# A Micro blog Recommendation System Based on User Clustering

Lei Chen[1,a], Chao Jiang[1,b] and Wei Wang[1,c]

[1]Zhejiang Provincial Key Laboratory of Service Robot, College of Computer Science, Zhejiang University, Hangzhou 310027, China

[a]chenl1990@zju.edu.cn, [b]jiangchao.cs@gmail.com, [c]wangwei_eagle@zju.edu.cn

**Keywords:** micro blog, recommendation system, cluster, semantic dictionary, similarity

**Abstract.** Nowadays, micro blog has been widely used as a platform of information sharing. How to help users to find what they are interested in from massive amount of data becomes a very challenging issue. Some micro blog recommend systems are user-based and its effect is not significant. In this paper, we implement the recommendation system based on micro blog, users can subscribe the post which they are interested in, and system will recommend the related posts to users combined with user interest. This paper cluster users by their interest and experimental results show that the improved K-Means clustering algorithm can achieve a better accuracy and running time and our algorithm is more effective than traditional algorithm.

## Introduction

Micro blog is a social networking platform which supports real-time information sharing. Due to the rapid development of Internet technology, microblog has been widely used, and users continued to grow in recent years.But now for the vast majority of microblog, users often can not get the information they want, mainly due to the restrictions of content words and the search mechanism is not perfect in microblog. Micro blog recommend methods which generally recommending users are not effective.

This paper introduce a method based on microblog content which first do the segmentation of the content published by user offline and extract keywords to establish interest model, then do the clustering of users. When doing the recommendation, we simply select the clustering results published in the same cluster, and recommend according to the similarity of the content of micro blog.

## Word segmentation and TF-IDF

After extracting the keywords, we using TF-IDF vector space model. The main idea of this model includes word frequency (TF) and inverse text frequency (IDF). Word frequency represents the frequency of occurrence of a word in an article, and the inverse document frequency represents the ability of distinguishing between the documents.

The formula of TF-IDF:

$$\mathrm{TF\text{-}IDF}(w_i) = \mathrm{TF}(w_i) * \mathrm{IDF}(w_i) \tag{1}$$

$$TF_{ij} = \frac{f_{i,j}}{\sum_z f_{z,j}} \tag{2}$$

$$\mathrm{IDF}_i = \log_2\left(\frac{n}{n_i}\right) \tag{3}$$

Where $fi,j$ represents number of occurrences of the word $i$ in document $j$, n represents the number of documents in the current, $ni$ represents the number of document that includes word $i$.

 

**Interest modeling and interests merge**

We describe user interset model through TF-IDF model,and we can get a word set WORD={*w1,w2,....wn*},and its weight vector Weight={*w1,w2,...wn*}.As the user's interests may change gradually, the interest to a thing may gradually decreased over time, so here introduce an attenuation factor q, and merge the interest in different times.

Suppose we have a list of the previous interest WORD' and the corresponding weights W', and get a list of the current interest WORD and the corresponding weights *W*, we may face three conditions when we want to merge them:

1 *wi* appears in both current and previous list, then update *Pi = Pi * q + Pi '* (1-q).*

2 *wi* appears in only previous list, then update *Pi = Pi '* (1-q).*

3 *wi* appears in only current list, then update *Pi=Pi* and add wi in WORD list.

After interest merging, we can use a matrix *M* to represent the user's attention to keyword.the element in row *i* column *j* in M indicates that score of user *i* to word *j*.

**K-means and improved K-means**

K-means is a clustering algorithm which is widely used, and it is necessary to give a parameter k denotes the k classes after clustering. Suppose n users are clustered into k classes, the main idea is: Randomly select k users as the central of a class and calculate the distance between no classification users and every center of class, each user will be assigned to his nearest class, and then update the location of the center of each class, and so on until convergence.

Define Euclidean distance between two m-dimensional vector *xi = (xi1, xi2... xim)* T and *xj = (xj1, xj2... xjm) T.*

$$\text{dis(xi, xj)} = |\text{xj - xi}| = \sqrt{\sum_{u \in I} | x_{iu-} x_{ju} |} \tag{4}$$

Although the effect of Kmeans is significant when the classes' distinction is obvious, there are also two drawbacks:

1 Kmeans need to determine the value of K, and this value is generally difficult to determine.

2 The select of initial point of the center have a huge impact on the clustering effect.

To the drawback 2, this paper introduces an algorithm which chooses the center of class before doing the Kmeans.

**Definition 1**: The center of two n-dimensional data *X=(x1, x2… xn)* and *Y=(y1, y2, ...,yn)* is

$$\text{CE(x, y)} = (\frac{x_1 + y_1}{n}, \frac{x_2 + y_2}{n}, ..., \frac{x_N + y_N}{n}) \tag{5}$$

**Definition2**: The average distance between two points is

$$Mdis = \frac{\sum_{i=1}^{n} \sum_{j=i}^{n} D(x_i, x_j)}{C_n^2} \tag{6}$$

The algorithms of choosing the center of class can be performed like this:

1 Calculate the average distance Mdis and randomly select one point, as an initial cluster center.

2 find a point which must satisfy the condition that the distance between it and the center of found class center is larger than Mdis

3 add this point to the class center set and update the center of the class center.

