A Feature Selection Method to Handle Imbalanced Data in Text Classification

Fengxiang Chang\(^1\)*, Jun Guo\(^1\), Weiran Xu\(^1\), Kejun Yao\(^2\)
\(^1\)School of Information and Communication Engineering
Beijing University of Posts and Telecommunications,
Beijing, 100876, China
\(^2\)Computing and Data Processing Center
GRUEEX Ltd., Zagreb, Croatia
haixiang189@163.com

ABSTRACT: Imbalanced data problem is often encountered in application of text classification. Feature selection, which could reduce the dimensionality of feature space and improve the performance of the classifier, is widely used in text classification. This paper presents a new feature selection method named NFS, which selects class information words rather than terms with high document frequency. To improve classifier performance further, we combine a feature selection method (NFS) with data resampling technology to solve the problem of imbalanced data. Experiments were evaluated on Reuters-21578 Collection, and results show that the NFS method performs better than chi-square statistics and mutual information on the original dataset when the number of selected features is greater than 1000. The maximum value of Macro-F1 is 0.7792 when the NFS method is applied to the resampling dataset, which represents an increase in Macro-F1 by 4.02% given the original dataset. Thus, our proposed method effectively improves minority class performance.

Subject Categories and Descriptors
I.2.7 [Artificial Intelligence]: Natural Language Processing - Text Analysis; I.5.4 [Pattern Recognition]: Applications - Text processing

General Terms: Algorithm, Performance

Keywords: Text classification, Feature selection, Imbalanced data

Received: 22 January 2015, Revised 2 March 2015, Accepted 11 March 2015

1. Introduction

The volume of digital documents available online is progressively increasing, and text classification is the key technology used to process and organize text data. Nonetheless, a major problem in text classification is that the high dimensionality of the feature space can easily increase the number of features to hundreds of thousands. This increase can deteriorate classifier performance as a result of redundant and irrelevant terms. An effective way to solve this problem is by reducing feature space dimensionality [1–2].

Dimension reduction methods include feature extraction and selection. In feature extraction, a new feature set is generated by combining or transforming the original dataset. In feature selection, a subset is derived from the original set without transforming the feature space. Three methods are used to perform feature selection: embedded, wrapper and filter. Embedded and wrapper methods rely on a learning algorithm, but filter method is independent of such algorithms. As the filter method is sampler and has lower computational complexity than the other two methods, it has been widely used in text classification. Many filter methods have been proposed, including the improved Gini index, information gain (IG), chi-square statistics (CHI), document frequency (DF), orthogonal centroid feature selection (OCFS), and DIA association factor (DIA) [3]. In fact, the new feature selection (NFS) method proposed in the current paper is a filter method as well.

Problems of imbalanced data are often observed in text classification because the number of positive samples is usually considerably smaller than that of negative samples. Imbalanced data generally cause classifiers to perform poorly on the minority class. The final aim of text classification on imbalanced datasets involves improving the minority-class classification performance without affecting the overall classification performance of the
classifier. A number of solutions have been proposed for this problem at both data and algorithmic levels. Data-level solutions include oversampling on minority-class samples and undersampling on majority-class samples. Algorithmic-level solutions include the design of new algorithms or the optimization of original algorithms to improve classifier performance.

Feature selection should be more important than classification algorithms in highly imbalanced situations [4]. Selecting a word with strong class information can improve classifier performance; for example, the word “football” usually appears in the class “sports” [5]. In this paper, we propose an NFS method that differs from many existing feature selection methods. Using this method, we select terms with class information; then, we combine the NFS method with data resampling technology to improve imbalanced classification. The effectiveness of our proposed method is proved by experiment results.

2. Method

Many feature selection methods have recently been used extensively in text classification [1–8], such as mutual information (MI), IG, and CHI. CHI and MI are considered in this study and are described as follows:

Let \( t \) be any word; then, we can define its presence and absence in class \( c_i \) as:

\[ A_i \] is the number of documents with word \( t \) that belongs to class \( c_i \);

\[ B_i \] is the number of documents without word \( t \) that belongs to class \( c_i \);

\[ C_i \] is the number of documents with word \( t \) that does not belong to class \( c_i \); and

\[ D_i \] is the number of documents without word \( t \) that does not belong to class \( c_i \).

