Tweets Mining Using WIKIPEDIA and Impurity Cluster Measurement

Qing Chen, Timothy Shipper, Latifur Khan
Department of Computer Science
University of Texas at Dallas
Dallas, USA
[qingch, timship, lkhan@utdallas.edu]

Abstract— **Twitter is one of the fastest growing online social networking services. Tweets can be categorized into trends, and are related with tags and follower/following social relationships. The categorization is neither accurate nor effective due to the short length of tweet messages and noisy data corpus. In this paper, we attempt to overcome these challenges with an extended feature vector along with a semi-supervised clustering technique. In order to achieve this goal, the training set is expanded with Wikipedia topic search result, and the feature set is extended. When building the clustering model and doing the classification, impurity measurement is introduced into our classifier platform. Our experiment results show that the proposed techniques outperform other classifiers with reasonable precision and recall.**

Keywords-tweet mining; extended features; wikipedia

I. INTRODUCTION

Twitter is extensively used in various fields as an easy, fast and convenient information sharing tool. On twitter .com, the tweets are composed of the user ID and tweet contents. With our scrawling program that retrieves messages from the twitter website, http://search.twitter.com/trends/weekly.json, we can obtain the most weekly hot trend tweets categorized by hash tag, which has more features, such as tweets ID timestamps, language, and so on.

The most challenging issue relates to the multiple associated trends. As we have shown before, the hash tags are actually not topics, just "tag"/keywords. The problem is one message might have multiple tags/keywords, and thus it might be belonging to several related trends. In other words, several trends might talk about the same thing overlap. Since the hash tags are created by the user, there is no formulated naming convention for the tags - one meaningful tag might be displayed at different forms. Another challenge we face is that tweets are always very short and noisy which makes them very hard to be categorized. In addition, many tweets are not well formed in the English language. Many contain online chatting slangs, obstructed words, and many different ways of obfuscated words tend to appear here and there, like "hollyyyy". Thus, traditional information retrieval algorithms like, TFIDF[2], NN [1], etc, cannot successfully classify the tweets with reasonable precision and recall.

In this paper, we first expand the training set instances and the feature set using Wikipedia to overcome the short length of the tweet messages. New instances will be augmented along with expanded feature spaces in the training set. Second, we propose a revised K-NN classifier which considers the cluster impurity measurement. The basic idea is that instead of only measuring the euclidean distance among tweets in clustering, we try to minimize the cluster impurity. Also, during classification, the purer neighbor cluster will be selected, and this cluster label will be treated as the final classification decision for the tweet. When comparing other widely used classification techniques, our approach performs the best. The paper is organized the following ways. In section 2, we present baseline approach and then discuss our enhanced approaches. In section 4, we present our experiment results and analyses in detail.

II. OUR PROPOSED APPROACH

In this section, we will introduce our algorithm, and how we apply advance methods to improve the classifications.

A. NN clustering with Expanded features from Wikipedia (NNEW)

For classification, K-NN approach can be used. Due to the large and noisy training set, we need to do the clustering first, and then do the classification. By that means, the classification is more efficient and accurate. The training set normally has tens of thousands of records. After grouping into hundreds of clusters, the K-NN classifier can find the nearest neighbor cluster much faster. On the other hand, since the training set is noisy, the neighbor cluster can mitigate the noisy data impact. First, we apply a semi-supervised clustering algorithm in the training dataset. We partition n observations (tweets from training set) into k clusters in which each observation belongs to the cluster with the nearest mean. An efficient way to do the cluster training is Bisecting K-means Algorithm [1], which easily converges. The idea is: initially all training points are in a large cluster. In each iteration, using the Euclidean distance to pick the largest cluster to split using bisecting algorithm (will discuss later), until the desired number of clusters – number of k clusters is reached. Here is the objective function for K-NN clustering:

$O_{Kmeans} = (\sum_{x \in X_i} \| x - u_i \|^2) / \| X_i \|$, $\| X_i \|$ is the number vectors in a training cluster (1)

We notice that the topic number within the training set is unknown since one trend may contain multiple topics while several associated trends might overlap and share several topics as we have discussed before. If we assume one cluster contains one topic, and one topic might spread into several clusters then we cannot pre-determine or predict how many clusters there would be. Thus, instead of the clustering training iteration

stopping at k clusters reached; we stop at the measuring the range of the cluster. If the objective functions of all the training clusters are less than some threshold then we stop splitting the cluster.

One major drawback of the tweets training set is that the tweets are too short to address the full story of a topic. This impacts our topic grouping in two ways: the feature sets are limited and not accurate since the tweets are short, and the key words used in tweets are also limited, and what is worse, the tweets are not formal language expression. Secondly, the training set is not sufficient to grasp the topics because the tweets always give hints to the related topics.

