CONTEXT BASED SEMENTIC RELATION IN TWEETS

[1]Prof. Sumeet Pate, [2]Swapnil Wankhede, [3]Shoyeb Khan, [4]Sanjay Chauhan
[1]Asst. Professor, [2,3,4]BE Student, [1,2,3,4]Comp. Engg. Dept, SSJCET, Asangaon, India.
[1]sumeetpate09@gmail.com, [2]swapnilwankhede93@gmail.com, [3]shoyebkhan22@gmail.com, [4]jay.199354@gmail.com

Abstract: Twitter, a popular social networking platform, provides a medium for people to share information and opinions with their followers. In such a medium, a flash event finds an immediate response. However, one concept may be expressed in many different ways. Because of users different writing conventions, acronym usages, language differences, and spelling mistakes, there may be variations in the content of postings even if they are about the same event. Analysing semantic relationships and detecting these variations have several use cases, such as event detection, and making recommendations to users while they are posting tweets. In this work, it applies semantic relationship analysis methods based on term co-occurrences in tweets, and evaluate their effect on detection of daily events from Twitter. The results indicate higher accuracy in clustering, earlier event detection and more refined event clusters.

Keywords —C4.5, datasets, entropy, stringtokenizer, decision tree, MOOC

I. INTRODUCTION

Social networking platforms, especially micro-blogging sites such as Twitter, act as an important medium where people share their opinions with their followers. With its millions of users around the world and half a billion tweets per day, textual content in Twitter is an abundant and still growing mine of data for information retrieval researchers. This retrieved information is used for analysing public trends, detecting important events, tracking public opinion or making on-target recommendations and advertisements. However, there may be variations in the contents of such a free and unsupervised platform, even if people refer to the same topic or the same concept in their tweets. Users may express the same thing in their own personal ways, such as by using unusual acronyms or symbols. Geographic, cultural and language diversities cause variations in the content. Even the regional setting or character set used on the device for posting affects uniformity. Moreover, the character limitation of a tweet in Twitter forces people to write in a compact form, possibly with abbreviations or symbols For these reasons, in order to apply information retrieval algorithms more effectively in such an environment, it is necessary to be able to figure out the semantic relationships among the postings under the variations. In this, goal is to identify such cases, and understand what the user could have meant, so that it can enrich a given tweet with possible similar terms.

In this work, it devises methods to extract semantic relationships among terms in tweets and use them to enhance the event detection capability. For the extraction of semantic associations, it uses co-occurrence based statistical methods. Although the techniques it uses are independent of language.

In this, intuition is that implicit similarities among terms are time-dependent. In other words, consistently with the tweet graph, it observes that a new day mostly starts with new events and new terms in Twitter. Therefore it chooses to analyses term relations, i.e., co-occurrences, within the scope of per day. Using such a context based relation extraction and applying these relations for tweet expansion, it aim to obtain earlier event detection, with longer lifetime and accurately clustered tweets. Moreover it obtains more refined results so that the users can follow the daily reports more easily.

II. LITERATURE SURVEY

The literature related with context- based computing in mobile computing. They defined context-aware applications that have been the terms context and context awareness, listed the built, discussed approaches to sense and model the context, and looked into supporting infrastructures, security and privacy issues. The selection criteria were used to select and accept the context- based articles. If the papers did not meet the selection criteria, then they were excluded. The criteria are described as follows

2.1 PARTICLE FILTERS ALGORITHM

Step 1. **Initialization**: Calculate the weight distribution $Dw(x, y)$ from twitter users geographic distribution in Japan.

