burst of words in the stream to identify hot topics. These approaches will detect a topic without any consideration if that topic is an actual event or a viral general topic. In addition to that, most of the research have been done and tested on individual word weight for calculation of similarity but taking the co-occurrence of the words could give better results. In the following sections we review some of these researches.

2.2 Platform Specific Approaches

Phuvipadawat and Murata [1] (2010) introduced an approach that tracks and ranks breaking news from Twitter stream. Tweets are retrieved using predefined search queries and indexed using Apache Lucene. Term Frequency-Inverse Document Frequency (TF-IDF) is used for tweet representation with a bias factor for named entities, hashtags, and usernames. Additional weight is added to the tweet based on a set of features such as the number of followers which represents reliability and the number of retweeted messages which represents popularity within a . A tweet is assigned to a cluster if it is similar to the first tweet added to that cluster and similar to the top K terms in the cluster. The authors claim that adding more weight to Named Entities, Hashtags, and Usernames produces betters results.

Becker, Naaman, and Gravano [2] (2011) proposed an approach to detect real-world events from Twitter with avoiding trendy topics. They used incremental online clustering algorithm to cluster similar tweets without specifying the number of clusters. A set of features are extracted from each clusters to classify event and nonevent clusters. These features are temporal features which represent the most frequent terms in the cluster. Social features which are represented by the percentage of messages containing social interaction such as retweet, reply, and mentions. Topical features that represent the central topic of a cluster, since the authors assume that clusters containing different topics tend to be non-event clusters, and clusters that contains central theme is likely to be an event cluster. In addition to the previous features, twitter hashtag feature is also used. The authors claim that clusters with hashtags of more than one concatenated words are likely to be a general discussion and not real-world event. Event clusters are identified by the classifier. Top K clusters are selected using the technique of selection introduced by Petrović, Osborne, and Lavrenko [3]. A manually collected dataset from Twitter is used for evaluation. Human annotators were used to label the resulted clusters using a subset of tweets in each cluster that belong to the top 10 frequent terms. A set of 100 randomly selected clusters were used to calculate the F-Measure of the classifier which

gives 0.837. Precision@K and the Normalized Discounted Cumulative Gain (NDCG) evaluation metrics are also used.

Weng and Lee [4] (2011) gave an event detection technique based on clustering of discrete wavelet signals. These signals were created using individual words that were produced by Twitter. Wavelet transformation have a special feature that they are localized in both frequency domain and time domain unlike fourier transforms, which were used for event detection with traditional media. This allows wavelet transformations to determine duration and time of a bursty event within the signal. The construction of a signal is done on the basis of a time dependent variant of the document frequency – inverse document frequency (DF-IDF). Here, DF will count the no. of documents (that is tweets in our case) that has a specific word, and idf counts frequency of a word till current time step. After this, we apply a sliding window that aims at capturing any change that occurs over time by making use of H-measure which is normalized wavelet entropy. Then, we filter out trivial words according to set threshold of signals cross-correlation. It is used to find out two signals' similarity as a function of lag in time. The next step involves clustering the rest of the words so as to form events. A modularity-based graph partitioning method is used for this. For the last step, they detect significant events on the basis of no. of words and cross-correlation between the words which are same event related.

Cordeiro [5] (2012) proposed a lightweight approach based on continuous wavelet transformation analysis of Twitter hashtag occurrences. Peaks and local maxima are calculated to find hashtags with higher signal which indicate an event. Latent Dirichlet Allocation (LDA) is then used on the collection of tweets that belongs to the hashtags to create topic inference model. We do not predict user behavior, thus depending on hashtags only as an indicator of an event is not always accurate, as users may not attach a hashtag when tweeting about an event. Also hashtags could be the presence of spam tweets.

Li Rui et al. [6] (2012) proposed an approach based on Twitter specific features

that uses various features of twitter like a hashtag (e.g., "#DACA"), that indicates a topic, a short URL (e.g., "http://ow.ly/hKTe50tpaR3"), or an "@", which means a reply. These features are helpful in specifying if a tweet is event-related tweet or not. Taking an example, suppose a tweet has a link that points to a new page, it is likely to be an event-based tweet and we can use the page content as additional text. The various information given in the tweet and some other features like a GPS tag, or location of the creator of a tweet so as to predict a tweet's location. However not all users' have locations in their profile. The author claims that with the model is able to achieve 63% accuracy in predicting the location of a user by using his tweets and friends.

Li, Sun and Datta [7] (2012) gave a segment-based event detection approach for tweets, called Twevent. It first includes detection of event segments(bursty tweet segments) and then it uses both its frequency distribution and the similarity in their content to cluster these segments into corresponding events. It uses tweet segment rather than a unigram for event detection. Twevent also utilizes external knowledge base to guide the process. Tweet segmentation used here is able to reduce noise by identifying informative phrases. Twevent is pretty robust to tweet spam, and self promotion, owing to its use of user frequency for the purpose of extracting bursty event segments. Twevent was compared with other methods using a dataset of 4.3 million tweets that was published in June 2010 by users based in Singapore. Experiments also show that Twevent outperforms majority of these methods in terms of both precision and recall by quite large margins. Twevent makes it possible to interpret the events detected with very little background information.Efficiency and scalability of Twevent is also achieved.

Guille and Favre [8] (2014) an approach is proposed based on the anomaly of dynamic link (mentions) creation frequencies. A normal distribution is created for each word that co-occur with a mention. The anomaly of a word is detected in an interval of time when the distribution value of this word exceeded the expectation (mean). The magnitude of the word is also calculated by finding the algebraic area of the distribution at that interval. The word is identified as an event in an interval when

it reaches the highest magnitude. To retrieve the most K words that co-occur with the event word, the similarity between their temporal dynamics and the event word temporal dynamic is calculated and compared with a threshold value. The approach was evaluated over English and French datasets. Parameters of the approach were set as 30 minutes for window size, K=10, and a similarity threshold value of 0.7. The evaluation was conducted manually using human annotators. Precision and F-score were used as measurements. With the English corpus the results were 0.775 and 0.682 for precision and f-score respectively. With the French corpus the results were 0.825 and 0.825 for precision and f-score respectively. Despite the approach does not rely on external knowledge, it depends on Twitter specific features (mentions), also the approach performs poorly when mentions are not considered.

