Dimension Reduction with Coevolutionary Genetic Algorithm for Text Classification

Tatiana Gasanova[1], Roman Sergienko[1], Eugene Semenkin[2] and Wolfgang Minker[1]
[1]*Institute of Communications Engineering, Ulm University, Albert Einstein-Allee 43, 89081 Ulm, Germany*
[2]*Department of System Analysis and Operation Research, Siberian State Aerospace University, Krasnoyarskiy Rabochiy Avenue 31, 660014 Krasnoyarsk, Russia*

Keywords: Text Classification, Coevolutionary Algorithm, Text Preprocessing, Clustering, Dimension Reduction.

Abstract: Text classification of large-size corpora is time-consuming for implementation of classification algorithms. For this reason, it is important to reduce dimension of text classification problems. We propose a method for dimension reduction based on hierarchical agglomerative clustering of terms and cluster weight optimization using cooperative coevolutionary genetic algorithm. The method was applied on 5 different corpora using several classification methods with different text preprocessing. The method reduces dimension of text classification problem significantly. Classification efficiency increases or decreases non-significantly after clustering with optimization of cluster weights.

1 INTRODUCTION

Control systems are designed for complex problems which may be described with variables of different types: numerical, rank, qualitative, or text. The use of textual information is most challenging for incorporation into a control process, because the present-day methods of control system design are geared towards formalized data. One approach for formalization of text information is text classification that is transforming text information into categorical features.

Text classification can be considered to be a part of natural language understanding, where there is a set of predefined categories and the task is to automatically assign new documents to one of these categories. The method of text preprocessing and text representation influences the results that are obtained even with the same classification algorithms. The most popular model for text classification is vector space model. In this case text categorization may be considered as a machine learning problem. Complexity of text categorization with vector space model is compounded by the need to extract the numerical data from text information before applying machine learning methods. Therefore text categorization consists of two parts: text preprocessing and classification using obtained numerical data.

All text preprocessing methods are based on the idea that the category of the document depends on the words or phrases from this document. The simplest approach is to take each word of the document as a binary coordinate and the dimension of the feature space will be the number of words in our dictionary. There exist more advanced approaches for text preprocessing to overcome this problem such as TF-IDF (Salton and Buckley, 1988) and ConfWeight methods (Soucy and Mineau, 2005). A term weighting method (Gasanova et al., 2013) is also considered, which has some similarities with ConfWeight method, but has improved computational efficiency. It is important to notice that we use no morphological and stop-word filtering before text preprocessing. It means that the text preprocessing can be performed without expert or linguistic knowledge and that the text preprocessing is language-independent.

After text preprocessing we obtain a vector of numerical variables for each document and the dimension of the feature space is the number of words in the dictionary. In this case direct application of the machine learning algorithms is time-consuming. It is possible to use clustering of words in the dictionary for dimension reduction. Many researchers have used a variety of unsupervised techniques for various tasks in order to improve classification quality or to decrease dimension of the features. One common approach is to induce word features with unsupervised methods (for example, clustering which was used in (Miller et al., 2004), (Koo et al., 2008), (Ratinov and

Gasanova T., Sergienko R., Semenkin E. and Minker W..
Dimension Reduction with Coevolutionary Genetic Algorithm for Text Classification.
DOI: 10.5220/0005020702150222
In *Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics* (ICINCO-2014), pages 215-222
ISBN: 978-989-758-039-0

Roth, 2009), (Huang and Yates, 2009) or neural language models which have been proposed by (Bengio et al., 2006), (Schwenk and Gauvain, 2002), (Mnih and Hinton, 2007), (Collobert and Weston, 2008)) and then apply supervised algorithm. Turian et al. (Turian et al., 2010) have suggested learning unsupervised word features in advance without task or model related information and combine and integrate them into existing NLP systems. Despite an obvious advantage of this approach word features can be directly taken and used by many researchers the performance might not be as good as the one obtained by a semi-supervised algorithm which learns unsupervised word features using task-specific information as in the semi-supervised models such as in (Ando and Zhang, 2005), (Suzuki and Isozaki, 2008), and (Suzuki et al., 2009).