Step 2. **Generation**: Generate and weight a particle set, which Means *N* discrete hypothesis.
(1) Generate a particle set *S*0 = (*s*0,0, *s*0,1, *s*0,2, . . . , *s*0,*N*−1) and allocate them on the
map evenly: particle *s*0,*k* = (*x*0,*k*, *y*0,*k*, *weight*0,*k*), where *x* corresponds to the longitude and *y*
corresponds to the latitude.
(2) Weight them based on weight distribution *Dw*(*x*, *y*).
Step 3. **Re-sampling:**
(1) Re-sample *N* particles from a particle set *St* using weights of each particle and
allocate them on the map. (We allow to re-sample same particles more than one.)
(2) Generate a new particle set *St*+1 and weight them based on weight distribution *Dw*(*x*,*y*).
Step 4. **Prediction**:
Predict the next state of a particle set *St* from the Newton's motion equation.
(*xt*,*k*, *yt*,*k*) = (*xt*−1,*k*+ *vx*,*t*−1Δ*t* + *ax*,*t*−1 2 Δ*t*2, *yt*−1,*k* + *vy*,*t*−1Δ*t* +*ay*,*t*−12 Δ*t*2)
(*vx*,*t*, *vy*,*t*) = (*vx*,*t*−1 + *ax*,*t*−1, *vy*,*t*−1, *ay*,*t*−1)
ax,*t*= *N*(0; *σ*2), *ay*,*t*= *N*(0; *σ*2).
Step 5. **Weighing**: Re-calculate the weight of *St* by measurement *m*(*mx*,*my*) as follows.
Step 6. **Measurement**:
Calculate the current object location *o*(*xt*, *yt*) by the average of *s*(*xt*, *yt*) ∈*St*.
Step 7. **Iteration**:
Iterate Step 3, 4, 5 and 6 until convergence. Implementing a Bayes filter, and a member
of the family of sequential Monte Carlo methods. For location estimation, it maintains a
probability distribution for the location estimation at time *t*, designated as the belief *Bel*(*xt*) ={*xit*,
wit}, *i* = 1. . . *n*. Each *xit* is a discrete hypothesis about the location of the object. The *wit* are nonnegative
weights, called *importance factors*, which sum to one. The Sequential Importance
Sampling (SIS) algorithm is a Monte Carlo method that forms the basis for particle filters. The
SIS algorithm consists of recursive propagation of the weights and support points as each
measurement is received sequentially. In this it use a more advanced algorithm with re sampling.
It employ weight distribution *Dw*(*x*, *y*) which is obtained from twitter user distribution to take
into consideration the biases of user locations8 The algorithm is shown in Algo.

2.2 ALGORITHM FOR WORKFLOW OUTLIERMINING

FIOF(*ii*)=support(i) * ||FIS(P,min_support)||
Calculate X(arithmetic mean of FIOF(I))and S(standard deviation of FIOF(I))
//Calculate all Instances' FIOF average X and standard deviation S **let** lower_bound=X-S;
//According to the Empirical Rule, there is 68% data which lower bound is X-S **for each Input**：*P*, all Process' set, *P*= (*p1,p2,p3…pi…pn*) ; *I(pi)*,
each Process' Instance's set, *I(pi)*=(*i1,i2,i3…ii…in*) ;
S, all Instance's support's set, *S=(s1,s2,s3…si…sn) ;*
Output：Abnormal workflow set *AW*, for eachProcess *pi* do min_support
(*pi*)= Average number of Support;
FIS_number(*pi*)
//Find how many frequent instances are larger than min_support，||FIS(P,min_support)||。
{
Intnumber=0;
for(i=1; i<=n; i++)
if(si>= min_support) FIS_number=FIS_number+1;
else return FIS_number;
returnFIS_number;
//Search from the first instance to the last one. If the instance support is larger than min_support,
add the number.
}
AbnormalWMe(P)
//Mine less-occurring workflow in each Process, and its Process Activities sequence is the
workflow outlier
{ **for each** Instance *ii* **do** FIOF(*ii*)= si * FIS_number(*pi*)
//Calculate each instance's

FIOF(*ii*) **do selection_sorting()**;
//Use selection sorting to sort all Instances from small FIOF to large FIOF
for(*ii=1*; *ii*< n; *ii* ++)
if(FIOF(*ii*)<=lower_bound) return AW[*ii*] ;
elsereturn 0;
return0;
//Start from the smallest FIOF. If FIOF is lower than lower_bound,
return this Instance's Process Activities sequence. These left range outliers are Abnormal
Workflow.
}

III. ALGORITHMS

3.1 CLUSTERING ALGORITHMS

Clustering algorithms can be categorized based on their cluster model, as listed above. The following overview will only list the most prominent examples of clustering algorithms. Clustering algorithms can be categorized as

connectivity based clustering (hierarchical clustering), centroid-based clustering, distribution-based clustering, and density-based clustering.