To solve this problem, we introduce the wikipedia search expansion. For each tweet, we first do the stemming process to filter out stop words, slang words, unknown words etc, and collect the entities we believe might be related to the topic – now, it is the trend name. We do a wikipedia search based on trend name plus each entity, then the wikipedia will return a search result list containing hundreds of records. The records on the top of the list are most relevant to our research, we pull out a fixed number (could be 3,5 or 7) of record. For each wikipedia search return record, it has a topic and an around 30 words snippet, we add the short snippet to our training set. For a rough estimation, suppose we have 1000 training tweets. Each tweet has 5 entities on average, and we put 5 wikipedia search records into the training set. Finally the training dataset would have 1000*5*5+1000=26000 records. Also based on the wikipedia search records, we can extract the feature set based on the term frequency. Due to the large wikipedia search records and the formality of the wikipedia expressions we believe this new feature set is much more sophisticated. Now, our clustering algorithm actually has two phrases: first, applying wikipedia search to expand the training dataset and feature set, and second, using bisecting algorithm to cluster training data using augmented features.

B. Improvement NNEW – (pure-NNEW)

Based on Extended features from wikipedia of NN clustering (NNEW), we generate a set of clusters. Ideally, if every cluster contains only one class then we can clearly declare the cluster label. The decision of the K-NN classifier result made by majority cluster label voting will be more accurate. However, the tweets are very noisy, and wikipedia search results may introduce noisier vectors into the training set. Thus, the clusters are more likely in some training vectors with multiple labels, which is, the clusters are mostly impure. Next, we apply a penalty on the impurity in the objective function, and the purpose of minimizing the impurity within the cluster can be achieved. The objective function in equation (1) becomes:

$$O_{MCIKmeans} = (\sum_{x \in X_i} \| x - u_i \|^2 + \sum_{x \in X_i} \| x - u_i \|^2 \times \mathrm{Im}\, p_i) / \| X_i \|$$

$$= \sum_{x \in X_i} \| x - u_i \|^2 \ (1 + \mathrm{Im}\, p_i) / \| X_i \| \qquad (2)$$

There are two levels of concerns of the cluster impurity: firstly, in an impure cluster there might be a few classes taking large portion within the cluster. For example: three classes appear in one cluster and the percentage distribution is 40% of A, 30% of B, 30% of C; in this case, even thought class A is only slightly outnumbered class B and class C, the cluster is still assigned the label as "A". We declare that this cluster label is very "weak", and we should find a way to avoid it. We name this impurity issue as an "impurity problem". Secondly, the impure cluster might contain many classes with one major class in it. However, there are many other classes that coexist in the cluster even though they all take a very little portion in the cluster which makes the cluster very impure. We call this issue a "dispersion problem". Of course, the worst case is that one impure cluster might have both an impurity problem and a dispersion problem.

In order to minimize the impurity of a cluster, we need to suppress both problems. $\mathrm{Im}\, p_i$ should be a compound coefficient that penalizes both problems. To penalize the impurity problem is actually penalizing the uncertainty of the cluster label. A well-known measurement of a system's uncertainty is entropy. The uncertainty of a cluster can be defined as:

$Ent_i = \sum_{c=1}^{C} (-p_c^i \times \log(p_c^i))$, where p_c^i is the possibility of label c is the cluster i.

Definition: $| L_i |$ is the total number of vectors in cluster I, and $| L_i(c) |$ is the vector count of label c in cluster I, thus, $p_c^i = | L_i(c) | / | L_i |$.

To penalize the dispersion problem, we introduce the statistic dispersion coefficient, Gini coefficient.

$$Gini_i = \sum_{c=1}^{C} (p_c^i)(1 - p_c^i)$$

In all, the Minimization of Cluster Impurity is to minimize the inner cluster impurity while simultaneously minimizing the cluster dispersion just as the normal K-means clustering. So the objective function would be:

$O_{MCIKmeans} = \sum_{x \in X_i} \| x - u_i \|^2 \ (1 + \mathrm{Im}\, p_i) / \| X_i \|$, where

$\mathrm{Im}\, p_i = \alpha \times Gini_i \times Ent_i$, α is a constant.

C. Classification with impurity measure

When the new tweets arrive and for classification we apply Minimized Impure Cluster – K-NN (MCI-KNN) classification. Normally, K-NN finds k nearest neighbor clusters, the distance from the newly arrived tweets to the clusters is measured by the Euclidean Distance from the new vector to the clusters' centroid vectors.

$$M_{KNN} = \| x - u_i \|^2 \qquad (4)$$

MCI-KNN applies some penalty on impure neighbor clusters to the new vector, and by those means, we make a judgment on the classification of new tweets vector. Thus, we can find out purer neighbor cluster and can make clearer decision.

