Text Classification Based on Transfer Learning and Self-Training

Yabin Zheng, Shaohua Teng, Zhiyuan Liu, Maosong Sun
State Key Laboratory on Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
{Yabin.zheng, tengshaohua, liuliudong}@gmail.com, sms@mail.tsinghua.edu.cn

Abstract

Traditional text classification methods make a basic assumption: the training and test set are homologous, while this naïve assumption may not hold in the real world, especially in the web environment. Documents on the web change from time to time, pre-trained model may be out of date when applied to new emerging documents. However some information of training set is nonetheless useful. In this paper we proposed a novel method to discover the constant common knowledge in both training and test set by transfer learning, then a model is built based on this knowledge to fit the distribution in test set. The model is reinforced iteratively by adding most confident instances in unlabeled test set to training set until convergence, which is a self-training process, preliminary experiment shows that our method achieves an approximately 8.92% improvement as compared to the standard supervised-learning method.

1. Introduction

With the development of the Internet, information on the web accumulates exponentially, which is usually organized as textual documents. This makes text mining a hot research field. An important topic in text mining is to associate it with the pre-defined semantically meaningful category, which is known as Text Classification [1]. Traditional text classification methods always follow a supervised learning strategy: use training set and machine learning technologies, like SVM [2], KNN [3] to build a model, and then classify the unlabeled instances in the test set.

However, all those traditional strategies make a default and basic assumption: the training and test set are homologous. When this naïve assumption does not hold, the performance may get worse. For example, model trained to do web-page classification task can be easily out of date, as the pages on the web change frequently. New terms emerge; old terms disappear; identical terms have different meanings. Pre-defined categories may be also changed, new categories arise with new terms introduced; large category is divided into more specific field; some categories may become more and more similar, which makes reasonable to merge them. Meanwhile, term distributions between categories may be also different as time goes by. We refer all those factors to temporal aspects, which is studied in [4]. Results show that accuracy can be gained by taking temporal aspects into consideration. In this paper, we only take the term distribution into account.

Nowadays, transfer learning [5, 6, 7] has drawn a large body of research in the machine learning community. Transfer learning is the application of skills and knowledge learned in one context being applied in another context. A related topic is multi-task learning [8], which learns a problem together with other related problems at the same time, using a common knowledge. This common knowledge exists in almost all the problems and is crucial for new problem solving. We restrict to the problem when training and test set are heterogeneous. However, they may contain constant common knowledge. We focus on how to discover this common knowledge using transfer learning technologies.

Meanwhile, we may encounter trouble when the common knowledge is not sufficient. A feasible way named semi-supervised learning [9] is to utilize both the labeled and unlabeled data for training, which can exploit the merit of both, such as co-training, transductive support vector machine, self-training. Self-training [9, 10] starts by training on the labeled data only, then it chooses most confident unlabeled instances to enlarge the labeled training set. This process continues until convergence or maximum iterations reached. We use self-training in this paper because of its efficiency.

In this paper, we utilize the methods mentioned above. First, transfer knowledge learned from the labeled training set to unlabeled test set, make some of the unlabeled instances labeled. We call those instances common knowledge transmitters (CKT). Second, we use the CKT set instead of the original training set because of homology. Thirdly, we implement the standard self-training process using the labeled CKT set as training set, then choose most confident unlabeled instances to retrain the model until convergence. Preliminary experiment shows that our method is effective.

The rest of the paper is organized as follows. In Section 2 and 3, we present our algorithm and some analysis in detail. Experiment results and discussions are shows in Section 4. Section 5 concludes the whole paper and gives future works.

2. Common Knowledge Transmitter Detection

In this section, we will discuss how to detect the common knowledge transmitters using transfer learning technology. In fact, transfer learning also happens in the daily life, it is much easier for people who know Italian to learn Spanish, or play chess having already learned checkers. The reason is that all those tasks have some shared common knowledge. Other tasks will be simpler to solve if this knowledge is detected.