3.2 CONNECTIVITY BASED CLUSTERING (HIERARCHICALCLUSTERING)

Connectivity based clustering, also known as hierarchical clustering, is based on the core idea of objects being more related to nearby objects than to objects farther away. As such, these algorithms connect "objects" to form "clusters" based on their distance. Single-linkage clustering and complete linkage clustering are the some example of this method.

3.3 CENTROID-BASED CLUSTERING

In centroid-based clustering, clusters are represented by a central vector, which may not necessarily be a member of the data set. Here it finds the k cluster centers and assign the objects to the nearest cluster center, such that the squared distances from the cluster are minimized. Lloyd's algorithm, often actually referred to as "k-means algorithm" is the example of this method.

3.4 DISTRIBUTION-BASED CLUSTERING

The clustering model most closely related to statistics is based on distribution models. Clusters can then easily be defined as objects belonging most likely to the same distribution. A nice property of this approach is that this closely resembles the way artificial data sets are generated by sampling the random objects from the given distribution. Expectation-maximization algorithm is the example of this method.

3.5 K-MEANS CLUSTERING

K-Means is an algorithm that will find k clusters for a given dataset. The number of clusters k is user defined. Each cluster is described by a single point known as the centroid. Centroid means it's at the center of all the points in the cluster. After this step, the centroids are all updated by taking the mean value of all the points in that cluster.

MSE = largenumber;
Select initial cluster centroids {*mj*}*j K* = 1;
Do
OldMSE= *MSE*;
*MSE*1 = 0;
For *j* = 1 to *k*
mj= 0; *nj*= 0;

endfor
For i=1 to n
For j=1 to k
Compute squared Euclidian disstanced2(xi,mj);
end for
find the closest centroid mj to xi ;
mj = *mj* + *xi*; *nj*= *nj*+1;
*MSE*1=*MSE*1+ *d*2(*xi*, *mj*);
Endfor
For *j* = 1 to *k*
nj = max(*nj*, 1); *mj* = *mj*/*nj*;
endfor
MSE=*MSE*1;
while (*MSE*<*OldMSE*)

IV. PROPOSED SYSTEM

FIELDS	DISTANCE BASED CLUSTERING	DENSITY BASED CLUSTERING	HIERARCHICAL CLUSTERING	K-MEANS CLUSTERING
Size of Dataset	Huge dataset	Huge dataset	Huge dataset & Small dataset	Huge dataset & Small dataset
Number of clusters	Large & Small	Does not require	Large & Small	Large & Small
Type of Dataset	Non-convex	factoextra package	Ideal & Random	Ideal & Random
Type of Software	reeview	R Packages	LNKnet & Treeview	LNKnet &Treeview
Method	It uses mean and medoid for representing cluster.	Distance between Nearest elements.	It forms a tree like Structure	partition the points into *k* groups
Working Process	It works by interating through each object that has to find in cluster.	Clusters are dense of each object that is being Separated by low density density region.	Use Divisive and Agglomerative method	Determining the number of categories that exist
Advantages	Robust, easy to understand and it does not require domain knowledge.	Random shaped cluster are formed.	We do not need to Know how many clusters are required In initial phase. No input parameters are Necessary.	Computationally faster than hierarchical clustering, Fast, robust and easier to understand.
Disadvantages	Preset number of cluster will make it difficult to Predict.	Cannot work efficiently with huge datasets.	Does not scale well	Unable to handle noisy Data and outliers.

Table 4.1 : Proposed System

V. MATHEMATICAL MODEL

5.1 FIRST ORDER RELATIONSHIPS

As explained before, in order to find the first order relationships, it uses the raw co-occurrence values. In previous work, after finding the number of times the term pairs co-occur, it identified the semantically related term pairs if they appear in more than 50 tweets together. Moreover, if two terms are found to be semantically related, it assigned a constant similarity score of 0.5. In this work, it developed a more generic solution and adopted the approach that it used for discovering hash tag similarities in. Instead of using a threshold for deciding the similarity of two terms and giving them a constant similarity score, it assign normalized similarity scores for each term pair by using their co-occurrence values. For example, the term pair with the maximum number of co-occurrence value, cmax, on a given day has the similarity score 1.0. For other term

pairs *ti* and *tj* with a co-occurrence count of *ci,j*, their similarity score is given by the ratio of *ci,j/cmax*.

