Chinese text classification method using FastText and term frequency-inverse document frequency optimization

Tiantian Zhou1*, Yintong Wang1,a, Xin Zheng1
1School of Information Engineering, Nanjing Xiaozhuang University, Nanjing 211171, China
*a zhou_tiantianzt@163.com
awyt@njxzc.edu.cn

Abstract. With the development of information technology, obtaining information quickly and accurately has become an indispensable part of people's lives. Text classification can filter and organize massive text data to find valuable information, and its practical application is of great significance. This paper proposes Chinese text classification method using FastText and term frequency-inverse document frequency optimization(I-FastText). This method introduces term frequency-inverse document frequency(TF-IDF) optimization into the input layer of FastText model, removes words with high frequency and low discrimination ability, and achieves high-quality word vectors to train the text classification model. The experimental results on the THUCNews dataset shown that the Chinese text classification accuracy of I-FastText is significantly better than TF-IDF and FastText methods.

1. Introduction

With the rapid development of Internet technology, the Internet has become the most convenient and effective carrier for information dissemination. In the face of the massive data generated by the Internet, it has become more and more important for people to obtain effective information. Text classification technology can achieve effective screening and management of text data, and plays an important role in practical applications such as information retrieval, automatic summarization and fraud detection. At the same time, in the fields of human-computer communication and artificial intelligence related to natural language processing, many studies have taken text classification as the basis to facilitate the screening, sorting and retrieval of text dataset[1]. Text classification is performed by a computer of the target text according to certain classification system and standard auto-tagging technology.

The original text classification methods are realized based on semantic rules, which is more suitable for text classification with obvious feature words. Text classification based on machine learning mainly include naive Bayes, K-nearest neighbour and support vector machines. The naive Bayes method is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions[2]. The K-nearest neighbour method is a lazy learning algorithm based on statistics, the K value is small, clusters number is small, and classification accuracy is not high[3]. Support vector machine is a machine learning method based on statistical learning theory, uses linear function hypothesis space in high-dimensional feature space[4, 5]. Deep learning is a type of machine learning that trains a computer to perform human-like tasks, sets up basic parameters about the data and trains the computer to learn on its own by recognizing patterns using many layers of processing. Liu et al. proposed convolutional neural networks text classification method, through different sizes of...
kernels extract the key information in the sentence, to better capture the local relevance of the information [6]. Bahdanau et al. proposed recurrent neural network text classification method, experimental results shown that it can better express the contextual information of text [7]. Joulin et al. proposed attention mechanism recurrent neural network method, which more intuitively reflects the importance of each sentence and word to the classification category [8]. However, the test and training speed of this model is slow, which limits its application. Yoshikawa proposed FastText text classification method, its main advantages including fast model training, low cost and high efficiency [9, 10]. Further analysis shows that the n-gram processing in FastText obtains the word vectors with low-frequency and meaningless. These words will increase sharply with the number of characters in the text, which will eventually affect the text classification accuracy.

In response to the above problems, this paper proposes Chinese text classification method using FastText and term frequency-inverse document frequency optimization. The main work includes reconstructing the FastText input layer, removing meaningless words in the word vector obtained by n-gram processing, and calculating and sorting the FT-IDF values of each word to filter the words with high frequency and low discrimination ability.

2. FastText model

The fastText method is a supervised model, similar to the CBOW architecture of word2vec [11, 12]. Its model has three layers: input layer, hidden layer and output layer, as shown in Figure 1. Input is a number of words and their n-gram features, these features are used to represent a single document, the hidden layer is the superimposed average of multiple feature vectors, it solves the maximum likelihood function, and then constructs a Huffman tree according to the weights and model parameters of each category, and uses the Huffman tree as the output. In addition, FastText can achieve good results and fast speed, mainly because of two important factors, firstly using the sub-word n-gram information, and secondly using a Huffman coding tree-based hierarchical Softmax method.

![Fig. 1 The structure of FastText model](image)

In addition, from the structure of the FastText model in fig. 1, the text data be converted to word vector through N-gram word segmentation technology; the word features learned by the hidden layer need to be realized from word features to text categories through the Softmax mapping function.