We proposed a method for dimension reduction based on clustering of terms. At first we apply hierarchical agglomerative clustering. After that we optimize weights of clusters with cooperative coevolutionary genetic algorithm (Potter and Jong, 2000).

In this paper we have used *k*-nearest neighbours algorithm, Bayes classifier, support vector machine (SVM), Rocchio classifier or Nearest Centroid Algorithm (Rocchio, 1971) and neural network as classification methods. *RapidMiner* has been used as implementation software.

For the application of algorithms and comparison of the results we have used the DEFT (Dfi Fouille de Texte) Evaluation Package 2008 (DEFT08, 2008) which has been provided by ELRA and publicly available corpora from DEFT07 (DEFT07, 2007).

The main aim of this work is to test the novel method of dimension reduction for text classification using different text preprocessing and different classification methods.

This paper is organized as follows: in Section 2, we describe details of the corpora. Section 3 presents text preprocessing methods. In Section 4 we describe the novel method for dimension reducing based on coevolutionary genetic algorithm. Section 5 reports on the experimental results. Finally, we provide concluding remarks in Section 6.

2 CORPORA DESCRIPTION

The focus of DEFT 2007 campaign is the sentiment analysis, also called opinion mining. We have used 3 publicly available corpora: reviews on books and movies (Books), reviews on video games (Games) and political debates about energy project (Debates).

The topic of DEFT 2008 edition is related to the text classification by categories and genres. The data consists of two corpora (T1 and T2) containing articles of two genres: articles extracted from French daily newspaper Le Monde and encyclopaedic articles from Wikipedia in French language. This paper reports on the results obtained using both tasks of the campaign and focuses on detecting the category.

Table 1: Corpora description (DEFT07).

Corpus	Size	Classes
Books	Train size = 2074 Test size = 1386 Vocabulary = 52507	0: negative, 1: neutral, 2: positive
Games	Train size = 2537 Test size = 1694 Vocabulary = 63144	0: negative, 1: neutral, 2: positive
Debates	Train size = 17299 Test size = 11533 Vocabulary = 59615	0: against, 1: for

Table 2: Corpora description (DEFT08).

Corpus	Size	Classes
T1	Train size = 15223 Test size = 10596 Vocabulary = 202979	0: Sport, 1: Economy, 2: Art, 3: Television
T2	Train size = 23550 Test size = 15963 Vocabulary = 262400	0: France, 1: International, 2: Literature, 3: Science, 4: Society

All databases are divided into train (60 percentage of the whole number of articles) and test set (40 percentage). To apply our algorithms we extracted all words which appear in the training set regardless of the letter case and we also excluded dots, commas and other punctual signs. We have not used any additional filtering as excluding the stop or ignore words

3 TEXT PREPROCESSING METHODS

3.1 Binary Preprocessing

The simplest approach is to take each word of the document as a binary coordinate and the size of the feature space will be the size of our vocabulary.

3.2 TF-IDF

TF-IDF is a well-known approach for text preprocessing based on multiplication of term frequency tf_{ij} (ratio between the number of times i^{th} word occurs in j^{th} document and the document size) and inverse document frequency idf_i.

$$tf_{ij} = \frac{t_{ij}}{T_j}, \tag{1}$$

where t_{ij} is the number of times the i^{th} word occurs in the j^{th} document. T_j is the document size (number of the words in the j^{th} document).

There are different ways to calculate the weight of each word. In this paper we run classification algorithms with the following variants.

1) TF-IDF 1

$$idf_i = \log \frac{D}{n_i}, \tag{2}$$

where D is the number of documents in the training set and n_i is the number of documents which have the i^{th} word.

2) TF-IDF 2

Formula is given by equation (2) except n_i is calculated as the number of times i^{th} word appears in all documents from the training set.

3) TF-IDF 3

$$idf_i = \left(\frac{D}{n_i}\right)^{\alpha}, \alpha \in (0;1), \tag{3}$$

where n_i is calculated as in TF-IDF 1 and α is the parameter (in this paper we have tested α = 0.1, 0.5, 0.9).

3) TF-IDF 4

Formula is given by equation (3) except n_i is calculated as in TF-IDF 4.