5.2 SECOND ORDER RELATIONSHIPS WITH COSINE SIMILARITY

For the second order relationships, term co-occurrence vectors are generated. Let *ci,j* represent the number of co-occurrences of the terms *ti* and *tj*. Then, for each term*ti*, it count its co-occurrences with other terms *t*1, *t*2,…*t*|*w*| where *W* is the set of distinct terms collected on that day's tweets. After forming the term vectors as given in, **ti**= *(ci,*1*, ci,*2*, …, ci,i*−1*,* 0*, ci,i*+1*, …, ci,*|*W*|−1*, ci,*|*W*|*)* it compare their cosine distance by using the cosine distance equation in.

$$sim_{cosine}(t_i, t_j) = \frac{t_i \cdot t_j}{|t_i||t_j|} = \frac{\sum_{k=1}^{|W|} c_{i,k}c_{j,k}}{\sqrt{\sum_{k=1}^{|W|} c_{i,k}^2 \sum_{k=1}^{|W|} c_{j,k}^2}}$$

Again it do not use any threshold for the decision of similarity but rather use the cosine distance as the similarity score, which is already in the range [0,1].

5.3 SECOND ORDER RELATIONSHIPS WITH CITY-BLOCK DISTANCE

City-block distance is another simple vector comparison metric. After forming the co-occurrence vectors, while comparing two vectors, it finds the sum of absolute Differences for each dimension as given in.

Similar to the solution it applied for first order relations, it normalize the distances in [0, 1] and use these values as similarity scores.

$$sim_{city\text{-}block}(t_i, t_j) = \sum_{k=1}^{|W|} |c_{i,k} - c_{j,k}|$$

5.4 EVENT DETECTION AND SEMANTIC EXPANSION

In this work, it performs offline event detection on tweets. However the algorithms it implements can also are used online with further performance optimizations. The flow of the event detection process is depicted in Fig. Dashed arrows indicate the extension that we implemented on a traditional clustering algorithm. It first present the data collection, tweet vector generation, clustering and event detection steps.

Then it explains how it carries out lexico-semantic expansion and improves event detection quality. For tweet collection from the Twitter Streaming API, it use Twitter4J,2a Java library that facilitates the usage of Twitter API. It applies a location filter and gathers tweets posted by users in Turkey, with Turkish characters. Posts with other character sets such as Greek or Arabic letters are filtered out. The gathered tweets are immediately stemmed with a Turkish morphological analyser called TR Morph. After further pre-processing, including the removal of stop words and URLs, they are stored into the database. Using this process, it collects around 225K tweets per day. Further details regarding the tweet collection and pre-processing steps are found in this previous work.

VI. SYSTEM ARCHITECTURE

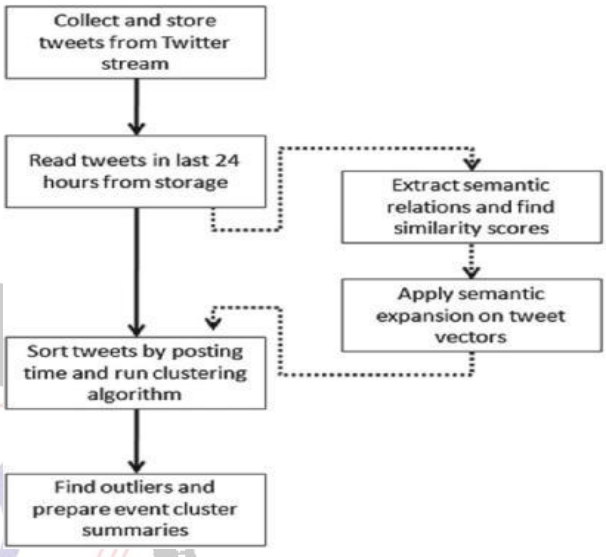

Fig 6.1 : **System Architecture**

VII. RESULTS AND OUTPUT

In the first three algorithms, the event detection times are almost the same. On the other hand, the duration of the first event cluster is longer for FO and COS algorithms. This is considered to be the result of successfully identified semantic associations.