2.1. N-gram Features

FastText introduces the concept of subword n-gram to solve the problem of word order loss, splits a word into character level and uses character-level n-gram information to capture the order relationship between characters. Such as, a word vector \{w_1, w_2, w_3\}, which size is 3. A word vector \{w_1, w_2, w_3, w_12, w_23\} obtained by using 2-gram, which size is 5. As we known that this word vector can
retain the order of the original word vector. And then, the average value of all word vectors is obtained through the hidden layer, the calculation formula is \( h = \frac{1}{5}(w_1 + w_2 + w_3 + w_{12} + w_{23}) \).

2.2. Layered SoftMax
Softmax is a generalization of logistic regression on multi-classification tasks and is the last layer in training neural networks. When the vocabulary number is large, the Softmax calculation is very costly. fastText uses a layered Softmax based on the Huffman tree, where each leaf node in the tree represents a text category. The model calculates the probability from the root node to each leaf node and selects the highest probability as the target category, then its probability is:

\[
p(w_i) = \prod_{j=1}^{(i-1)/2} \sigma(\text{sign}(w_i, j) \cdot \theta_{n(w_i,j)}^T h)
\]

where, \( \theta_{n(w_i,j)} \) represents the vector of the non-leaf node \( n(w_i,j) \), as the output vector. \( h \) represents the output value of the hidden layer, which is calculated from the vector of the input word. \( \text{sign}(w_i, j) \) represents a special function, its value is \( \{-1, 1\} \).

3. Chinese text classification method

3.1. Basic Conception
Word segmentation is considered an important first step for Chinese natural language processing tasks, Chinese word can be composed of multiple characters without any space. It's obvious that high-quality word segmentation can significantly improve the text classification accuracy. In the FastText, the word vector processed by N-gram is directly used as the input data of the model. When the character number in Chinese text is large, the length of the word vector is tremendous, which eventually leads to the failure of text classification due to the excessive parameters. To solve this problem, this paper introduces TF-IDF to optimize the word vector between the input layer and the hidden layer of the FastText model, calculates the weight value of each word vector in the pre-processed text, and sorts them in descending order by their weights, selects a pre-set number of word vectors as the input data of FastText model.

TF-IDF is an information retrieval technique that weighs a term’s frequency (TF) and its inverse document frequency (IDF) [13]. It is used to weigh a keyword in any content and assign the importance to that keyword based on the number of times it appears in the document. More importantly, it checks how relevant the keyword is throughout the whole corpus. Generally, the importance of a word increases in proportion to the number of times it appears in the document, but it decreases in inverse proportion to the frequency of its appearance in the corpus. Each word or term has its respective TF and IDF score. The product of the TF and IDF scores of a term is called the TF × IDF weight of that term.

TF represents the frequency of words in the text. Usually, the frequency needs to be normalized to avoid bias in the long text. Its formula is as follows:

\[
tf_y = \frac{n_{i,j}}{\sum_k n_{k,j}}
\]

where, \( n_{i,j} \) represents the number of times the \( i-th \) word appears in the \( j-th \) text. It can be seen from equation (2) that the length of the \( i-th \) text affects the size of \( n_{i,j} \), so the total words in the corpus is needed to normalize the word frequency.

IDF is obtained by dividing the number of texts in the corpus by the number of texts in the word, and then taking the logarithm of the obtained quotient. For a word, the less text that contains the word, the larger the IDF, which also shows that the word has a good classification ability. Its formula is as follows:

\[
idf_i = \log \frac{|D|}{|\{j: t_i \in d_j\}| + 1}
\]
where, $|D|$ represents the number of texts in the corpus, and $|\{j : t_i \in d_j\}|$ represents the number of texts containing word $t_i$. Assuming that the word $t_i$ comes from the corpus, and the word may not appear in some texts, it needs to adopt the form of adding 1 to the denominator in equation (3).