In text classification task, if training and test set are heterogeneous, traditional methods always have a reduction in performance. However, there might be constant common knowledge in both. Pre-defined meaningful categories are identified by a set of terms. Term distribution may be different, but there must be some invariable terms in training and test set. The more invariable terms exist, the better performance we will get. Experiment results in section 4 demonstrate this.

In this paper, we use common knowledge transmitters to transfer knowledge from training set to test set. Model is built using training set, and then applied to test set. Instances in test set are ordered according to their labeling confidence. Instances fall in the top λ% part are chosen as common knowledge transmitters. It is reasonable to believe that these transmitters' labels are more reliable than the rest. Larger λ value means more instance picked, but more noise introduced, making the accuracy decrease; smaller λ value means more accurate instances picked, but may be not sufficient for model training. We do some experiment for threshold parameter tuning in section 4. The pseudo-code of common knowledge transmitter detection is shown in Table 1.

Table 1. Common Knowledge Transmitter Detection

Input: labeled training set L, unlabeled test set U, confidence threshold λ
Output: labeled common knowledge transmitter set LT

```
1). Build a svm model using original training set L,
label instances in the test set U
2). Order the instances according to confidence
for each instance i in U
        if i lies in top λ%
                put i in labeled transmitter set LT
Remove LT from U
        U ← U – LT
3). Return
```

Since we get common knowledge transmitters, we abandon the original training set, because they are heterogeneous from test set. In fact, we have done contrast trial: build models using only transmitters and combining both transmitters and original training set, performance of two models is comparable. This shows that transmitters already transferred enough knowledge from training set to the test set. There is no need to use the original training set; moreover, training time can be reduced.

3. Self-Training

In the last section, we proposed the idea of detecting common knowledge transmitters from test set. But we may still lack of sufficient labeled data, as the transmitters account for a small portion of the whole test set. We utilize both the labeled transmitters and unlabeled data for training by taking advantage of semi-supervised learning theory.

Let LT and U denote the Labeled Transmitter set and Unlabeled set drawn from the same distribution, respectively. We built preliminary model using LT as training set, then, self-training technology is applied to reinforce the model. In standard self-training process, the model keeps on choosing the most confident instances, say L′, from the unlabeled set, and reinforces the model on LT∪L′ until convergence or maximum iterations reached. Self-training requires neither estimation nor maximization of some probability like EM algorithm. Besides, it does not require the datasets include conditional independent attributes, which is necessary for Co-Training, so it is much simpler to use and more efficient.

We use SVM to do self-training in this paper, which is proved to be outstanding in many machine learning tasks. The convergence of self-training SVM algorithm is addressed in [10]. The only difference is that they use the whole unlabeled instances to do self-training; we only use the most confident instances to enlarge the training set.

First, a standard two-class SVM classifier can be described as:

$$\min \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i, \quad subject\ to \qquad (1)$$

$$y_i(w^T x_i + b) \geq 1-\xi_i, i = 1,2,...N, \xi_i \geq 0$$

Where x_i is a training instance, y_i is the label of x_i, C is a constant. N is the size of labeled CKT set.

We define an objection function, which is used as terminal condition in each iteration.

$$f(w^{(k)},\xi^{(k)}) = \frac{1}{2}\left\|w^{(k)}\right\|^2 + C\sum_{i=1}^{N+\sum_{j=1}^{k}N_j}\xi_i^{(k)} \qquad (2)$$

Where $w^{(k)}$ and $\xi^{(k)}$ are parameters in the kth iteration, N_j is the size of L′ in the jth iteration. It can be proved $f(w^{(k)},\xi^{(k)})$ is a monotonically nonincreasing function and $f(w^{(k)},\xi^{(k)})\geq 0$, so the algorithm is convergent.

The pseudo-code of self-training process is in Table 2.

Table 2. Self-training for TC

Input: labeled set LT, unlabeled set U, confidence threshold λ, maximum iteration M, constant δ
Output: label predictions for U
Repeat for M iterations:
1). Build a svm model using LT, label instances in U and order them by labeling confidence
2). Put top λ% instances in confident set L′
Add L′ to LT, remove L′ from U at the same time
LT ←L′∪LT, U ← U - L′
if $|f(w^{(k)},\xi^{(k)})- f(w^{(k-1)},\xi^{(k-1)})|< \delta$
break
End of repeat

4. Experiments

As mentioned before, we focus on the problem that the training and test set are heterogeneous. We use two data collections in Chinese language.