3.3 ConfWeight

Maximum Strength (Maxstr) is an alternative method to find the word weights. This approach has been proposed in (Soucy and Mineau, 2005)). It implicitly does feature selection since all frequent words have zero weights. The main idea of the method is that the feature f has a non-zero weight in class c only if the f frequency in documents of the c class is greater than the f frequency in all other classes. The ConfWeight method uses Maxstr as an analogy of IDF:

$$CW_{ij} = \log(tf_{ij} + 1) \cdot Maxstr(i). \tag{4}$$

Numerical experiments (Soucy and Mineau, 2005) have shown that the ConfWeight method is more effective than TF-IDF with SVM and k-NN as classification methods. The main drawback of the ConfWeight method is computational complexity. This method is more computationally demanding than TF-IDF method because the ConfWeight method requires time-consuming statistical calculations such as Student distribution calculation and confidence interval definition for each word.

3.4 Novel Term Weighting (TW)

The main idea of the method (Gasanova et al., 2013) is similar to ConfWeight but it is not so time-consuming. The idea is that every word that appears in the article has to contribute some value to the certain class and the class with the biggest value we define as a winner for this article.

For each term we assign a real number term relevance that depends on the frequency in utterances. Term weight is calculated using a modified formula of fuzzy rules relevance estimation for fuzzy classifiers (Ishibuchi et al., 1999). Membership function has been replaced by word frequency in the current class. The details of the procedure are the following:

Let L be the number of classes; n_i is the number of articles which belong to the i^{th} class; N_{ij} is the number of j^{th} word occurrence in all articles from the i^{th} class; $T_{ji} = N_{ji}/n_i$ is the relative frequency of there j^{th} word occurrence in the i^{th} class.

$R_j = max_i(T_{ji})$, $S_j = arg(max_i(T_{ji}))$is the number of class which we assign to the j^{th} word;

The term relevance, C_j, is given by

$$C_j = \frac{1}{\sum_{i=1}^{L} T_{ji}} \cdot \left(R_j - \frac{1}{L-1} \cdot \sum_{i=1, i \neq S_j}^{L} T_{ji} \right). \tag{5}$$

C_j is higher if the word occurs more often in one class than if it appears in many classes. We use novel TW as an analogy of IDF for text preprocessing.

The learning phase consists of counting the C values for each term; it means that this algorithm uses the statistical information obtained from the training set.

4 CLASSIFICATION ALGORITHMS

We have considered 11 different text preprocessing methods (8 modifications of TF-IDF, binary representation, ConfWeight and novel TW method) and compared them using different classification algorithms. The methods have been implemented using RapidMiner (Shafait et al., 2010). The classification

methods are:

-k-nearest neighbours algorithm with weights (we have varied k from 1 to 15);
-kernel Bayes classifier with Laplace correction;
-neural network with error back propagation (standard setting in RapidMiner);
-Rocchio classifier with different metrics and parameter;
-linear support vector machine (SVM) (standard setting in RapidMiner).

5 DIMENSION REDUCTION WITH COOPERATIVE COEVOLUTIONARY ALGORITHM

5.1 Term Clustering

For each word extracted from the train data we assigned the weight and the number of class where it contributes. It is possible to use clustering of words in the dictionary for dimension reduction. Therefore, we suggest to preprocess our dictionary such that words of equal or similar weights are placed in the same cluster and one common weight will be assigned to all words in this cluster. It should be mentioned that our preprocessing stage does not use any specific linguistic information, expert knowledge or domain related information. Therefore it can be easily transportable to the data from another domain or even in another language.

In order to reduce the dictionary size we take hierarchical agglomerative clustering(Ward, 1963). As a common weight of the cluster we calculate the arithmetic mean of all word weights from this cluster. To choose which clusters are joint on the current step we calculate all distances between clusters:

$$dist\,(X,Y) = \frac{1}{N} \cdot \frac{1}{M} \sum_i \sum_j \left\| X_i - Y_j \right\|, \qquad (6)$$

where N is the number of words in cluster X and M is the number of words in cluster Y; and we unite the closest clusters.

It is important to notice that we can not apply clustering for binary preprocessing.

5.2 Genetic Algorithm for Cluster Weights Optimization

After we clustered words in the dictionary there is a hierarchical tree for each category and assigned values to all clusters. The question if these values are a global optimum remains open. There is no evidence that the current values are even a local maximum of classification quality function.