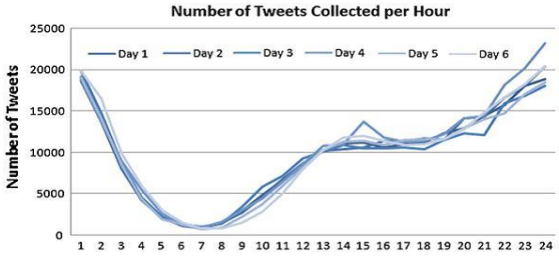

Fig 7.1 : Output

VIII. CONCLUSIONS

In this work, it aims to extract co-occurrences and use them in a semantic expansion process on tweets in order to detect events with higher accuracy, with larger time span, and in a user-friendly form. It improves its previous work by using similarity scores instead of thresholds and constant multipliers for semantic expansion. Moreover, it identifies context dependent associations by evaluating terms in specific associations among terms in Twitter by using their

time windows. Daily event clusters are determined by making an outlier analysis. Although in this, methods are applied on tweets in each day, they can be adapted to work in different time granularities or in an online system, which are planning to implement as a future work. Moreover, it would like to experiment periodically merging and/or dividing event clusters in the course of event detection in order to improve the resulting event clusters. In this, methods are tested on a set of around five million tweets collected in three weeks with Turkis content. It implemented three different semantic similarity metrics and evaluated them on this test data set. Results of these metrics are further analysed in the evaluations of our event detection methods. Improvements are observed in event detection in several aspects, especially when second order associations are used. As the methods it implement do not require a dictionary or thesaurus, they can be used for other languages as well.

REFERENCES

[1] James Cannady Jay Harrell ―A Comparative Analysis of Current Intrusion Detection Technologies.
[2] Mrudula Gudadhe, Prakash Prasad, Kapil Wankhade, Lecturer, ―a new data mining based network intrusion detection model‖iccct'10
[3] N.s.chandolikar & v.d.nandavadekar,International Journal of Computer Science and Engineering (IJCSE),Vol.1, Issue 1 Aug 2012 81-88 ―comparative analysis of two algorithms for intrusion attack classification using KDD cup dataset‖
[4] Dai Hong Li Haibo,2009 15th IEEE Pacific Rim International Symposium on Dependable Computing, A Lightweight Network Intrusion Detection Model Based on Feature Selection
[5] Bhavani Thuraisingham, Latifur Khan, Mohammad M. Masud, Kevin W.Hamlen The University of Texas at Dallas 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing,‖ Data Mining for Security Applications‖
[6] Radhika Goel, Anjali Sardana, and Ramesh C. Joshi ―Parallel Misuse andAnomaly Detection Model‖ International Journal of Network Security, Vol.14, No.4, PP.211-222, July 2012
[7] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. GhorbaniA Detailed Analysis of the KDD CUP 99 Data Set.
[8] Thales Sehn Korting,‖ C4.5 algorithm and Multivariate Decision Trees.
[9] P Amudha, H Abdul Rauf, ―Performance Analysis of Data Mining Approaches in Intrusion Detection‖
[10] Knowl Inf Syst (2008) 14:1–37 DOI 10.1007/s10115-007-0114-2 SURVEYPAPER, Top 10 algorithms in data mining XindongWu · Vipin Kumar · J.Ross Quinlan · Joydeep Ghosh · Qiang Yang ·Hiroshi Motoda · Geoffrey J. McLachlan · Angus Ng · Bing Liu · Philip S. Yu ·Zhi-Hua Zhou · Michael Steinbach · David J. Hand · Dan Steinberg
[11] Mrutyunjaya Panda1 and Manas Ranjan Patra2, IJCSNS International Journal of Computer Science and Network Security,VOL.7 No.12, December 2007‖ Network Intrusion Detection Using Naïve Bayes‖
[12] Shaik Akbar, Dr.K.Nageswara Rao, Dr.J.A.Chandulal, International Journal of Computer Applications (0975 –8887) Intrusion Detection System Methodologies Based on Data Analysis
[13] Nathan Einwechter, ―An Introduction To Distributed Intrusion Detection Systems‖
[14] Payam Emami Khoonsari and AhmadReza Motie, ―A Comparison of Efficiency and Robustness of ID3 and C4.5,Algorithms Using Dynamic Test and Training Data Sets‖, International Journal of Machine Learning and Computing, Vol. 2, No. 5, October 2012.