1. The electronic version of Chinese Encyclopedia (CE): it has 55 semantically meaningful categories and 7140 single-labeled documents. We use this collection as original training set in this paper.
2. The Chinese Web Documents Collection (CWD): it has the same taxonomy as CE, including 24016 single-labeled plain-text documents. CWD reflects the characteristic of Web documents. The distributions of the two collections are different though under the same taxonomy. We use this collection as test set.

Libsvm [11] is used as our basic classifier as it has been proved to be effective on many machine learning tasks especially text classification. We use probability svm models in this paper. Previous work [12] shows that Chinese character bigram has better performance than Chinese word unit. Besides, we don't need to take Chinese word segmentation into consideration. CHI-square is adopted in feature selection, with a dimension cutoff value of 60,000. Finally, Micro-average F1-Measure is calculated as performance evaluation.

4.1. Heterogeneous Problem

First we show the heterogeneous problem. Given the collections of CE and CWD, we can use either of them to train a svm model, test on both. The results are shown in Table 3.

Table 3. Heterogeneous Distribution

Training Set	Test Set	F1-Measure
CE	CE	87.61%
CE	CWD	65.75%
CWD	CE	54.45%
CWD	CWD	94.82%

It is obvious to find that heterogeneous distributions between training and test set make the performance degrade dramatically. In this paper, we use CE as training set, CWD as test set. So, our baseline result is 65.75%.

4.2. Term Distribution Perspective

Many factors can cause the heterogeneous problem. We focus on the term distribution of training and test set. Terms may emerge or disappear, or become more or less discriminative for corresponding category, and the identical terms may have different meanings in two sets. For example, the term "Album" always appears in category literature in the CE set, while the category changes to music in the CWD set.

In order to make qualitative and quantitative analysis on this problem, we create a vocabulary $V_{k,l}$ containing the top *t* terms with the highest CHI-square in category C_k in training set. $V_{k,u}$ is also created in test set. Those terms can be considered as the most informative units to describe the category. Next we compute the similarity of term distribution of category k in training and test set.

$$Sim(k) = \frac{|V_{k,l} \bigcap V_{k,u}|}{\left|V_{k,l} \bigcup V_{k,u}\right|} \qquad (3)$$

It is obvious that $0 \le sim(k) \le 1$, if the vocabulary in category k keeps constant in two sets, sim(k) reaches the upper limit. On the other hand, if vocabulary in category k is completely different, sim(k) reaches the lower limit. It is reasonable to make a conclusion that the larger sim(k) is, the higher performance will be gained in category k.

We perform the experiment to testify this. Parameter *t* is set by 1000. Table 4 and 5 shows the first and last 5 categories according to their similarity calculated by formula (3).

Table 4. Term Distribution of Top 5 Categories

Category	Term Distribution Similarity
Philosophy	21.21%
Language	19.05%
Military	18.91%
Archaeology	18.27%
Jurisprudence	17.72%

Table 5. Term Distribution of Last 5 Categories

Category	Term Distribution Similarity
Electronic and Computer Engineering	2.77%
Foreign Literature	1.83%
Oceanography, Meteorology and Hydrology	1.32%
Geography	1.16%
Geophysics, Topography and Geospace Science	0.60%

As can be seen, different categories have different term distribution similarities. The more constant or static the category is, the more similar the term distributions are. For example, distributions of *Philosophy* have a similarity of 0.21 while in class *Electronic and Computer Engineering* it is only 0.00277. This indicates that *Electronic and Computer Engineering* is more dynamic.

We also examine the relationship between term distribution similarity and the Macro-average F1-Measure of the corresponding class, generally speaking, sim(k) has positive relation to the Macro-F1 of category k. The more terms they share, the better performance obtained. For instance, class *Language* has a similarity of 0.19, the corresponding Macro-F1 is 0.76. While similarity of class *Geophysics, Topography and Geospace Science* is only 0.0006, making the Macro-F1 only reaches 0.24.