Dong, Mavroeidis, Calabrese and Frossard [9] (015) proposed a multi scale event detection system using social media data. It considers varying temporal and spatial scales of events in data. Wavelet transform properties, and a multiscale transform in signal processing was explored to enable automatic handling of the interaction between temporal and spatial scales. An algorithm to create a data similarity graph at appropriate scales was also proposed. Another objective was to be able to identify events of varying scales simultaneously by making use of only a single graph-based clustering technique. A spatio-temporal analysis on the noisy information present in a data stream was performed to be able to give a term filtering technique for the proposed algorithm and using simulated noisy data they could study it's behavior.The paper claimed that this framework can be further extended to various other application domains involving analysis of data both at multi-scale and multi-resolution level. The experimental results showed the approach's effectiveness.

eDoulamis et al. [10] (2016) proposed an approach based on a multiple assignment graph partitioning algorithm where event is represented by a cluster of related words. The authors addresses the problem of message posting delays which lead to event attributes being scattered in different timestamps, thus the significance of event-related words will be decreased as time goes on. Words are modeled using three

Twitter-based information theoretic metrics. Conditional Word Tweet Frequency-Inverse Trend Word Tweet Frequency (CWTF-ITWTF), which is a time varying measurement similar to the popular IDF-TF. The objective of this measurement is to decrease the weight of trendy ongoing events. Word Frequency-Inverse Trend Word Frequency (WF-ITW), which is a time varying measure that consider the frequency of keywords. Lastly, Weighted Conditional Word Tweet Frequency-Inverse Trend Weighted Conditional Word Tweet Frequency (WCWTF-ITWCWTF). This measure depends on features from Twitter such as no. of followers, no. of retweets in order to find the importance of a keyword. A fuzzy time series signal is produced from the three metrics. The approach is evaluated over a manually collected dataset using Twitter stream API. A time window of size 6 hours is used over a one month time horizon. To create a ground truth, for each time window the most frequent keywords are extracted and presented to experts to annotate them. The evaluation measurements used are keyword recall/precision and F1-Score. This approach cannot be used in microblogging streams that do not produce these features, however, our approach does not consider any Twitter based features. It also depends only on the bursty pattern of a set of keywords thus the resulted detected event can be a trending topic and not a real-world event. The approach require different parameter tuning to achieve good F1-Score.

Yilmaz and Hero [11] (2016) proposed an approach for event detection using multimodal factor analysis model. The approach depends on two features set. The hashtags' bag of words created from all the tweets containing the hashtag. Also, geolocation vector containing the latitude and longitude values of all tweets containing the hashtag. A probabilistic generative modal is used to fuse these features and after that an expectation maximization algorithm is proposed for finding the MLE estimates of the model parameters. The approach assumes that hashtags are used during event occurrence and tweets containing these hashtags are geographically closed together.

Hasan, Orgun and Schwitter [12] (2016) in their paper, proposed TwitterNews+, which is an event detection scheme that uses an incremental clustering technique,

and uses specialized inverted indices to detect major as well as minor events that are newsworthy in real-time from an incoming twitter stream. An extensive parameter sensitivity analysis was also conducted by them in order to fine tune the parameters that were used in TwitterNews+ to achieve increased performance.Various experimental results show that TwitterNews+ was able to detect the maximum no. of ground truth events (30 events out of 31) when evaluated against other existing different event detection systems. It resulted in a recall of 0.96. It was also able to achieve the highest precision of 0.89.

Nguyena and Jung [13] (2016) proposed a Twitter based real time event detection method where a frequency-based analytics was used to show the improved performance of method by using social streams and it was also shown that online behavioral analysis on multiple users can be applied with the help of big social data.User behavior is basically a stream of social data, with real time characteristics.It constitutes of notifications related to friends' posts after suffering a short delay for diffusion over network. The data stream may have news related to real social facts or unfocused information. Before performing event detection, textual dataset needs to be transformed into dataset of discrete signals. It makes use of common user behavior features during news exchange over internet by trying to combine it with a transformation method.This helps in reducing the data complexity with experimental results showing that the approach shows very high accuracy levels and high efficiency levels in being able to track abnormal phenomena.

2.3 Bursty-Words Based Approaches

These approaches depends on the presence of words that create bursts or spikes in the words frequency distribution and we use it to detect the trending or the hot topic in a time-frame.

Petrović et al. [3] (2010) presented a first story detection (FSD) in which the first text of a detected topic or event is extracted. A local sensitive hashing (LSH) is

used to reduce the search space, thus similarity measure is calculated between a tweet and a subset of neighbors identified by LSH. If LSH fails to find a neighbor, a limited search is performed considering only 2000 tweets. Threads that grow faster are identified as events. A preprocessing step is conducted that removes non English characters and Twitter-specific features such as mentions and hashtags. A manual evaluation procedure was used to calculate precision. Limiting the search space could boost the performance, however, in twitter stream, events features are very likely to be scattered over time (Doulamis et al., 2016), thus this approach will fail to detect events in this scenario. In our work, we used a greedy search approach considering only important words.

Weng and Lee [4] (2011) proposed an approach that depends on clustering wavelet signals. A signal is built for each word using wavelet analysis to reduce space and storage. Auto-correlation is calculated for each signal. Signals that produces skewed auto-correlation are identified as insignificant words and then removed. Similarity between words is calculated using the cross-correlation between every pair of words. Similar words are determined using a threshold value where words that have a similarity higher than the threshold are clustered together. The approach is evaluated over a manually collected dataset from Twitter where non-English characters are removed from the text. The evaluation is conducted manually considering only precision, as the authors cannot enumerate all events that occurred at the time of collection, thus recall is discarded. Different experiments with different configuration are used and the best result achieved was 16.7%. Clustering based on only a pair of words will produce generic events or topics (Petkos, Papadopoulos, Aiello, Skraba, & Kompatsiaris, 2014). For example, if a bombing event occurred in the same country with different locations, then using such algorithm will detect event that does not differentiate between the two different locations.

Gaglio, Re, and Morana [14] (2015) proposed an enhancement for SFPM that tackle the limitation of the selection process. The new method uses dynamic reference corpus. Also it depends on the product of a combination of measures such as the TF-IDF of the term, a bias factor used if the term is a Named Entity, and the

likelihood measure. Terms with higher values are selected. Both approaches require identifying the number of top terms with the presence of reference corpus. However, our approach identifies event triggers as important words, and these are extracted using syntactical analysis only.

Alkhamees and Fasli [15] (2016) proposed an approach based on the traditional FPGrowth method. FP-Growth produces the most frequently used combinations of words that co-occur together in a tweet. Determining what is most frequent depends on a fixed threshold value. On the other hand, the approach introduces a dynamic procedure for calculating the threshold, so that it can handle the dynamic nature of words size over time. The procedure depends on a combination of statistical values which are the average and median of the words' frequencies. A preprocessing step is performed for text tokenization and removing stop words, mentions, URLs, and hashtags. A post processing step is also performed to eliminate duplicate patterns. Duplicate patterns are determined by calculating the cosine similarity between patterns with a threshold of 0.75. The approach was tested using two datasets manually collected by querying the Twitter stream API using a set of keywords that identify two event, the UK General Elections 2015 and the Greece Crisis 2015.

Katragadda, Benton, and Raghavan [16] (2017) introduced a multiple source approach that collect data from twitter stream and newswire websites. Every source is considered as an independent stream. Every stream undergoes two stages. First a weighted graph is built in which nodes represent words and edges represent the number of documents in which the two connected words co-occur together. A pruning process is conducted on the graph to keep emerging and important words. The multiple sources are merged by merging the pruned graph. Events are detected using voltage based clustering algorithm on the resulted graph. The approach was tested using two sources Twitter and Tumblr and achieved F-Measure of 0.897.