4 if the number of found cluster center is less than K, back to step 2, else end this algorithm.

**Result analysis**

In this paper, the selected data set contains 400 data points, and each point contains five properties, divided into five categories. The results are as follows:

Table 1: the result of K-Means

| K-Means | | | | |
|---|---|---|---|---|
| Experiment number | Number of data | Number of right case | Accuracy(%) | Runtime(s) |
| 1 | 100 | 83 | 83 | 0.312 |
| 2 | 200 | 168 | 84 | 0.547 |
| 3 | 300 | 250 | 83.3 | 0.781 |
| average | | | 83.4 | 0.547 |

Table 2: the result of improved K-Means

| Improved K-Means | | | | |
|---|---|---|---|---|
| Experiment number | Number of data | Number of right case | Accuracy (%) | Runtime(s) |
| 1 | 100 | 89 | 89 | 0.284 |
| 2 | 200 | 181 | 90.5 | 0.469 |
| 3 | 300 | 270 | 90 | 0.697 |
| average | | | 89.83 | 0.492 |

Because the improved Kmeans algorithms choose better class center so it led to the less runtime in convergence.It is obvious that the improved Kmeans is better than tradition Kmeans not only in runtime but also in accuracy.

**Similarity calculation of microblog**

There are concepts and sememe in hownet, every word has many concepts and sememe is the minimal unit of concept, hierarchy between the sememe is the basis for calculating similarity.A sememe hierarchy is as follows:

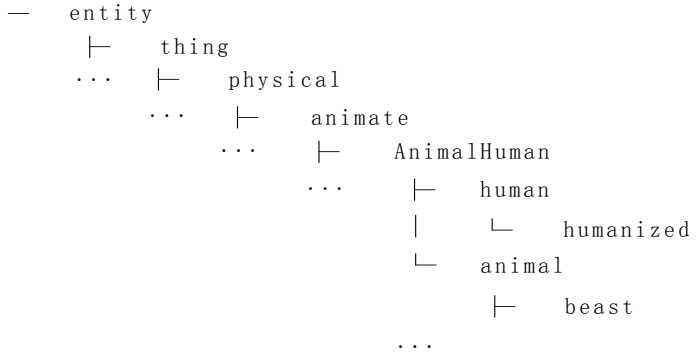


Fig. 1: A sememe hierarchy

**Definition3**: the similarity between two words

$$sim(A,B) = \max_{i,j} sim(ai,bj) \tag{7}$$

Where A has n sememe *a1, a2 ... an* and word B has m sememe *b1, b2 ...bm, and* the similarity between two sememes is the quotient of the length difference *p* in sememe hierarchy plus *q* and *q*, and *q* is a variable parameter.

**Definition4**: the similarity between two microblog

$$S_{AB} = \frac{\sum_{i=1}^{L} S_{\max i}}{L} \tag{8}$$

Where *Smax1,* Smax2 ...*SmaxL* are largest element from the similarity matrix and *L* is a parameter which can be adjusted.

**The result of recommendation**

To verify our algorithms, this paper chooses 5 topics and calculates the recall and precise in both traditional recommend algorithm and microblog content algorithm, results are as follows:

Table 3: the result of micro blog content algorithm

| | Recall R（%） | | Precise P（%） | |
|---|---|---|---|---|
| topics | Traditional recommend algorithms | microblog content algorithm | Traditional recommend algorithms | microblog content algorithm |
| 1 | 29 | 33.5 | 76.4 | 86.7 |
| 2 | 25 | 28.5 | 69.1 | 80 |
| 3 | 31 | 35 | 85.6 | 93.3 |
| 4 | 29 | 30.5 | 77.7 | 80.4 |
| 5 | 28.5 | 30 | 74.5 | 80.3 |

It is obvious to see microblog content algorithm has significantly improved in the recall and precision compared to the traditional algorithm.

## Conclusion

To the issue of recommend effect in microblog is not significant ,this paper introduce a method based on microblog content,through word segmentation,interest model establishing and emerging, user clustering and similarity calculation,the result is better than traditional algorithm.However,there is room for improvement in this paper,like more improvement in Kmeans and more accurate similarity calculation.

## Acknowledgement

This work is supported by National Key Technology R&D Program (Grant No. 2012BAI34B01)

## References

[1] Vance Faber. Clustering and the Continuous K-Means Algorithm [J]. Los Alamos Science, 1994, 22:138-144.

[2] Salton G. Automatic Text Processing [M]. Addison-Wesley, 1989.

[3] Breese JS, Heckerman D, Kadie C. Empirical analysis of predictive algorithms for collaborative filtering[C]. Proc. 14th Conf. Uncertainty in Artificial Intelligence, 1998, 7.

[4] Salton G. The smart retrieval system-experiments in automatic document processing [M]. Upper Saddle River: Prentice-Hall, 1971:207-214.

[5] Chen Y H,George E I,A bayesian Model for Collaborative Filtering[C]. Process of the 7th International Workshop on Artificial Intelligence and Statistics, 1999.

[6] Ungar L H, Foster D P. Clustering Methods for Collaborative Filtering[C]. Process of Workshop on Recommendation System, 1998.

[7] Billsus D, Pazzani M, Learning Collaborative Information Filters[C]. Process of International Conference on Machine Learning, 1998:46-54.