4.3. Confidence Threshold Tuning

In the common knowledge transmitters detection step, we have to decide how many instances in test set picked as transmitters, which is controlled by a confidence threshold parameter λ. λ=0 indicates that none is chosen, while λ=100 means that all the instances are chosen. Small λ makes the chosen instances more accurate, but those instances may be insufficient for self-training; large λ will choose more instances but with more noise introduced. A trade-off between accuracy and noise should be taken into consideration. We perform the following experiment: the parameter λ is varied from 0 to 100 stepped by 5, and record the corresponding the accuracy in the CKT set as well as F1-Measure in test set. Results are shown in Fig 1 and Fig 2.As illustrated in Fig. 1, the noise accumulates when λ increases. Accuracy of CKT set is 88.14% when λ=5, while it drop down to 67% when λ=95. In Fig.2, F1-Measure first increases as more transmitters added, which means that more common knowledge is transferred than the noise. Then F1 reaches the peak point of 73.26% when λ=35. After that, performance decreases as too much noise introduced. Furthermore, if λ=100, all the instances are chosen as transmitters, which is the result of baseline. We use λ=35 in this paper.

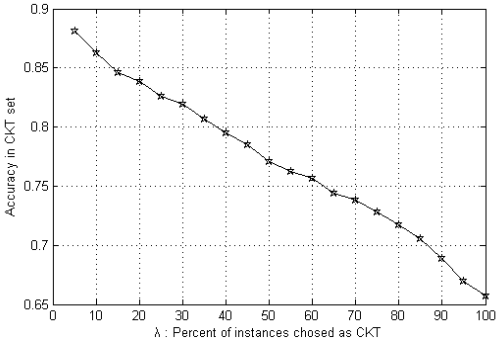

Fig 1. Relationship between accuracy and λ in CKT set

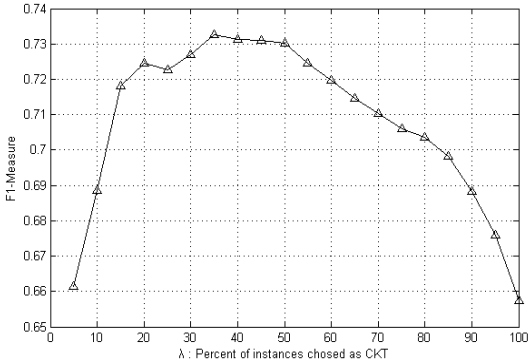

Fig 2. Relationship between F1-Measure and λ

4.4. Instances Dropping

After CKT detection step is done, we believe that those transmitters contain enough information transferred from training set to test set. To give evidence of this, we do two contrast trials, one model built with original training set and the labeled transmitters while the other built only with the labeled transmitters. Results show that the two models are comparable, which means that we can abandon the original training set to enhance the efficiency of the algorithm. In fact, the performance is better if we drop original training set when λ=15, and the gap is only 0.29% when λ=35.

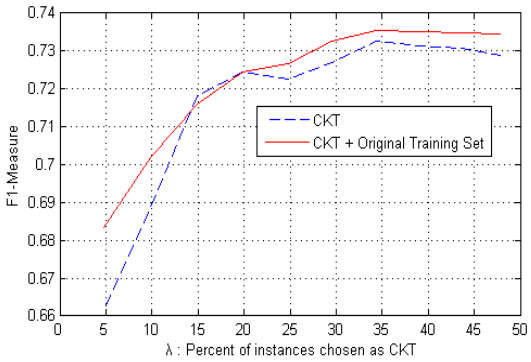

Fig 3. Contrast Trials for Instance Dropping

4.5. Convergence of Self-Training

The convergence of self-training SVM is proved in [10]. We do some experiments to support this. Results show that the algorithm converges under various values of λ. Generally speaking, F1-Measure ascends in first three iterations of self-training process, and then converges very fast. When λ = 35, we get the best result in the third iteration, which is 74.67% in F1-Measure. In other words, we gain an approximately 8.92% improvement as compared to the baseline.