2.4 Other Approaches

Chen K. et al. [17] (2016) proposed a new term weighting scheme called TF-IGM (term frequency and inverse gravity moment). In this, a new measure was developed that assigned weights to term on the basis of its class distinguishing power. Igm model was combined with the TF component, to finally propose a term-weight metric tf-igm. Experiments show that it performed significantly better than other similar approaches for text classification on three commonly used benchmark datasets.

2.5 Summary

In this chapter we presented a review of some related works in the field of event detection, and identified various existing approaches. We also showed the lack of research in using co-occurrences of words to calculate term weights and in clustering process of event detection.

Chapter 3

Existing Approaches

This chapter discusses the already existing tf-idf based and tf-igm based weighing schemes, and their limitations.

3.1 Term Frequency-Inverse Document Frequency Based Metric

In Becker et al. (2011) paper[2], tf-idf value is used as the similarity(weight) metric. It is the most basic and widely accepted similarity metric.It tells us the importance of a word in a document. The tf–idf weight value will increase as we increase the term frequency in the document and will start to decrease with increase in its occurrences in a large number of documents in a collection.

Here, assuming a real world setting of twitter messages, the dataset is read tweet by tweet, and current tweet is represented in the form of a tf-idf based vector. Tf is calculated for each word as its frequency of appearance in tweet, and idf is calculated as the no. of clusters the term appears in to calculate its rarity. Tf-Idf vector is represented as follows:

$$tf(term) = 1 + \log X \tag{3.1}$$

$$idf(term) = \log \frac{N}{M} \tag{3.2}$$

where X is the the term frequency in current tweet, N represents total number of clusters, and M is the total number of clusters in which the term occurs.

$$\textit{tf-idf(term)} = tf(term) * idf(term) \tag{3.3}$$

Cosine similarity is used to find out how similar the documents are irrespective of their size. Cosine similarity between a cluster tf-idf weight vector and a tweet tf-idf weight vector is computed.

$$\textit{Cosine-similarity(d1,d2)} = \frac{(d1.d2)}{|d1| * |d2|} \tag{3.4}$$

Owing to the online setting of our problem, a clustering algorithm was needed that could be used with large dataset, and is scalable. It must not require beforehand, the no of clusters. Thus, an incremental online clustering algorithm is proposed so as to cluster a stream of twitter messages in real time. A threshhold value is required which is determined emperically. This algorithm will take the incoming messages one by one, as they arrive in time, and assign a suitable cluster based on the similarity.

3.2 Term Frequency-Inverse Gravity Moment Based Metric

An improvement over tf-idf was proposed by Chen K. et al. (2016) paper.TF-IGM(term frequency and inverse gravity moment). assigns weight to a term based

Algorithm 1 Tf-idf based clustering algorithm

Input : Incoming Tweet M_i, set of existing cluster C = c_1, c_2, c_3,....,c_k, threshhold τ
T = cleanTweet(M_i)
maxSim = 0.0
for each cluster c_j in C **do**
  V1 = TF-IDF vector of T
  V2 = TF-IDF vector of c_j
  Sim = cosineSimilarity(V1, V2)
  if Sim >maxSim **then**
    maxSim = Sim
    c_m = c_j
  end if
end for
if maxSim >τ **then**
  add Tweet T to cluster c_m
else
  create new cluster c_{k+1}
  add Tweet T to cluster c_{k+1}
end if

on its class (maybe a document, in case of query searching, or a cluster like in our case.) distinguishing power. IGM is defined as:

$$igm(t_k) = \frac{f[k1]}{\sum_{r=1}^{m} f[kr] * r} \tag{3.5}$$

Here, igm(t[k]) denotes the inverse gravity moment of the term t_k, m specifies number of classes and f[kr] for (r = 1, 2, ..., m) are the frequencies of term t_k in descending order, and r is the rank.

Two weighting factors - local and global have been defined. The global weighing factor i.e. igm specifies the importance of a word in the document and its contribution to text classification. Thus, a new global weighing factor is defined as:

$$wg(t_k) = 1 + \lambda * igm(t_k) \tag{3.6}$$

where wg(t_k) is the global weighing factor of term t_k , is adjustable coefficient. The tf-based local weighing factor, wl(t_k,d), can be given by:

$$wl(t_k, d) = 1 + \log(tf[t_k]) \tag{3.7}$$

where tf[kd] is the number of term t_k's occurrences in a document d.The TF-IGM weight of term t_k in a document d is thus, the product of the tf-based local weighing factor and the IGM-based global weighing factor. Thus, we can write :

$$w(t_k, d) = wl(t_k, d) * wg(t_k) \qquad tf[kd] > 0 \tag{3.8}$$

and w(t_k,d)=0 if we have, tf[kd]=0.

The incremental online clustering algorithm as described in previous section is used with tf-igm weight vectors to give a cluster assignment based on the similarity between message and the existing clusters.

Algorithm 2 Tf-igm based clustering algorithm

Input : Incoming Tweet M_i, set of existing cluster C = c_1, c_2, c_3,....,c_k, threshhold τ
T = cleanTweet(M_i)
maxSim = 0.0
for each cluster c_j in C **do**
  V1 = TF-IGM vector of T
  V2 = TF-IGM vector of c_j
  Sim = cosineSimilarity(V1, V2)
  if Sim >maxSim **then**
    maxSim = Sim
    $c_m = c_j$
  end if
end for
if maxSim >τ **then**
  add Tweet T to cluster c_m
else
  create new cluster c_{k+1}
  add Tweet T to cluster c_{k+1}
end if

3.3 Limitations: Tf-Idf and Tf-Igm Based Approach

- Also, microblog posts, are very short and have large number of grammatical errors and informal language, which can result in statistical inaccuracies, and cannot capture synonyms and semantics.
- Different authors have different writing styles which results in noisy estimation of term frequency metric.
- These two approaches compute similarity between the documents, directly in the word count space, thus it becomes very slow as vocabularies increase in size.
- It also assumes that each word contribute independently to similarity, and does not take into any associativity between co-occurring words.
- These approaches does not consider any semantic similarities between words.

Chapter 4

Relational Term Weighing Metric

This chapter discusses the proposed word co-occurrence based weight metric for calculating similarity for event detection, and compares it with already existing weight metrics.

4.1 Overview

Microblogs now-a-days have become a widely used source of up-to date information for latest topics, trends or events. There are a lot of existing techniques that tries to provide useful information from this data, but an analysis based on co-occurrence of words is not exploited.

Our approach for the weight metric consists of using word co-occurrences and maintain corresponding data structure Bloom filter for each to calculate term weights for the posts. A number of limitations with existing approaches has been discussed in the previous chapter. The proposed approached solves some of these problems by considering the repetition in co-occurrence of the terms, rather than taking each term individually.