To optimize weights of clusters when there is a predefined set of clusters for the certain category we suggest to apply genetic algorithm hybridized with local search due to its relative simplicity and global convergence, and it does not require any a priori information about behaviour of the classification quality function.

In this work we apply a local search algorithm only to the best obtained individual to make sure that it reaches at least a local maximum. The weights of other categories are fixed and only the weights of the current class are being optimized. Each individual represents weight for the current category encoded to a binary string. As a fitness function we use the F-score on train set calculated with the fixed weights and weights of the individual after classification.

Application of standard classification methods is time-consuming in optimization process because the classification algorithm must learn for each individual at each generation. In this case it is proposed to use of a simple decision rule which can be calculated quickly. Relative frequency of each word from the dictionary in each class is calculated before optimization. After that the class with the maximal relative frequency of the word is chosen. Therefore, every word in the dictionary is obtained to one class. During optimization process classification efficiency with classification of each utterance from train sample is calculated. For each utterance we calculate a sum of weights of words in the utterance which are obtained to each class. After that the class with the best sum is chosen as a winner.

$$A_i = \sum_{j:S_j=i} W_j; winner = \arg\max_i \{A_i\}, \qquad (7)$$

where i is the number of a class, j is the number of a word in the utterance, W_j is the weight of the j^{th} word, S_j is the number of the class which corresponds to the j^{th} word.

5.3 Cooperative Coevolutionary Genetic Algorithm

In order to take advantage of all weights improvement we propose to apply a cooperative coevolutionary genetic algorithm with local search. The main stages of applied method are shown in Figure 1.

On the first phase all individual genetic algorithms work separately (for each of them other weights are fixed and the task is to optimize weights which belong to the corresponding class), the length of this

phase defines how many generations individual algorithms can work without information exchange. Then we stop all separate algorithms and update all weights which have been obtained by the best individuals of all algorithms. We repeat these two stages until we reach the maximum number of generations.

This variant of coevolutionary algorithm uses a cooperative scheme in order to achieve higher performance than each individual algorithm, in this case subpopulations do not exchange individuals, only information that influences the fitness function calculation.

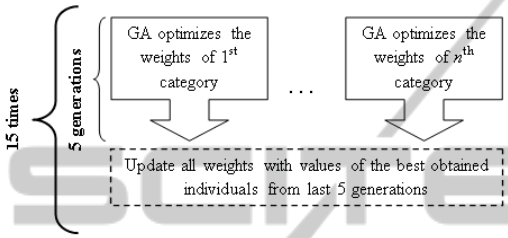

Figure 1: Coevolutionary genetic algorithm for cluster weights optimization.

6 RESULTS OF NUMERICAL EXPERIMENTS

The DEFT (Dfi Fouille de Texte) Evaluation Package 2008 and publicly available corpora from DEFT07 (Books, Games and Debates) have been used for algorithms application and results comparison. We used precision, recall, and F-score as the measure of classification quality.

Precision for each class i is calculated as the number of correctly classified articles for class i divided by the number of all articles which algorithm assigned for this class. Recall is the number of correctly classified articles for class i divided by the number of articles that should have been in this class. Overall precision and recall are calculated as the arithmetic mean of the precisions and recalls for all classes (macro-average). F-score is calculated as the harmonic mean of precision and recall.

The tables 3-7 present the F-scores obtained on the test corpora before clustering and cluster weights optimization. The best values for each problem are shown in bold. Results of all classification algorithms are presented with the best parameters. We also present for each corpus only the best TF-IDF modification. We use decision rule (7) also as a classification method.

We have performed hierarchical agglomerative clustering for each corpora with 50 and 100 clusters. Parameters of cooperative coevolutionary genetic algorithm for cluster weights optimization are presented in the Table 8. Mutation probability we calculate with formula:

$$Probability = \frac{1}{BitsNumber * ClustersNumber}. \quad (8)$$

As a illustrating of the efficiency of optimization we present classification efficiency (F-score) with decision rule (7) before and after optimization in the Table 9. Tables 10-14 presents the F-scores obtained on the test corpora after clustering and cluster weights optimization. There are numbers of clusters in the brackets.