The CHI and MI metrics for a word \( t \) are defined by

\[
CHI(t, c_i) = \frac{N(A_i D_i - B_i C_i)^2}{(A_i + C_i)(A_i + B_i)(B_i + D_i)(C_i + D_i)} \tag{1}
\]

\[
MI(t, c_i) = \log \frac{A_i N}{(A_i + C_i)(A_i + B_i)} \tag{2}
\]

where \( N \) is the total number of documents and can be defined as \( N = A_i + B_i + C_i + D_i \).

The documents that belong to class \( c_i \) are defined as positive documents, whereas those that do not belong to class \( c_i \) as regarded as negative documents.

### 2.1 Feature Selection Method Given Imbalanced Datasets

The proposed feature selection method is based on the following hypothesis:

1. The more a word \( t \) occurs in class \( c_i \) documents, the better it can represent this class.

2. If a word \( t \) is mainly detected in most positive documents and rarely occurs in negative documents, then the word exhibits strong class discriminative power.

3. The higher the inverse class frequency of a word \( t \) is, the greater the contribution of the word to classification.

Inverse class frequency is defined as follows:

\[
ICF_{t, c_i} = \frac{1}{n} \cdot \frac{n_{t}}{n_i} \tag{3}
\]

where \( n \) is the total number of the classes and \( n_i \) is the total number of the classes in which the word \( t \) appears.

The NFS algorithm is defined as

\[
NFS(t, c_i) = \frac{A_i}{B_i} \cdot \frac{A_i}{C_i} \cdot \frac{1}{n} \cdot \frac{n_{t}}{n_i} \tag{4}
\]

### 2.2 Data Resampling on Imbalanced Data

To overcome the imbalance problem in text classification and to improve classifier performance, data resampling technology is applied to imbalanced data.

Solutions to address this issue include oversampling and undersampling. Alternatively, both methods can be integrated to lower the degree of imbalanced distribution. Oversampling techniques include random and synthetic minority oversampling techniques (SMOTE). Undersampling techniques include Tomek links, edited nearest neighbor rule, and cluster-based undersampling [9]. In the present study, SMOTE is used to overcome the problem of imbalanced data.

In this approach, the minority class is oversampled by generating “synthetic” examples as follows: taking the difference between the sample under consideration and its nearest neighbor, multiplying this difference by a random number between 0 and 1, and then adding the resultant value to the sample under consideration. The algorithm is presented as follows:

![Figure 1. Synthetic examples](image)
Algorithm SMOTE \((T, N, k)\)

**Input:** Number of minority class samples \(T\); Amount of SMOTE \(N\%\); Number of nearest neighbors \(k\)

**Output:** \((N/100)*T\) synthetic minority class samples

1. \((* \text{If } N \text{ is less than } 100\%, \text{randomize the minority class samples as only a random percent of them will be SMOTEEd.} *)\)
2. if \(N < 100\)
3. then randomize the \(T\) minority class samples
4. \(T = (N/100)*T\)
5. \(N = 100\)
6. endif
7. \(numattrs = \text{Number of attributes}\)
8. \(Sample[ ][ ]: \text{array for original minority class samples}\)
9. \(newindex: \text{keeps a count of number of synthetic samples generated, initialized to 0}\)
10. \(Synthetic[ ][ ]: \text{array for synthetic samples}\)
   \((* \text{Compute } k \text{ nearest neighbors for each minority class sample} *)\)
11. for \(i \leftarrow 1\) to \(T\)
12. Compute \(k\) nearest neighbors for \(i\) and save the indices in the \(nnarray\)
13. \(Populate(N, i, nnarray)\)
14. End for
15. \(Populate(N, i, nnarray) (* \text{Function to generate the synthetic samples} *)\)
16. while \(N \neq 0\)
17. Choose a random number between 1 and \(k\) and call it \(nn\). This step chooses one of the \(k\) nearest neighbors of \(i\).
18. for \(attr \leftarrow 1\) to \(numattrs\)
19. Compute: \(dif = Sample[nnarray][nn][attr] - Sample[i][attr]\)
20. Compute: \(gap = \text{random number between 0 and 1}\)
21. \(Synthetic[newindex][attr] = Sample[i][attr] + gap*dif\)
22. \(newindex++\)
23. \(N=N-1\)
24. end while
25. \(\text{return} (* \text{End of Populate} *)\)

The algorithm is divided into three parts: randomizing the minority class samples if \(N < 100\) (lines 2–6); computing \(k\) nearest neighbors (KNN) for each minority class sample (lines 11–14); and generating synthetic samples (lines 15–25) \(\text{http://www.cs.cmu.edu/afs/cs/project/jair/pub/volume16/chawla02a-html/node6.html}\).