4.2 Bloom Filter

A Bloom filter is a probabilistic data structure that aims at determining whether an element is probabilistically present in the set or definitely not. It is very efficient in terms of memory space and the computation time. But because of probabilistic nature of membership testing, false positives are possible. This data structure does not store the actual elements, so it is not possible to get them. It is a m bit array, with each bit, initially set to 0. A k no. of hash functions are required to insert and query the elements in a bloomfilter.

For searching an element, as soon as we find a 0 in any of the k hash indices, we return a definite absence of the element, and if all the bit positions are set to 1, we can say that element may be possible in the set.

A formula for calculating number of items in a Bloom filter, was given by Swamidass and Baldi (2007):

$$n^* = \frac{-m}{k}(\ln(1 - \frac{X}{m})) \tag{4.1}$$

Here, n* = the estimate of number of items in the filter
m = the length of filter
k = number of hash functions, and
X = number of bits set to one.

4.3 Proposed Weight Metric

The concept behind our proposed metric is that if a word is co-occurring with same set of words again and again, then we can say that it is associated with a few distinct words, thus in a sense, it is rare, and is more informative for calculating similarity.

We define a weighting scheme, that assigns each term in a document, a weight based on its average co-occurrence repetition per word defined as ACR, such that a term appearing with a large distinct set of other words, acquires a low value, and the term appearing with same set of words only, is assigned a higher value.
For each term,

$$ACR(t) = \frac{Total\ no.\ of\ co\text{-}occurrence\ repetition\ of\ t}{Distinct\ words\ with\ which\ t\ co\text{-}occurred\ at\ least\ twice} \tag{4.2}$$

ACR is calculated by making use of Bloom filters, where insertions are done in bloom filters associated with co-occurring terms. Consider a twitter message T_1 = w_1,w_2,w_3, which is added to say, cluster C_1, then a bloom filter is created for each of these words in T1 if it does not exist already, however no insertions are done in the cluster since this is the first instance of w1,w2 co-occurring. Now at a later time, a tweet T_k = w_4, w_1, w_5,w_2 is added to the cluster C_1, then a bloom filter already exists for w_1 and w_2 in that cluster, this means that they are co-occurring again, and therefore we insert w_1 in bloom filter[w_2] and w_2 in bloom filter[w_1].
So, we can re-write ACR as:

$$ACR(t) = \frac{No.\ of\ insertions\ in\ bloom\ filter\ of\ t}{No.\ of\ items\ in\ bloom\ filter\ of\ t} \tag{4.3}$$

Normalized values of ACR is used for our purpose by taking logarithm,

$$Normalized\ ACR(t) = 1 + \log(ACR(t)) \tag{4.4}$$

The normalization of ACR values is necessary because the raw values are not proportional to the importance of term in the document(or, cluster). Suppose, ACR(t_1) = 2 such that there are 3 co-occurrences with a single term, and ACR(t_2) = 1 such

that there are 2 co-occurrences with a term, it does not mean that first term is twice as important as the second one.Thus, normalization is performed.

Term frequency (tf) is used to measure how frequently a term appears in a document (here, a cluster or a tweet). TF component, that determines the frequency of a word in the document is combined with the ACR measure that reflects importance of each term, to give combined final metric (eq. 4.4) Taking normalized values of tf and the above calculated normalized ACR , we give our final term-weight metric for a term in cluster c as:

$$\textit{Tf-ACR(t)} = 1 + log(tf(t)) * 1 + \log(ACR(t)) \tag{4.5}$$

where tf gives the raw term frequency of the term t and ACR is the average co-occurrence repetition per word of term t.

4.4 Algorithm

The following algorithm is proposed, for using the above described metric in an incremental clustering framework[4] as discussed in section 3.1 to cluster each incoming tweet to the cluster with highest similarity. We use cosine similarity metric as similarity function. Threshold value is determined empirically. Following is a description of some of the functions that are used in the algorithm :

- cleanTweet() function performs the necessary data cleaning steps including tokenization, stop-word removal, lemmatization, URL removal and returns a list of terms after the pre-processing.
- createBloomfilter(c,w) creates a bloomfilter for word w in cluster c
- existsBloomfilter(c,w) returns 1 if bloomfilter for word w exists for cluster c, else return 0
- insertBloomfilter(c,w1,w2) inserts w2 in bloomfilter for word w1 for cluster c

Algorithm 3 Co-occurrence based clustering algorithm

Input : Incoming Tweet M_i, set of existing cluster C = c_1, c_2, c_3,....,c_k, threshhold τ
T = cleanTweet(M_i)
maxSim = 0.0
for each cluster c_j in C **do**
  V1 = TF-ACR vector of T
  V2 = TF-ACR vector of c_j
  Sim = cosineSimilarity(V1, V2)
  if Sim >maxSim **then**
    maxSim = Sim
    cluster = c_j
  end if
end for
if maxSim >τ **then**
  add Tweet T to cluster c_j
else
  create new cluster c_{k+1}
  add Tweet T to cluster c_{k+1}
  cluster = c_{k+1}
end if
for every pair of words (w_1,w_2) in T **do**
  if existsBloomfilter(cluster,w1) and existsBloomfilter(cluster,w2) **then**
    insertBloomfilter(cluster,w1,w2)
    insertBloomfilter(cluster,w2,w1)
  end if
end for
for every term in T **do**
  if not existsBloomfilter(cluster,term) **then**
    createBloomfilter(cluster,term)
  end if
end for

Chapter 5

Results and Discussions

This chapter discusses the experimental setting and evaluation results of our implemented approach and presents comparison with the other approaches.

5.1 Experimental Set-up

The proposed metric was implemented with the help of bloomfilter as discussed in the algorithm 4.3. The size of the bloomfilter used is 64 bit with 2 hash functions. xxhash function is used to calculate hash values since it is a non-cryptographic hash algorithm aimed at speed and quality.

5.1.1 Dataset and Ground Truth Creation

We took the open general event dataset from [18] GWU Libraries Dataverse (George Washington University) - Harvard Dataverse. This dataset was collected over a timeperiod of January 27, 2017 to January 2, 2019 from the Twitter API using Social Feed Manager. It maybe possible that some tweets are from before this time period. This dataset was manually annotated for creation of ground truth to evaluate clustering algorithms. The dataset was divided into three sets, and the results were taken individually for each of the dataset.

5.1.2 Evaluation Metrics

The evaluation criteria of precision, recall and F-Measure was used to evaluate the efficiency of the proposed algorithm and then to compare it to other methods. Hatzivassiloglou and McKeown (1993) present an evaluation method for clustering. The evaluation is done by considering common cluster membership of object pairs in clustering. This common cluster membership is used to calculate recall and precision. Recall and precision is calculated as give in equations (5.1) and (5.2).
True positives tp, are the two points that are present in same cluster in both the predicted and the ground truth cluster set. False positives, fp are the two points that are present in the same cluster in the predicted cluster set but not in the ground truth cluster set. False negatives, fn are the two points that are in same cluster in the ground truth cluster set but not in the predicted cluster set True negatives, tn are the two points that are not there in same cluster in the ground truth cluster set or the predicted cluster set.

$$Precision = \frac{tp}{tp + fp} \tag{5.1}$$

$$Recall = \frac{tp}{tp + fn} \tag{5.2}$$

The formula for the traditional f-measure (F1 score) is the harmonic mean of these two values.

$$F\text{-}Measure = (1 + \beta^2)\frac{precision * recall}{\beta^2 * precision + recall} \tag{5.3}$$

For our purposes we have taken $\beta = 1$.