Table 3: Classification results for *Books* before dimension reduction.

Classifier	Binary	TF-IDF	Conf Weight	Novel TW
Bayes	0.489	0.506	0.238	0.437
k-NN	0.488	0.517	0.559	0.488
Rocchio	0.479	0.498	0.557	0.537
SVM	0.509	0.500	0.534	0.486
Neural network	0.475	0.505	**0.570**	0.493
Decision rule (7)	0.238	0.405	0.372	0.505

Table 4: Classification results for *Games* before dimension reduction.

Classifier	Binary	TF-IDF	Conf Weight	Novel TW
Bayes	0.653	0.652	0.210	0.675
k-NN	0.703	0.701	**0.720**	0.700
Rocchio	0.659	0.678	0.717	0.712
SVM	0.668	0.685	0.210	0.675
Neural network	0.701	0.679	0.717	0.691
Decision rule (7)	0.210	0.580	0.641	0.658

Table 5: Classification results for *Debates* before dimension reduction.

Classifier	Binary	TF-IDF	Conf Weight	Novel TW
Bayes	0.555	0.645	0.363	0.616
k-NN	0.645	0.648	0.695	0.695
Rocchio	0.636	0.646	0.697	0.696
SVM	0.655	0.642	0.634	0.702
Neural network	0.656	0.647	**0.705**	0.697
Decision rule (7)	0.363	0.586	0.695	0.694

Table 6: Classification results for *T1* before dimension reduction.

Classifier	Binary	TF-IDF	Conf Weight	Novel TW
Bayes	0.501	0.690	0.837	0.794
k-NN	0.800	0.816	0.855	0.837
Rocchio	0.794	0.825	0.853	0.838
SVM	0.775	0.812	0.848	0.836
Neural network	0.783	0.830	0.853	**0.854**
Decision rule (7)	0.728	0.807	0.832	0.838

Table 7: Classification results for *T2* before dimension reduction.

Classifier	Binary	TF-IDF	Conf Weight	Novel TW
Bayes	0.569	0.728	0.712	0.746
k-NN	0.728	0.786	0.785	0.811
Rocchio	0.765	0.825	0.803	0.834
SVM	0.794	0.837	0.813	**0.851**
Neural network	0.799	0.838	0.820	0.843
Decision rule (7)	0.388	0.780	0.771	0.803

Table 8: Settings of the coevolutionary genetic algorithm.

Parameter	Value
Subpopulation size	100
Number of generations when subpopulations work separately	5
Number of iterations	15
Number of bits to code each variable	17
Variables lie in the interval	[0;1]
Selection	Tournament (size = 3)
Crossover	Uniform

Table 9: Classification results with decision rule (7) before and after optimization.

Problem	TF-IDF	Conf Weight	Novel TW
Books	0.238(100) 0.238(50)	0.219(100) 0.219(50)	0.506(100) 0.506(50)
Books (optim.)	0.537(100) 0.519(50)	0.566(100) 0.560(50)	0.539(100) 0.517(50)
Games	0.373(100) 0.373(50)	0.373(100) 0.373(50)	0.373(100) 0.373(50)
Games (optim.)	0.722(100) 0.706(50)	0.695(100) 0.699(50)	0.717(100) 0.716(50)
Debats	0.363(100) 0.363(50)	0.363(100) 0.363(50)	0.363(100) 0.363(50)
Debats (optim.)	0.682(100) 0.676(50)	0.701(100) 0.704(50)	0.695(100) 0.690(50)
T1	0.652(100) 0.652(50)	0.652(100) 0.652(50)	0.652(100) 0.652(50)
T1 (optim.)	0.812(100) 0.800(50)	0.830(100) 0.815(50)	0.830(100) 0.833(50)
T2	0.668(100) 0.668(50)	0.668(100) 0.668(50)	0.668(100) 0.668(50)
T2 (optim.)	0.835(100) 0.835(50)	0.831(100) 0.832(50)	0.837(100) 0.838(50)

Table 10: Classification results for *Books* after dimension reduction.