$$M_{MCIKNN} = \| x - u_i \|^2 \ (1 + \mathrm{Im}\, p_i) \qquad (5)$$

We observe that the impurity problem of a cluster is when the entropy of the cluster is large, and the label of the cluster is uncertain. This cluster's "vote" is also uncertain. Thus, we should avoid the cluster's vote. However, the dispersion problem of a cluster will not do significant harm to the cluster's vote. Hence, the measure function would be:

$$M_{MCIKNN} = \| x - u_i \|^2 \ (1 + \beta \times Ent_i) \qquad (6)$$

Table 1 Classification result comparison

	Naïve Bayes	Thematic Importance	Tag Classifier	NNEW Twitter	NNEW Wiki	NNEW Combine
TP	3378	3170	3915	3184	3432	3509
FP	11184	4542	3402	5329	1702	1570
TN	72816	79458	80598	1017	1854	82430
FN	822	1030	285	508	957	691
Precision	0.23	0.41	0.53	0.37	0.67	0.69
Recall	0.8	0.75	0.93	0.86	0.78	0.84
F-Measure	0.36	0.53	0.68	0.52	0.72	0.76

III. Results and Analysis

From http://search.twitter.com/trends/weekly.json, we retrieve weekly hot trends using a tweets crawling program written in Perl script. The raw data is in free text and a parser is implemented to retrieve the tweets attributes (tweets' from_id, to_id, timestamp, content, encoding type etc) and then store them into database. With the tweets DB, we applied naïve bayes classifier and tag classifier to build our research baselines (refer to 3.1).

A. Compare the baseline approaches and our approach

In table 1, we include the experiments result of our baseline approach and NNEW classifier. The experiment is setup for 21 hot trends; for each trend we collect 500 tweets for training and 200 tweets for testing.

In this comparison, we can see that the Naïve Bayes classifier does not work well for tweets classification, due to the short length of the messages and highly noisy and cross-related trends. Thematic importance Naïve Bayes classifier emphases the strongly associated keywords, and shows much better performance than regular Naïve Bayes classifier. The precision improves significantly, but is still quite low. Since the tweets are short and noisy, by just enhancing the associated keywords, naïve bayes classifier cannot overcome the problem.

Tag classifier works much better, especially in TP rate. However, FP rate is also high. This is due to exact word matching in Tag classifier. The tweet tagged as a particular trend name, normally contains its trend name in its message content, either hash tag or tag. If some trends of tweets are cross related or have class-subclass relationships then the FP rate will be high. In the last three columns of Table 1 we show the experiment results of NNEW for purer twitter dataset training, NNEW for wikipedia search results dataset training, and the twitter + wikipedia search results dataset training. It shows that if we apply NNEW on twitter training set only then the precision is quite low and F-measure is only 0.52. With wikipedia search training set the classification results improve to 0.72 in F-measure. If we combine the twitter messages and the wikipedia search results as a single training set, and then apply NNEW classifier, it gives the best results.

With the wikipedia search expanded features and expanded training set, the NNEW approach shows highest F-measure value. This means measuring both precision and recall, NNEW approach outperform the baseline approaches.

B. Figures and Tables

For NNEW, we expanded the training set and feature set for clustering with wikipedia search, the experiment result is shown in Figure 1, the y-axis refers to the F-measure value. We improved basic NNEW and applied the impurity minimization algorithm discussed in section 2.3. In NNEW-purity with purity measure we improve further by applying impurity measurement during classification. As a result, the cluster impurity minimization and classification impurity measurement improves the experiment results around 6%.

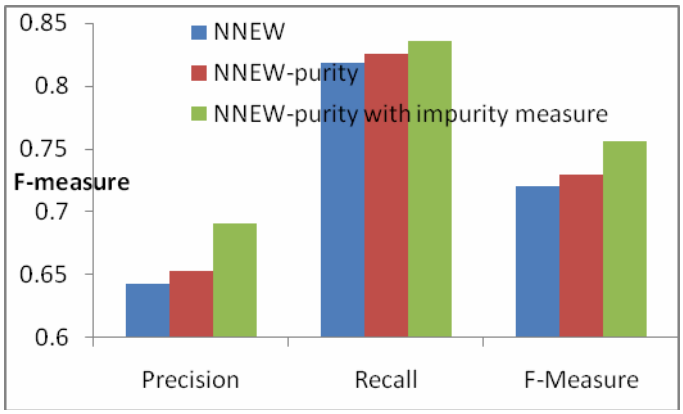

Figure 1 NNEW improvement

References

[1] Douglass R. Cutting, David R. Karger, Jan O. Pedersen, and John W. Tukey, Scatter/Gather "A Cluster-based Approach to Browsing Large Document Collections," SIGIR '92, Pages 318 – 329, 1992.

[2] Meenakshi Nagarajan, Karthik Gomadam, etc. "Spatio-Temporal-Thematic Analysis of Citizen-Sensor Data - Challenges and Experiences" Tenth International Conference on Web Information Systems Engineering, Oct 5-7, 2009, Poland