5.2 Results

We have examined the performance of tf-acr based clustering with tf-idf and tf-igm weight based clustering. The performance on the three datasets tries to reflect the

accuracy of each clustering method. The first set of results, Fig 5.1, 5.2, 5.3 compares precision, recall, f-measure for each of the three algorithms on three datasets individually.

Not surprisingly, tf-acr clustering performed well across all three datasets, better than both tf-idf and tf-igm clustering in terms of f-measure. An improvement of as high as 7% in fmeasure is obtained with respect to tf-idf, and as high as 3% with respect to tf-igm. Precision and recall also shows improvements with respect to all three datsets.

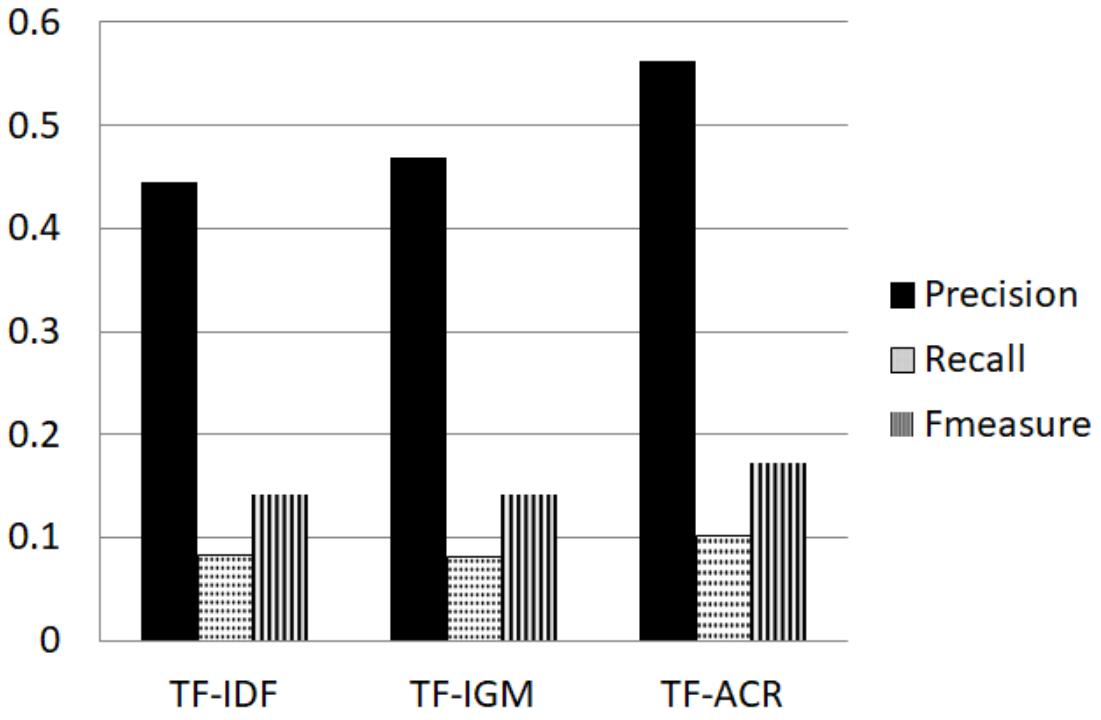

FIGURE 5.1: Comparison of three metrics on Dataset 1

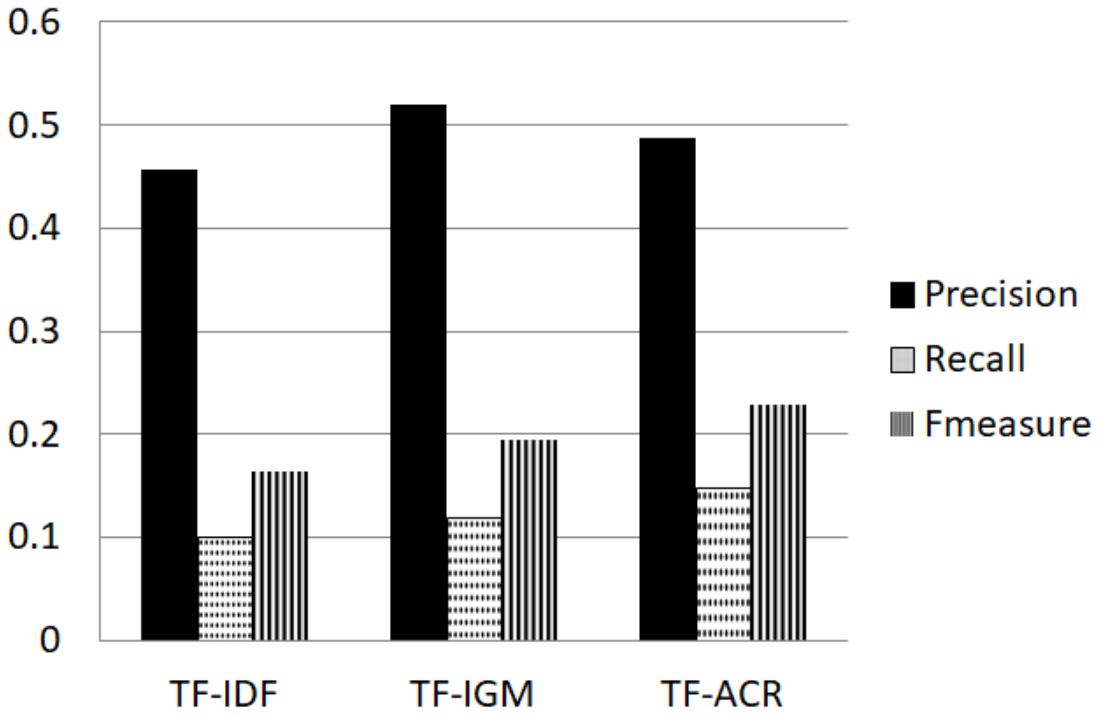

FIGURE 5.2: Comparison of three metrics on Dataset 2

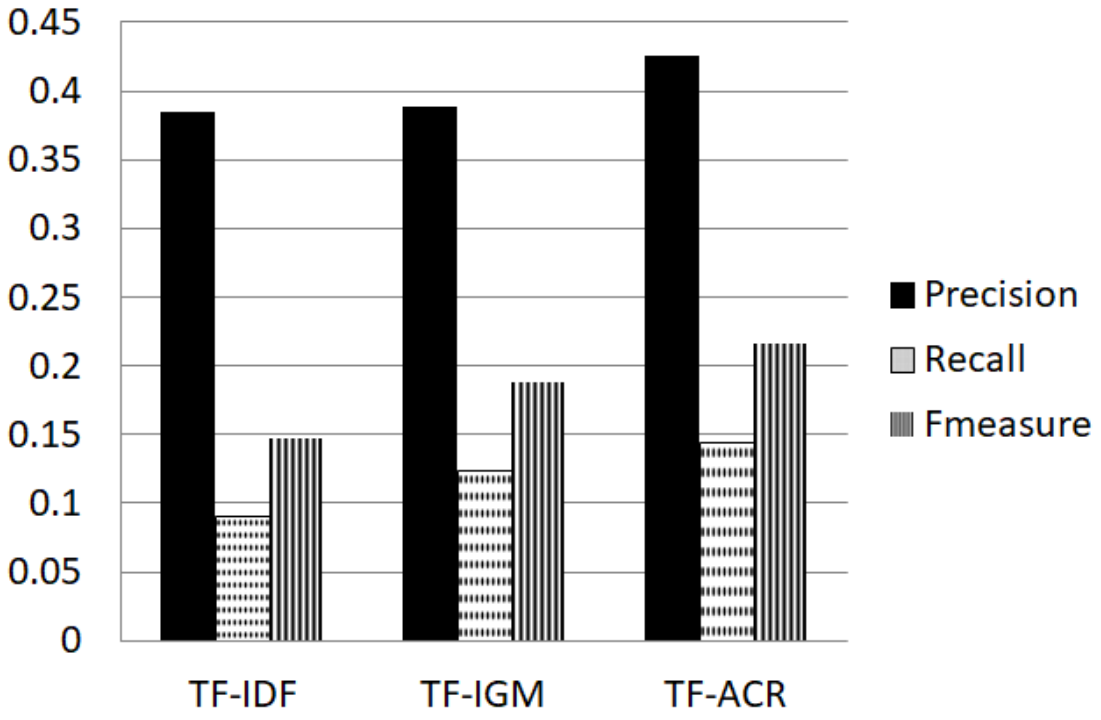

FIGURE 5.3: Comparison of three metrics on Dataset 3

Fig 5.4, 5.5, 5.6 compares our proposed metric against the two existing ones in terms of average precision, average recall, and average f-measure. As per the results shown in these figures, it can be observed that the proposed relational metric, tf-acr gives better performance against the two baselines. The average f-measure over three datasets is improved by 5% as compared to tf-idf, and 3% as compared to tf-igm. Average precision is increased by 7% and 4% respectively for tf-idf and tf-igm, and average recall has increased by 4% and 3% against the two metrics respectively.