Classifier	TF-IDF	Conf Weight	Novel TW
Bayes	0.515(100) 0.489(50)	0.432(100) 0.450(50)	0.499(100) 0.504(50)
k-NN	0.528(100) 0.513(50)	0.455(100) 0.445(50)	0.526(100) 0.538(50)
Rocchio	0.545(100) 0.516(50)	0.455(100) 0.479(50)	0.519(100) 0.536(50)
SVM	0.536(100) 0.521(50)	0.444(100) 0.429(50)	0.519(100) 0.505(50)
Neural network	0.534(100) 0.503(50)	0.455(100) 0.435(50)	0.512(100) 0.515(50)
Decision rule (7)	0.537(100) 0.519(50)	**0.566**(100) 0.560(50)	0.539(100) 0.517(50)

We can see from the Tables 3-7 that the best F-scores have been obtained with either ConfWeight or novel TW preprocessing and with different classification algorithms (neural network, k-NN, or SVM). After term clustering and clusters weight optimization the best F-scores have been obtained with TF-IDF, ConfWeight, or novel TW preprocessing and also with different classification methods (k-NN, Rocchio classifier, or decision rule (7)).

From the Table 9 we can see efficiency of clusters weight optimization. After clustering and before optimization classification efficiency is very low. Therefore, optimization of cluster weights with cooperative coevolutionary genetic algorithm is necessary for effective dimension reduction.

After clustering with optimization the best values of F-measure increase for *Games* (0.720 - 0.731)and *Debates* (0.705 - 0.712) and decrease non-significantly for *Books* (0.570 - 0.566), *T1* (0.854 - 0.841), and *T2* (0.851 - 0.843). It is important to no-

Table 11: Classification results for *Games* after dimension reduction.

Classifier	TF-IDF	Conf Weight	Novel TW
Bayes	0.670(100) 0.666(50)	0.557(100) 0.541(50)	0.682(100) 0.691(50)
k-NN	**0.731**(100) 0.722(50)	0.587(100) 0.587(50)	0.723(100) 0.717(50)
Rocchio	0.726(100) 0.708(50)	0.590(100) 0.569(50)	0.716(100) 0.722(50)
SVM	0.699(100) 0.702(50)	0.557(100) 0.563(50)	0.710(100) 0.708(50)
Neural network	0.721(100) 0.712(50)	0.570(100) 0.588(50)	0.708(100) 0.713(50)
Decision rule (7)	0.722(100) 0.706(50)	0.695(100) 0.699(50)	0.717(100) 0.716(50)

Table 12: Classification results for *Debates* after dimension reduction.

Classifier	TF-IDF	Conf Weight	Novel TW
Bayes	0.600(100) 0.606(50)	0.558(100) 0.606(50)	0.593(100) 0.611(50)
k-NN	0.679(100) 0.678(50)	0.688(100) 0.703(50)	0.692(100) 0.690(50)
Rocchio	0.682(100) 0.668(50)	0.698(100) **0.712**(50)	0.693(100) 0.688(50)
SVM	0.683(100) 0.679(50)	0.697(100) 0.705(50)	0.695(100) 0.694(50)
Neural network	0.684(100) 0.675(50)	0.699(100) 0.709(50)	0.695(100) 0.691(50)
Decision rule (7)	0.682(100) 0.676(50)	0.701(100) 0.704(50)	0.695(100) 0.690(50)

Table 13: Classification results for *T1* after dimension reduction.

Classifier	TF-IDF	Conf Weight	Novel TW
Bayes	0.661(100) 0.669(50)	0.576(100) 0.283(50)	0.692(100) 0.714(50)
k-NN	0.816(100) 0.801(50)	0.710(100) 0.338(50)	0.825(100) 0.830(50)
Rocchio	0.820(100) 0.807(50)	0.723(100) 0.319(50)	0.837(100) **0.841**(50)
SVM	0.808(100) 0.797(50)	0.667(100) 0.250(50)	0.789(100) 0.823(50)
Neural network	0.821(100) 0.819(50)	0.691(100) 0.246(50)	0.774(100) 0.835(50)
Decision rule (7)	0.812(100) 0.800(50)	0.830(100) 0.815(50)	0.830(100) 0.833(50)

Table 14: Classification results for *T2* after dimension reduction.