In summary, I-FastText is based on the traditional FastText method, using TF-IDF to analyze the weight of the words obtained by n-gram processing, and retain the word vector with a large weight value as a model input data. The advantage of this algorithm is to optimize and remove some words with high frequency and low distinguishing ability, so the retained word vectors input into the model has a higher classification ability.

### 3.2. Method Implementation

The I-FastText reconstructs the input layer of the traditional FastText model, filters the n-gram processed data to reduce the meaningless words, forming high-quality text data into the hidden layer of the model. Its implementation is including four parts: (1) Text preprocessing, operations on the text data in the corpus, Chinese word segmentation and stop words removal, and then divide the corpus data set into train data and test data. (2) Word vector optimization, splits a word into character level and uses character-level n-gram information to capture the order relationship between characters, the weight value is calculated by the TF-IDF, and the words after the weight value calculation are formed into the input data. (3) Model training, parameters pre-set including learning rate ($lr$), training vector dimension layer ($dim$), loss function ($loss$). The data is input into the algorithm model for model training, and the hierarchical SoftMax is used in the input layer to obtain the category label of the short text, and the model parameters are optimized until the training conditions are met. (4) Text category prediction, using the trained model, the test data is used to evaluate the classification effect of the improved model through three model evaluation criteria: accuracy, recall, and F value. The specific algorithm implemented is as follows.

**Algorithm 1: I-FastText algorithm**

```
1. def I-FastText_Implementation ( txt )
2.   train, test = fun_process(txt) //text preprocessing
3.   for text in train
4.     n = n-gram (text) //n-gram processing
5.     for word in n // TF-IDF optimization
6.       Wi = sorted (calculate word TF-IDF value) //weight calculation and sorting
7.       wi = Wi[i * int(len(Wi)*0.25)] //select the top 25% word vector
8.     if Wi > withen add word to input //compose input data
9.   model = FastText (input) //FastText model training
10.  output = softmax (test) //text category prediction
11.  return output //return classification results
```

### 4. Experiment Results and Analysis

#### 4.1. Experiment preparation

To evaluate the Chinese text classification effectiveness, we compare I-FastText against two well-known text classifiers, including TF-IDF and FastText. All the methods are on the THUCNews Chinese news text data, which can be obtained from the Natural Language Processing Laboratory, Tsinghua University. According to the historical data of Sina News RSS subscription channel from 2005 to 2011, 14 candidate categories. In order to better evaluate the text classification accuracy of the three algorithms, we divide the THUCNews dataset into four sub datasets, including Dataset1, Dataset2, Dataset3 and Dataset4. Among them, Dataset1 contains financial, lottery and realestate three categories; Dataset2 contains stock, home and education three categories; Dataset3 contains technology, society, fashion and politics four categories; Dataset4 contains sports, constellations, game and entertainment four categories. Each category selected 6000 articles for this experiment. Each dataset is divided into 90%
training set, 6% verification set and 4% test set using numpy tool to realize cross-validation, that is to
group the original data set, one part is used for model training, the other part is used to verify the training
model, as the performance index of evaluation algorithm.

In addition, precision($P$), recall rate($R$) and F-measure($F$) are used as evaluation indexes in this paper. 
The specific formulas are as follows:

$$ P = \frac{TP}{TP+FP} \quad (4) $$

$$ R = \frac{TP}{TP+FN} \quad (5) $$

$$ F = 2PR / (P+R) \quad (6) $$

where, TP represents a positive Example predicted the correct number, FP represents the negative
predictive embodiments wrong format, TN represents the negative predictive embodiment the correct
number, FN represents the positive cases predicted number of errors.

4.2. Results and Discussion

In the experiment, the text classification algorithms based on TF-IDF, FastText and I-FastText are
implemented by python programming. Table 1 shows the text classification results of the three
algorithms, in which the black bold value represents the maximum value of the corresponding index in
the three methods. It can be seen that the I-FastText method proposed in this paper achieves significant
text classification accuracy on the four datasets, and the corresponding $F$ values are 84.47%, 85.79%,
84.29% and 84.75%, respectively, which are greater than the $F$ values of TF-IDF and FastText methods.