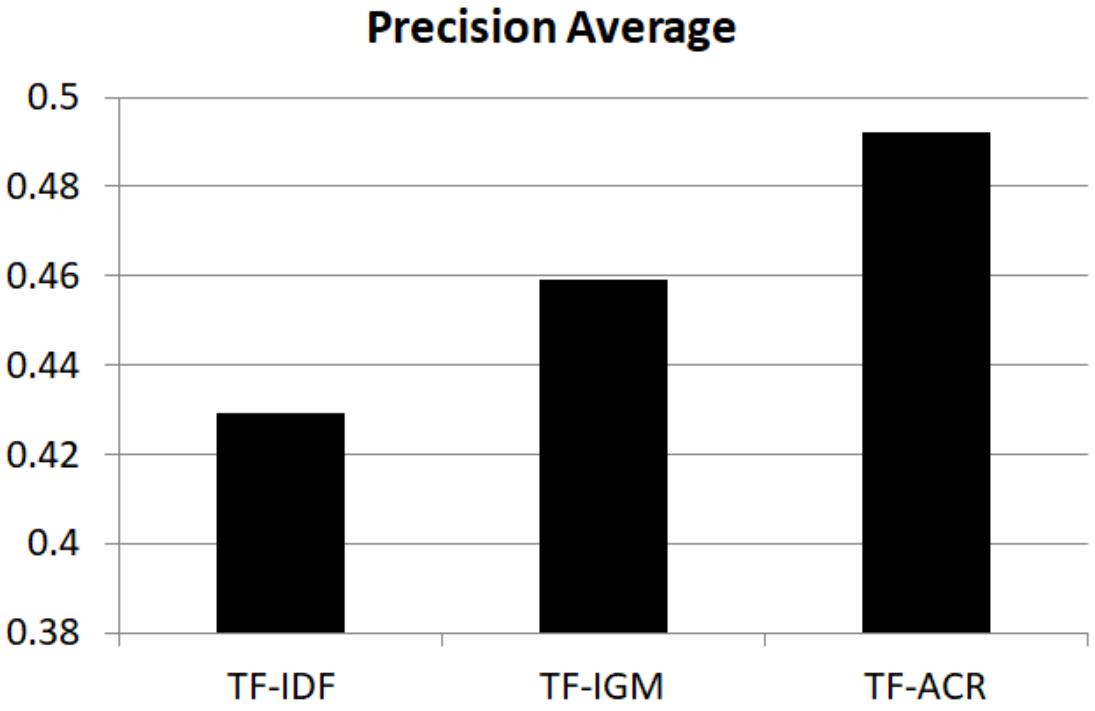

FIGURE 5.4: Comparison of average precision for three metrics

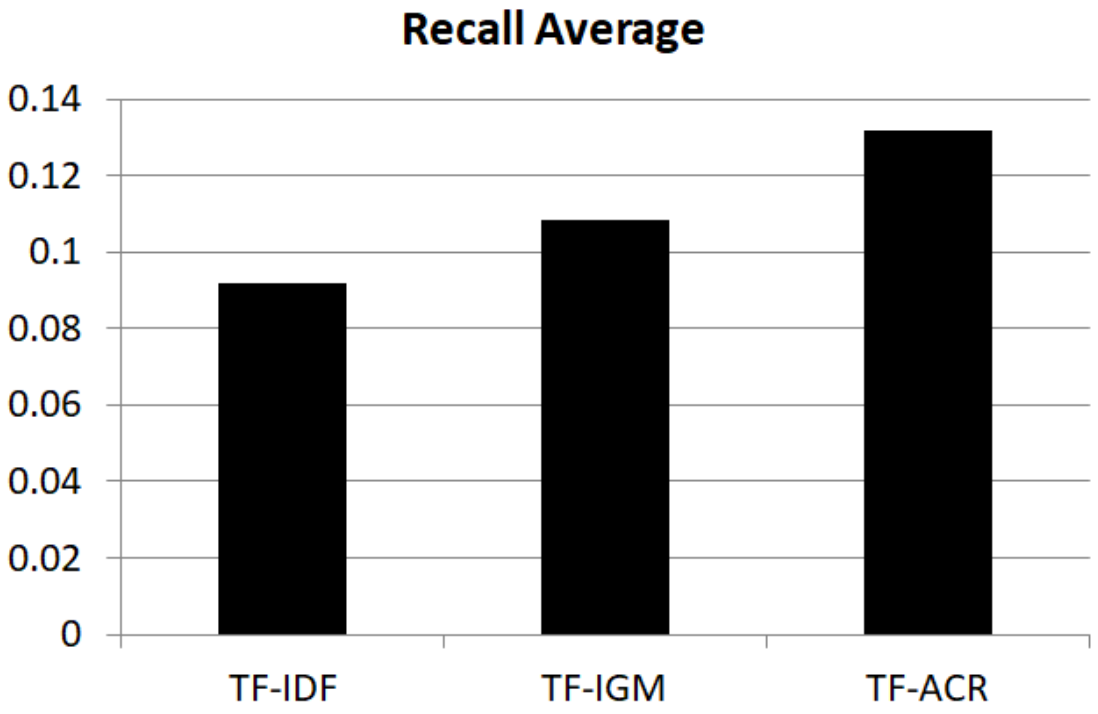

FIGURE 5.5: Comparison of average recall for three metrics

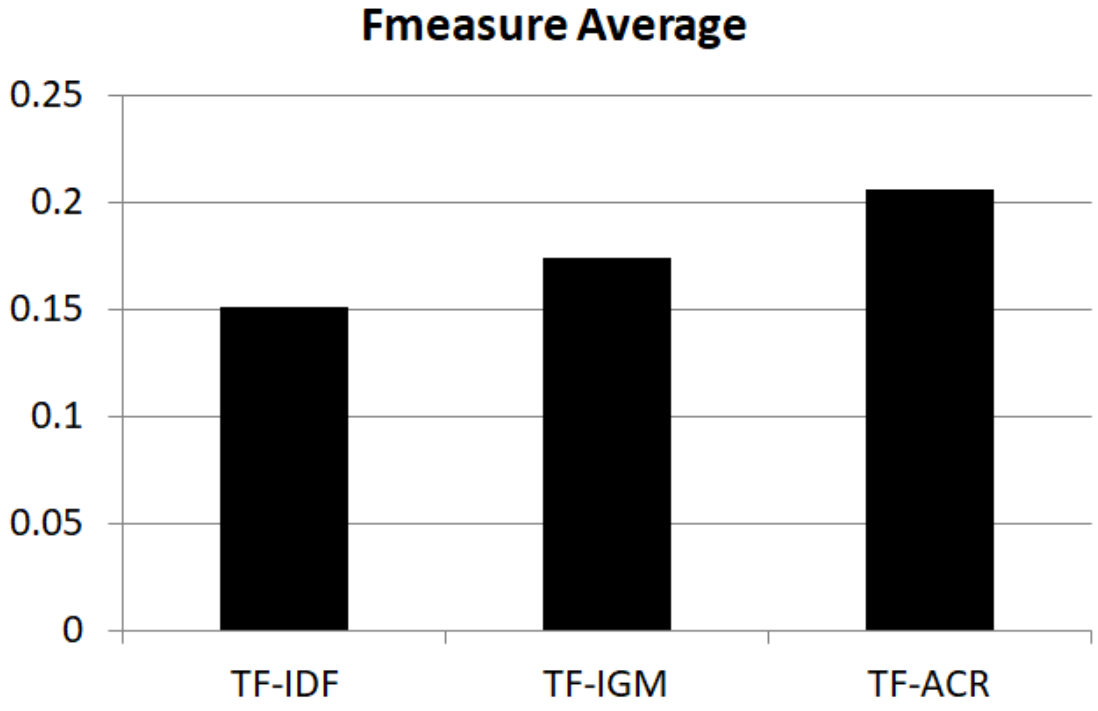

FIGURE 5.6: Comparison of average f-measure for three metrics

The second set of results Table 5.1 show the most significant words i.e. the words assigned highest weightage according to the underlying weighing scheme from two randomly selected clusters describing same event respectively from each of the three clustering implementations.

Weight Metric	**Most significant words**
TF-IDF	organisation, voluntary, connect, register, evacuation, apply, harvey, location, marked
TF-IGM	transport, conclude, assist, still, harvey, reminder, evacuation, run, well
TF-ACR	harvey, info, reservoir, affected, evacuation, shelter, flood, water, news

TABLE 5.1: Most significant words describing event related to hurricane harvey floods for each of the three metric

Weight Metric	**Most significant words**
TF-IDF	classmate, nursing, imminent, heretostay, title, dreamer, reduce, military, defenddaca
TF-IGM	renewal, apply, requirement, expire, meet, request, daca, dreamer, compassion
TF-ACR	dreamer, defenddaca, stand, economy, child, community, contribute, daca, leader, member

TABLE 5.2: Most significant words describing event related to daca for each of the three metric

In the table 5.1, we can observe that the main keyword harvey appears at seventh position in tf-idf, fifth position in tf-igm, however, it appears as the most significant word in case of tf-acr metric. Also, the set of top significant words in tf-acr are better

at describing the hurricane harvey floods related event. This is mainly because tf-igm is able to mitigate effect of noisy words, and exploits co-occurrence in words to give higher weightage to such terms.

In the table 5.2, we can observe that the main keywords dreamer, daca, defenddaca occupy much higher positions in tf-acr as compared to tf-idf and tf-igm owing to the above described reasoning.

5.3 Complexity Analysis