Classifier	TF-IDF	Conf Weight	Novel TW
Bayes	0.687(100) 0.691(50)	0.609(100) 0.638(50)	0.687(100) 0.696(50)
k-NN	0.839(100) 0.840(50)	0.819(100) 0.819(50)	**0.843**(100) 0.842(50)
Rocchio	0.835(100) 0.834(50)	0.806(100) 0.809(50)	0.837(100) 0.836(50)
SVM	0.840(100) 0.837(50)	0.815(100) 0.817(50)	0.842(100) 0.841(50)
Neural network	0.834(100) 0.832(50)	0.813(100) 0.815(50)	0.836(100) 0.802(50)
Decision rule (7)	0.835(100) 0.835(50)	0.831(100) 0.832(50)	0.837(100) 0.838(50)

tice that we have reduced dimension of the problems very significantly: from the size of the vocabulary (see the Tables 1-2) to 100 or 50.

7 CONCLUSIONS

This paper reported on text classification experiments on 5 different corpora using several classification methods with different text preprocessing. We have used TF-IDF modifications, ConfWeight and novel term weighting approach as preprocessing techniques. K-Nearest neighbours algorithms, Bayes classifier, Support Vector Machine, Rocchio classifier, and Neural Network have been applied as classification algorithms.

After that we have performed dimension reduction based on hierarchical agglomerative clustering of terms and cluster weight optimization using cooperative coevolutionary genetic algorithm. This method reduces dimension of text classification problem significantly. Classification efficiency may increase or decrease non-significantly after clustering with optimization of cluster weights.

REFERENCES

Ando, R. K. and Zhang, T. (2005). A high-performance semi-supervised learning method for text chunking. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 1–9. Association for Computational Linguistics.

Bengio, Y., Schwenk, H., Senécal, J.-S., Morin, F., and Gauvain, J.-L. (2006). Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

DEFT07 (2007). *Proceedings of the 3rd DEFT Workshop*. AFIA, Grenoble, France.

DEFT08 (2008). *Proceedings of the 4th DEFT Workshop*. TALN, Avignon, France.

Gasanova, T., Sergienko, R., Semenkin, E., Minker, W., and Zhukov, E. (2013). A semi-supervised approach for natural language call routing. *Proceedings of the SIGDIAL 2013 Conference*, pages 344–348.

Huang, F. and Yates, A. (2009). Distributional representations for handling sparsity in supervised sequence-labeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1*, pages 495–503. Association for Computational Linguistics.

Ishibuchi, H., Nakashima, T., and Murata, T. (1999). Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 29(5):601–618.

Koo, T., Carreras, X., and Collins, M. (2008). Simple semi-supervised dependency parsing. *ACL*, pages 595–603.

Miller, S., Guinness, J., and Zamanian, A. (2004). Name tagging with word clusters and discriminative training. In *HLT-NAACL*, volume 4, pages 337–342.

Mnih, A. and Hinton, G. (2007). Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, pages 641–648. ACM.

Potter, M. and Jong, K. D. (2000). Cooperative coevolution: an architecture for evolving coadapted subcomponents. *Trans. Evolutionary Computation*, 8:129.

Ratinov, L. and Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155. Association for Computational Linguistics.

Rocchio, J. (1971). Relevance feedback in information retrieval, in the smart retrival system-experiments in automatic document processing. *Prentice-Hall*, pages 313–323.

Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, pages 513–523.

Schwenk, H. and Gauvain, J.-L. (2002). Connectionist language modeling for large vocabulary continuous speech recognition. In *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, volume 1, pages I–765. IEEE.

Shafait, F., Reif, M., Kofler, C., and Breuel, T. M. (2010). Pattern recognition engineering. In *RapidMiner Community Meeting and Conference*, volume 9.

Soucy, P. and Mineau, G. (2005). Beyond tf-idf weighting for text categorization in the vector space model. *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005)*, pages 1130–1135.

Suzuki, J. and Isozaki, H. (2008). Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *ACL*, pages 665–673. Citeseer.

Suzuki, J., Isozaki, H., Carreras, X., and Collins, M. (2009). An empirical study of semi-supervised structured conditional models for dependency parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 551–560. Association for Computational Linguistics.

Turian, J., Ratinov, L., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.

Ward, J. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244.