In addition, the average values of $P$, $R$ and $F$ of I-FastText are 85.95%, 85.17% and 84.83%, respectively,
which are better than other two methods.

| Datasets   | TF-IDF  | FastText | I-FastText |
|------------|---------|----------|------------|
|            | $P$     | $R$      | $F$        |
| Dataset1   | 77.96   | 75.02    | 76.51      |
| Dataset2   | 78.96   | 75.97    | 75.58      |
| Dataset3   | 78.47   | 74.82    | 76.60      |
| Dataset4   | 79.02   | 74.91    | 76.66      |
| Average    | 78.60   | 75.18    | 76.34      |

Figure 2 shows the comparison chart of $P$, $R$ and $F$ values of the three methods, in which the ordinate
is the calculation result of each index, the abscissa is different data sets. Figure 2(a), 2(b) and 2(c) are
the precision, recall rate and $F$ value comparison of the three methods under different datasets. It can be
seen from Figure 2 that I-FastText proposed in this paper is better than TF-IDF and FastText methods
in the three evaluation indicator values of $P$, $R$ and $F$ value.
5. Conclusion
In this paper, we have presented a Chinese text classification method using FastText and term frequency-inverse document frequency optimization. The method is based on the FastText, widely applied to text classification, calculating the weights of words by term frequency-inverse document frequency optimization and removing high frequency and low distinguishing ability words, and implementing the input layer reconstruction of FastText model. Experimental results shown that the I-FastText is effective in Chinese text classification compared to the TF-IDF and FastText methods.

Acknowledgments
This work was financially supported by the Natural Science Foundation of Jiangsu Province (BK20180142), the Innovation and Entrepreneurship Training Program for College Students in Jiangsu Province (201811460041X), and the Key Laboratory of Intelligent Information Processing Project of Nanjing Xiaozhuang University (19AIP02).

References
[1] Lei Zhong. Chinese text classification based on Bayesian classifier[J]. Electronic Technology and Software Engineering, 2016(22):156-156.
[2] Qingguo Zhang, Hongwei Zhang, Junyu Zhang. A fast text categorization approach based on k-nearest neighbor[J]. Journal of the Graduate School of Chinese Academy of Sciences, 2005(05): 554-559.
[3] Zhongbao Ma, Guanrong Liu. Research on Chinese text classification model based on SVM[J]. Computer Technology and Development, 2006,16(11):70-72.
[4] Yanhua Zhang, Wanghai Yong, Liying Zheng. A survey on techniques of text categorization based on support vector machines[J]. Journal of Gansu Sciences, 2006, 18(03): 72-74.
[5] Yoon Kim. Convolutional Neural Networks for Sentence Classification[C]. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, 2014 :1746–1751.
[6] Pengfei Liu, Xipeng Qiu, Xuanjing Huang. Recurrent Neural Network for Text Classification with Multi-Task Learning[C]. In Proceeding IJCAI'16 Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, 2016 :2873-2879.
[7] Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate[C]. In Proc.International Conference on Learning Representations, 2015: 1-15.
[8] Armand Joulin, Edouard Grave, et al. Bag of Tricks for Efficient Text Classification[C]. Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, 2016: 427-431.
[9] Yoshikawa Hiyori, Iwakura Tomoya. Fast Training of a graph boosting for large-scale text classification[C]. Pacific Rim International Conference on Artificial Intelligence. 2016 :638-650.
[10] Lingling Dai, Jiang Kan. Based fastText of Chinese text categorization[J]. Computer and Modernization, 2018, 273(5): 39-44,89.
[11] Bojanowski Piotr, Grave E, et al. Enriching word vectors with subword information[C]. Association for Computational Linguistics. 2017: 135-146.
[12] Tong, Simon, Daphne Koller. Support vector machine active learning with applications to text classification[J]. Journal of Machine Learning Research, 2001, 2: 45-66.
[13] Salton Gerard, Christopher Buckley. Term-weighting approaches in automatic text retrieval[J]. Information Processing & Management, 1988, 24(5):513-523.