If m is number of existing clusters, and n is no. of terms in a tweet, then for clustering of each tweet, and for every term t in the tweet, In tf-idf and tf-igm based approaches, we need to search the term t in a cluster and obtain its frequency, with which current similarity is being calculated, as well as we need to search every other cluster if it contains the term t, resulting in time complexity of the order O(m*m*n). ACR based approach avoids cross checking with other clusters, rather we need to only obtain the co-occurrence statistics of that term in only the cluster, with which current similarity is being calculated, thus reducing the time complexity to order O(m*n).

5.4 Discussions

It can be observed that our proposed co-occurrence metric performed significantly better than the two other approaches. Following are the salient features of our proposed metric which attributed to these results:

1. Idf (Inverse document frequency) or Igm (Inverse Gravity Moment)of a term gives the rarity of the term in set of documents. A rare term is more informative, thus its term weight is high. ACR(Average Co-occurrence Repitition per word) can be used to replicate this functionality of idf because if a word is co-occurring with a different set of words(low ACR), that means it is less rare whereas if ACR is high,

then it implies that a word is co-occurring with same set of words again and again, thus are more informative, and in the sense of co-occurrence are more rare.

2. Twitter messages, contain little textual information, and exhibit low quality (e.g., with typos and ungrammatical sentences). Both tf-idf based and tf-igm based approaches are greatly impacted by such noisy words, since they are treated same as rest of the words. In co-occurrence based approach, a term's weight depends on its probability of co-occurrence with similar words. Statistically, it is less probable for noisy words(spelling errors, slangs, etc.) to occur with same set of words only, thus decreasing their weight. Hence, improving the clustering performance.

3. As discussed in previous section, complexity of our proposed approach has reduced, making our algorithm faster and efficient.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

Users of microblogs tend to write about what they are experiencing. Especially, when a real-world event occur in their surrounding environment, they tend to write about it to inform their network of users. Information about occurring events are buried within a huge amount of unnecessarily textual data. There is an urgent need to detect events for decision makers to take actions early.