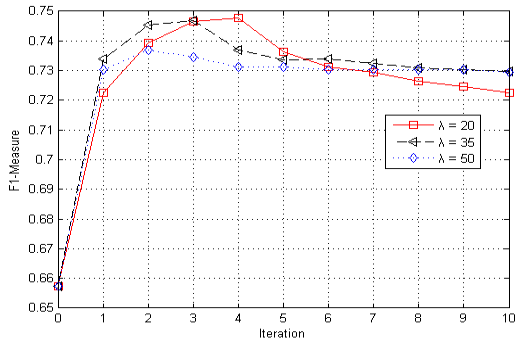

Fig 4. Convergence of Self-Training

5. Conclusion and Future Work

In this paper, we proposed a novel method to solve the problem that the training and test set are heterogeneous. We make a basic assumption that the two sets share some common knowledge indeed rather than homologous assumption. Then, transfer learning is applied to detect the common knowledge transmitter. Experiment shows that those transmitters contain enough shared knowledge of the two sets. Self-training enhances the performance gradually, by choosing most confident unlabeled instances to enlarge CKT set. This process continues until reaches convergence. Experiments indicate that performance is ascending during the first three iterations. Our method gains an approximately 8.92% improvement as compared to the standard supervised-learning method. We also do an experiment which using CWD as original training set and CE as test set. Our method gets a result of 61.05% while the baseline is only 54.45%.

We note that although we have proved the convergence of the algorithm both theoretically and experimentally, the method is still sensitive to the supervised machine learning technology we used. SVM is a discriminative model which is not so sensitive to self-training. In the future, we will extend the algorithm to some generative model such as Naïve Bayesian model. Finally, we only do experiments on CE dataset, whether this method is effective on other datasets like 20Newsgroups is left as future work.

6. Acknowledgement

The research is supported by the National Natural Science Foundation of China under grant number 60573187.

7. Reference

[1] F. Sebastiani, Machine Learning in Automated Text Categorization, *ACM Computing Surveys*, 34(1): pp. 1-47, 2002.

[2] T. Joachims, Text categorization with Support Vector Machines: Learning with many relevant features, *In Proceedings of the 10th European Conference on Machine Learning*, pp. 137-142, 1998.

[3] Y.M. Yang and X. Liu, A re-examination of text categorization methods, *In Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 42–49, 1999.

[4] F. Mourão, L. Rocha, et al., Understanding Temporal Aspects in Document Classification, *In Proceedings of the International Conference on Web Search and Web Data Mining*, pp.159-169, 2007.

[5] W.Y. Dai, Q. Yang, G.R. Xue and Y. Yu, Boosting for Transfer Learning, *In Proceedings of the 24th Annual International Conference on Machine Learning*, pp. 193-200, 2007.

[6] W.Y. Dai, G.R. Xue, Q. Yang and Y. Yu, Transferring Naive Bayes Classifiers for Text Classification, *In Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pp. 540-545, 2007.

[7] C.Do, A. Ng, Transfer Learning for text classification, *In Proceedings of Neural Information Processing Systems*, 2005.

[8] R. Caruana, Multitask Learning, *Machine Learning*, 28(1): pp. 41-75, 1997.

[9] O. Chapelle, A. Zien, and B. Scholköpf, *Semi-supervised learning*, MIT Press, Cambridge, MA,2006.

[10] Y.Q. Li, H.Q. Li, C.T. Guan and Z.Y. Chin, A Self-Training Semi-Supervised Support Vector Machine Algorithm and its Applications in Brain Computer Interface, *In International Conference on Acoustics, Speech and Signal Processing*, pp. 385-388, 2007.

[11] C.C Chang and C.J. Lin, LIBSVM: a library for support vector machines, Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001.

[12] J.Y. Li et al., A Comparison and Semi-Quantitative Analysis of Words and Character-Bigrams as Features in Chinese Text Categorization, *In Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL*, pp. 545–552, 2006.