Figure 1 shows synthetic examples generated by SMOTE. In this figure, \(x_i\) represents the \(i^{th}\) minority class sample; \(x_{i1}, x_{i2}, x_{i3}\), and \(x_i\) are the four nearest neighbors of \(x_i\); and \(s_1, s_2, s_3,\) and \(s_4\) are synthetic samples.

3. Experiments and Analysis

3.1 Data Collection

To evaluate the performance of the proposed method, we employed the Reuters-21578 Collection in the experiment. Modapte split is selected. The collection contains 135 categories in the original version; however, we considered only a subset of the top 10 categories in this study.

### 3.2 Classifier

Many classification algorithms are available for text classification, such as naïve Bayes, KNN, and support vector machine (SVM). In this experiment, the SVM method is used due to its simplicity and accuracy in this type of classification.

This method \([10–11]\) is a popular machine learning method for data classification. SVM distinguishes classes by
creating a decision boundary as a hyperplane that separates documents of different classes.

The solution can be considered an optimization problem,

$$\min \frac{1}{2} w^T w + C \sum_{i=1}^{n} \xi_i$$

$$y_i((w^T x + b) \geq 1 - \xi_i),$$

(5)

$$\xi_i \geq 0, 1, \ldots n$$

where $C$ is a penalty parameter and $\xi$ is a loss function. Additional details are provided in the works conducted by Vladimir and Vapnik (1995) and Vapnik (1998). A library for SVM toolkit (Chang & Lin, 2001) was used in the current study, as was the radial basis function kernel.

### 3.3 Evaluation Methodology

Micro- and Macro-F1 were employed to evaluate the effectiveness of our method. F1 is measured on the basis of precision and recall values; for a given class $c_i$, these variables are defined as

$$p(c_i) = \frac{TP_i}{TP_i + FP_i}$$

(6)

$$r(c_i) = \frac{TP_i}{TP_i + FN_i}$$

(7)

where $TP_i$ is the number of documents that are correctly classified to class $c_i$; $FP_i$ is the number of documents that are incorrectly classified to class $c_i$; and $FN_i$ is the number of documents belonging to class $c_i$ that have been incorrectly classified to other classes.

Micro-F1 denotes the F1 measurement computed over global precision and recall. This measurement can be determined with

$$P_{\text{micro}} = \frac{\sum_{i=1}^{C} TP_i}{\sum_{i=1}^{C} TP_i + \sum_{i=1}^{C} FP_i}$$

(8)

$$R_{\text{micro}} = \frac{\sum_{i=1}^{C} TP_i}{\sum_{i=1}^{C} TP_i + \sum_{i=1}^{C} FN_i}$$

(9)

$$F1_{\text{micro}} = \frac{2P_{\text{micro}}R_{\text{micro}}}{P_{\text{micro}} + R_{\text{micro}}}$$

(10)

where $C$ is the total number of classes.

Macro-F1 is the average of the F1 values for each class, i.e.,

$$\sum_{i=1}^{C} \frac{2p(c_i)r(c_i)}{p(c_i) + r(c_i)}$$

$$F1_{\text{macro}} = \frac{C}{C}$$

(11)

Micro-F1 is influenced by classifier performance on common classes; by contrast, Macro-F1 is dominated by rare classes [12].

### 3.4 Experimental Results

#### 3.4.1 Effectiveness of the NFS Method on the Original Dataset

We compare our feature selection method with two typical feature selection methods, i.e., CHI and MI. Figure 2 shows the results for Micro-F1 given imbalanced data without resampling. CHI performs well given small feature sets of 500 or less. However, our method is a favorable option when the feature sets are increased. The NFS method performs better than both CHI and MI when the number of feature sets is greater than 1000; in fact, performance is maximized when the number of selected features is 2000.

![Figure 2. Micro-F1 results](image)

Table 1 presents detailed results. The bold value in the table indicates that it is the best performance of the classifier among these feature selection methods. Optimum classifier performance is observed with NFS (0.8579).