In this research, we have successfully designed a co-occurence based term-weighing metric which is used to computer similarity between microblog posts, and cluster these posts based on computed similarity. Our approach makes use of only the contextual information and does not consider any structural information like user-names, geolocations, etc. Experimental results show that our approach performs significantly better than other similar existing ones. This approach mitigates the inefficiencies because of noisy words in these posts, and by eliminating the need to cross-reference every other cluster during decision making of a tweet, reduces the time complexity of clustering algorithm.

6.2 Future Scope

After understanding the results of experiments and limitations that we faced in our thesis, this work can be improved as follows:

- Use an efficient and adaptive threshold function, rather than computing it emperically.
- This approach is currently efficient for clustering small independent posts, but can be extended for use with retrospective documents.
- We can add a cluster summarization component to the existing methodology so that the approach can be more useful to end users

Bibliography

[1] S. Phuvipadawat and T. Murata, "Breaking news detection and tracking in twitter," in *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, vol. 3, pp. 120–123, Aug 2010.

[2] H. Becker, M. Naaman, and L. Gravano, "Beyond trending topics: Real-world event identification on twitter," in *Fifth international AAAI conference on weblogs and social media*, 2011.

[3] S. Petrović, M. Osborne, and V. Lavrenko, "Streaming first story detection with application to twitter," in *Human language technologies: The 2010 annual conference of the north american chapter of the association for computational linguistics*, pp. 181–189, Association for Computational Linguistics, 2010.

[4] J. Weng and B.-S. Lee, "Event detection in twitter," in *Fifth international AAAI conference on weblogs and social media*, 2011.

[5] M. Cordeiro, "Twitter event detection: combining wavelet analysis and topic inference summarization," in *Doctoral symposium on informatics engineering*, pp. 11–16.

[6] R. Li, K. H. Lei, R. Khadiwala, and K. C.-C. Chang, "Tedas: A twitter-based event detection and analysis system," in *2012 IEEE 28th International Conference on Data Engineering*, pp. 1273–1276, IEEE, 2012.

[7] C. Li, A. Sun, and A. Datta, "Twevent: segment-based event detection from tweets," in *Proceedings of the 21st ACM international conference on Information and knowledge management*, pp. 155–164, ACM, 2012.

[8] A. Guille and C. Favre, "Mention-anomaly-based event detection and tracking in twitter," in *2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014)*, pp. 375–382, IEEE, 2014.

[9] X. Dong, D. Mavroeidis, F. Calabrese, and P. Frossard, "Multiscale event detection in social media," *CoRR*, vol. abs/1404.7048, 2014.

[10] N. D. Doulamis, A. D. Doulamis, P. Kokkinos, and E. M. Varvarigos, "Event detection in twitter microblogging," *IEEE Transactions on Cybernetics*, vol. 46, pp. 2810–2824, Dec 2016.

[11] Y. Yılmaz and A. O. Hero, "Multimodal event detection in twitter hashtag networks," *Journal of Signal Processing Systems*, vol. 90, pp. 185–200, Feb 2018.

[12] M. Hasan, M. A. Orgun, and R. Schwitter, "Real-time event detection from the twitter data stream using the twitternews+ framework," *Information Processing Management*, vol. 56, no. 3, pp. 1146 – 1165, 2019.

[13] D. T. Nguyen and J. E. Jung, "Real-time event detection for online behavioral analysis of big social data," *Future Generation Computer Systems*, vol. 66, pp. 137 – 145, 2017.

[14] S. Gaglio, G. L. Re, and M. Morana, "Real-time detection of twitter social events from the user's perspective," in *2015 IEEE International Conference on Communications (ICC)*, pp. 1207–1212, IEEE, 2015.

[15] N. Alkhamees and M. Fasli, "Event detection from social network streams using frequent pattern mining with dynamic support values," in *2016 IEEE International Conference on Big Data (Big Data)*, pp. 1670–1679, Dec 2016.

[16] S. Katragadda, R. G. Benton, and V. V. Raghavan, "Framework for real-time event detection using multiple social media sources," in *HICSS*, 2017.

[17] K. Chen, Z. Zhang, J. Long, and H. Zhang, "Turning from tf-idf to tf-igm for term weighting in text classification," *Expert Systems with Applications*, vol. 66, pp. 245 – 260, 2016.

[18] [online] GWU Libraries Dataverse (George Washington University) - Harvard Dataverse. Available at: `https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/UIVHQR`.

[19] J. Allan, J. G. Carbonell, G. Doddington, J. Yamron, and Y. Yang, "Topic Detection and Tracking Pilot Study Final Report," 5 2003.

[20] B. Batrinca and P. C. Treleaven, "Social media analytics: a survey of techniques, tools and platforms," *AI & SOCIETY*, vol. 30, pp. 89–116, Feb 2015.

[21] J. Hurlock and M. L. Wilson, "Searching twitter: Separating the tweet from the chaff," in *ICWSM*, 2011.

[22] G. Petkos, S. Papadopoulos, L. Aiello, R. Skraba, and Y. Kompatsiaris, "A soft frequent pattern mining approach for textual topic detection," in *Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS14)*, p. 25, ACM, 2014.

[23] [online] Data types - Scipy.org . Available at: `https://docs.scipy.org/doc/numpy-1.13.0/user/basics.types.html`.

[24] [online] stackoverflow.com . Available at: `https://stackoverflow.com/questions/942543/operation-on-every-pair-of-element-in-a-list/37907649`.

[25] [online] Bloom Filters - Wisc.edu . Available at: `http://pages.cs.wisc.edu/~cao/papers/summary-cache/node8.html`.

[26] [online] Text Preprocessing for Machine Learning & NLP - Kavita Ganesan. Available at: `http://kavita-ganesan.com/text-preprocessing-tutorial/`.

[27] [online] Text Preprocessing in Python - Medium.com . Available at: `https://link.medium.com/umWZbCfA1U`.

[28] [online] Bloom filter - Wikipedia. Available at: https://en.wikipedia.org/wiki/Bloom_filter#Approximating_the_number_of_items_in_a_Bloom_filter.

[29] [online] GeeksforGeeks.com . Available at: https://www.geeksforgeeks.org/count-set-bits-in-an-integer/.

[30] [online] Itertools in Python 3. Available at: https://realpython.com/python-itertools/.

[31] [online] TF-IDF - Wikipedia. Available at: https://en.wikipedia.org/wiki/Tf%E2%80%93idf.

[32] [online] TF-IDF Model for Page Ranking - GeeksforGeeks. Available at: https://www.geeksforgeeks.org/tf-idf-model-for-page-ranking/.

[33] [online] Clustering Algorithms and Evaluations. Available at: https://www.ims.uni-stuttgart.de/institut/mitarbeiter/schulte/theses/phd/algorithm.pdf.

[34] [online] Calculation of clustering success. Available at: https://www.quora.com/How-do-I-calculate-clustering-success.

[35] [online] Bloom filter - Github. Available at: https://llimllib.github.io/bloomfilter-tutorial/.