Learning Timeline Difference for Text Categorization

Fumiyo Fukumoto
Graduate Faculty of
Interdisciplinary Research
Univ. of Yamanashi, Japan
fukumoto@yamanashi.ac.jp

Yoshimi Suzuki
Graduate Faculty of
Interdisciplinary Research
Univ. of Yamanashi, Japan
ysuzuki@yamanashi.ac.jp

Abstract

This paper addresses text categorization problem that training data may derive from a different time period from the test data. We present a learning framework which extends a boosting technique to learn accurate model for timeline adaptation. The results showed that the method was comparable to the current state-of-the-art biased-SVM method, especially the method is effective when the creation time period of the test data differs greatly from the training data.

1 Introduction

Text categorization supports and improves several tasks such as creating digital libraries, information retrieval, and even helping users to interact with search engines (Mourao et al., 2008). A growing number of machine learning (ML) techniques have been applied to the text categorization task (Xue et al., 2008; Gopal and Yang, 2010). Each document is represented using a vector of features/terms (Yang and Pedersen, 1997; Hassan et al., 2007). Then, the documents with category label are used to train classifiers. Once category models are trained, each test document is classified by using these models. A basic assumption in the categorization task is that the distributions of terms between training and test documents are identical. When the assumption does not hold, the classification accuracy is worse. However, it is often the case that the term distribution in the training data is different from that of the test data when the training data may drive from a different time period from the test data. Manual annotation of tagged new data is very expensive and time-consuming. The methodology for accurate classification of the new test data by making the maximum use of tagged old data is needed in learning techniques.

In this paper, we present a method for text categorization that minimizes the impact of temporal effects. Our approach extends a boosting technique to learn accurate model for timeline adaptation. We used two types of labeled training data: One is the same creation time period with the test data. Another is different creation time period with the test data. We call the former *same-period* training, and the latter *diff-period* training data. For the same-period training data, the learner shows the same behavior as the boosting. In contrast, for diff-period training data, once they are wrongly predicted by the learned model, these data would be useless to classify test data. We decreased the weights of these data by applying Gaussian function in order to weaken their impacts.

2 Related Work

The analysis of temporal aspects is a practical problem as well as the process of large-scale heterogeneous data since the World-Wide Web (WWW) is widely used by various sorts of people. It is widely studied in many text processing tasks. One attempt is concept or topic drift dealing with temporal effects (Klinkenberg and Joachims, 2000; Kleinberg, 2002; Lazarescu et al., 2004; Folino et al., 2007; Song et al., 2014). Wang *et al.* developed the continuous time dynamic topic model (cDTM) (Wang et al., 2008). He *et al.* proposed a method to find bursts, periods of elevated occurrence of events as a dynamic phenomenon instead of focusing on arrival rates (He and Parker, 2010). They used Moving Average Convergence/Divergence (MACD) histogram which was used in technical stock market analysis (Murphy, 1999) to detect bursts.

Another attempt is domain adaptation. The goal of this attempt is to develop learning algorithms that can be easily ported from one domain to another (Daumé III, 2007; Sparinnapakorn and Ku-

Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 799–804,
Lisbon, Portugal, 17-21 September 2015. ©2015 Association for Computational Linguistics.

bat, 2007; Glorot et al., 2011; Siao and Guo, 2013). Domain adaptation is particularly interesting in Natural Language Processing (NLP) because it is often the case that we have a collection of labeled data in one domain but truly desire a model that can work well for another domain. Lots of studies addressed domain adaptation in NLP tasks such as part-of-speech tagging (Siao and Guo, 2013), named-entity (Daumé III, 2007), and sentiment classification (Glorot et al., 2011) are presented. One approach to domain adaptation is to use transfer learning. The transfer learning is a learning technique that retains and applies the knowledge learned in one or more tasks to efficiently develop an effective hypothesis for a new task. The earliest discussion is done by ML community in a NIPS-95 workshop[1], and more recently, transfer learning techniques have been successfully applied in many applications. Blitzer *et al.* proposed a method for sentiment classification using structural correspondence learning that makes use of the unlabeled data from the target domain to extract some relevant features that may reduce the difference between the domains (Blitzer et al., 2006). Several authors have attempted to learn classifiers across domains using transfer learning in the text classification task (Raina et al., 2006; Dai et al., 2007; Sparinnapakorn and Kubat, 2007). Raina *et al.* proposed a transfer learning algorithm that constructs an informative Bayesian prior for a given text classification task (Raina et al., 2006). They reported that a 20 to 40% test error reduction over a commonly used prior in the binary text classification task. Dai *et al.* presented a method called TrAdaBoost which extends boosting-based learning algorithms (Dai et al., 2007). Their experimental results show that TrAdaBoost allows knowledge to be effectively transferred from the old data to the new one. All of these approaches aimed at utilizing a small amount of newly labeled data to leverage the old data to construct a high-quality classification model for the new data. However, the temporal effects are not explicitly incorporated into their models.