Table 3 displays the measured results for Macro-F1. The Macro-F1 based on NFS method is especially suitable for high-feature sets. Although this method performs poorly given small feature set, it generates maximum value when the number of feature sets is 2000. CHI performs well given small feature sets; however, MI is inferior to the CHI and NFS methods in all of the feature sets in the experiment.

Table 2 presents the detailed results for Macro-F1. The best performance was obtained with NFS methods as 0.7390. The NFS method performed better than CHI when the numbers of selected features were 1000, 1500, and 2000. MI performed poorly, particularly given small feature sets. The performance obtained with MI was only 0.1455 when the number of selected features was 100.

#### 3.4.2 Effectiveness of Data Resampling Technology on Imbalanced Data

To further verify the efficiency of our proposed method, we compare the performance levels of the classifiers for the
original and resampling datasets (the minority class is oversampled with SMOTE). The NFS method alone is used in this experiment.

Table 3 shows the classification performance given the original and resampling datasets when the NFS method is employed.

<table>
<thead>
<tr>
<th>Feature selection methods</th>
<th>Number of features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>100</td>
</tr>
<tr>
<td>MI</td>
<td>0.6475</td>
</tr>
<tr>
<td>CHI</td>
<td>0.7514</td>
</tr>
<tr>
<td>NFS</td>
<td>0.7513</td>
</tr>
</tbody>
</table>

Table 1. Micro-F1 results

![Image](image1.png)

Figure 3. Macro-F1 results

<table>
<thead>
<tr>
<th>Feature selection methods</th>
<th>Number of features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>100</td>
</tr>
<tr>
<td>MI</td>
<td>0.1455</td>
</tr>
<tr>
<td>CHI</td>
<td>0.5781</td>
</tr>
<tr>
<td>NFS</td>
<td>0.3246</td>
</tr>
</tbody>
</table>

Table 2. Macro-F1 results

<table>
<thead>
<tr>
<th>Original dataset</th>
<th>Resampling dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of features</td>
<td>Micro-F1</td>
</tr>
<tr>
<td>500</td>
<td>0.8247</td>
</tr>
<tr>
<td>1000</td>
<td>0.8528</td>
</tr>
<tr>
<td>1500</td>
<td>0.8569</td>
</tr>
<tr>
<td>2000</td>
<td>0.8579</td>
</tr>
</tbody>
</table>

Table 3. Performance of the SVM classifier when the NFS method is used on the original and resampling datasets
According to Table 3, the Micro-F1 performance of SVM on the resampling dataset is superior to that on the original dataset in all feature sets. The maximum Micro-F1 value is 0.8721 and is obtained given the resampling dataset when the number of selected features is 2000. Macro-F1 performance improved when we increased the number of features. The maximum Macro-F1 value is 0.7792; this value is obtained on the resampling dataset as well and denotes an improvement in Macro-F1 by 4.02% over the original dataset.

Figure 4 and Figure 5 plot all of the Micro- and Macro-F1 values from Table 3 to provide a general view of the results. Classifier performance improves given the resampling dataset in all of the feature sets considered in the experiment, particularly with respect to Macro-F1 values.

The comparison results confirm that the data resampling technology used (SMOTE) can overcome the imbalance problem and improve minority class performance.

4. Conclusion

An NFS method is proposed in this paper. Unlike general feature selection methods, such as MI and CHI, the NFS method chooses class-informative words rather than high-frequency words. The experiment results highlight the effectiveness of this method, whose performance is compared with that of MI and CHI. The findings show that the proposed method performs better than the other two when the number of selected features is above 1000.

To improve classifier performance further, we combine a feature selection method (NFS) with data resampling technology (SMOTE) to handle the imbalance problem in text classification. The experiment demonstrates the effectiveness of our proposed method. Moreover, Macro-F1 performance in the resampling dataset increased by 4.02% over that obtained in the original dataset when the NFS method was applied and when the number of selected features was 2000.

Figure 4. Micro-F1 results

Figure 5. Macro-F1 results

Acknowledgement

This work was supported by 111 Project of China under Grant No.B08004, key project of ministry of science and technology of China under Grant No. 2011ZX03002-005-01, National Natural Science Foundation of China (61273217) and the Ph.D. Programs Foundation of Ministry of Education of China (20130005110004).

References