To our knowledge, there have been only a few previous work on temporal-based text categorization. Mourao *et al.* investigated the impact of temporal evolution of document collections on the document classification (Mourao et al., 2008). Salles *et al.* presented an approach to classify documents in scenarios where the method uses information about both the past and the future, and this information may change over time (Salles et al., 2010). They address the drawbacks of which instances to select by approximating the Temporal Weighting Function (TWF) using a mixture of two Gaussians. However, their method needs tagged training data across full temporal range of training documents to construct TWF.

There are three novel aspects in our method. Firstly, we propose a method for text categorization that minimizes the impact of temporal effects in a learning technique. Secondly, from manual annotation of data perspective, the method allows users to annotate only a limited number of newly training data. Finally, from the perspective of robustness, the method is automated, and can be applied easily to a new domain, or different languages, given sufficient old labeled documents.

3 Learning Timeline Difference

Our learning model, Timeline Adaptation by Boosting (TABoost) is based on AdaBoost (Freund and Schapire, 1997). AdaBoost aims to boost the accuracy of a weak learner by adjusting the weights of training instances and learn a classifier accordingly. The TABoost uses two types of training data, same-period and diff-period training data. The assumption is that the quantity of the same-period data is limited, while diff-period training data is abundant. The TABoost aims at utilizing the diff-period training data to make up the deficit of a small amount of the same-period to construct a high-quality classification model for the test data. Similar to the TrAdaBoost presented by (Dai et al., 2007), TABoost is the same behavior as boosting for the same-period training data. In contrast, once diff-period training instances are wrongly predicted, we assume that these instances do not contribute to the accurate test data classification, and the weights of these instances decrease in order to weaken their impacts. The difference between TrAdaBoost and TABoost is a weighting manner, *i.e.* TABoost is a continuous timeline model, and it weights these instances by applying Gaussian function in order to weaken their impacts. TABoost is illustrated in Figure 1.

The training data set Tr is partitioned into two labeled sets Tr_{dp}, and Tr_{sp}. Tr_{dp} in Figure 1

[1] http://socrates.acadiau.ca/courses/comp/dsilver/NIPS95_LTL/transfer.workshop.1995.html.

```
Input {
    The diff-period data $Tr_{dp}$, the same-period data $Tr_{sp}$, and the maximum number of iterations $N$.
}

Output {
    $h_f(x)$ = $\sum_{t=1}^{N} \alpha_t h_t(x_i)$.
}

Initialization {
    $\mathbf{w^1}$ = $(w_1^1, \cdots, w_{n+m}^1)$.
}

TABoost {
For $t$ = 1,$\cdots$,$N$
    1. Set $\mathbf{P^t}$ = $\mathbf{w^t}$/ ($\sum_{k=1}^{n+m} w_i^t$).
    2. Train a weak learner on the combined training set $Tr_{dp}$ and $Tr_{sp}$ with the distribution $\mathbf{P^t}$, and create weak hypothesis $h_t$: $X \rightarrow \{-1, +1\}$
    3. Calculate the error of $h_t$ on the combined training set $Tr_{dp}$ and $Tr_{sp}$:
           $\epsilon_t = \sum_{i=1}^{n+m} \frac{w_i^t \cdot |h_t(x_i) - c(x_i)|}{\sum_{i=1}^{n+m} w_i^t}$.
    4. Chose $\alpha_t = \frac{1}{2}\text{In}(\frac{1-\epsilon_t}{\epsilon_t})$
    5. Update the new weight vector:

       $w_i^{t+1} = \begin{cases} w_i^t \text{exp}^{F(\delta)\alpha_t |h_t(x_i) - c(x_i)|}, \\ \quad 1 \leq i \leq n \\ w_i^t \text{exp}^{-\alpha_t |h_t(x_i) - c(x_i)|}, \\ \quad n+1 \leq i \leq n+m \end{cases}$

}
```

Figure 1: Flow of the algorithm

shows the diff-period training data that Tr_{dp} = $\{(x_i^{dp}, c(x_i^{dp}))\}$, where $x_i^{dp} \in X_{dp}$ (i = 1, $\cdots$, n), and X_{dp} refers to the diff-period instance space. Similarly, Tr_{sp} represents the same-period training data that Tr_{sp} = $\{(x_i^{sp}, c(x_i^{sp}))\}$, where $x_i^{sp} \in X_{sp}$ (i = 1, $\cdots$, m), and X_{sp} refers to the same-period instance space. n and m are the number of documents in Tr_{dp} and Tr_{sp}, respectively. $c(x_i)$ returns a label for the input instance x_i. The combined training set Tr = $\{(x_i, c(x_i))\}$ is given by:

$$x_i = \begin{cases} x_i^{dp} & i = 1, \cdots, n \\ x_i^{sp} & i = n+1, \cdots, n+m \end{cases}$$

In each iteration round shown in Figure 1, if a diff-period training instance is wrongly predicted, the instance may be useless to classify test data correctly. We decrease its training weight to reduce the effect. To do this, we assume a standard lognormal distribution (Crow, 1988), *i.e.* $F(\delta)$ = $\frac{1}{\sqrt{2\pi}}\text{exp}(-\frac{\delta^2}{2})$. δ in $F(\delta)$ represents time difference between diff-period training and test data. For instance, if the training data is 1999, and test data is 2000, δ equals to 1. Similarly, if the training data is 2000, and test data is 1999, δ is -1. The greater the time difference value, the smaller the training weight. We fit the model according to the temporal range of the data. As shown in Figure 1, we decrease its training weight w_i^t to reduce its effect through multiplying its weight by $\text{exp}^{F(\delta)\alpha_t |h_t(x_i) - c(x_i)|}$.

We used the Support Vector Machines (SVM) as a learner. We represented each training and test document as a vector, each dimension of a vector is a noun word appeared in the document, and each element of the dimension is a term frequency. We applied the algorithm shown in Figure 1. After several iterations, a learner model is created by linearly combining weak learners, and a test document is classified by using a learner.

4 Experiments

We evaluated our TABoost by using the Mainichi Japanese newspaper documents.

4.1 Experimental setup

We choose the Mainichi Japanese newspaper corpus from 1991 to 2012. The corpus consists of 2,883,623 documents organized into 16 categories. We selected 8 categories, "International(Int)", "Economy(Eco)", "Home", "Culture", "Reading", "Arts", "Sports", and "Local news(Local)", each of which has sufficient number of documents. All documents were tagged by using a morphological analyzer Chasen (Matsumoto et al., 2000) and selected noun words. The total number of documents assigned to these categories are 787,518. For each category within each year, we divided documents into three folds: 2% of documents are used as the same-period training data, 50% of documents are the diff-period training data, and the remains are used to test our classification method.[2]

[2] When the creation time period of the training data is the same as the test data, we used only the same-period training data.

Table 1: The error rates across categories

Cat	TAB_s	SVM	TrAdaB	b-SVM	TAB
Int	0.409	0.467	0.326	0.253	**0.329**
Eco	0.368	0.429	0.243	0.228	**0.208**
Home	0.475	0.649	0.312	0.460	**0.172**
Culture	0.468	0.848	0.440	0.559	**0.196**
Reading	0.358	0.520	0.298	0.357	**0.337**
Arts	0.402	0.684	0.330	0.588	**0.331**
Sports	0.226	0.212	0.107	0.075	**0.123**
Local	0.586	0.305	0.400	0.156	**0.303**
M-Avg	0.411	0.514	0.307	0.334	**0.257**

We used LIBLINEAR (Fan et al., 2008) as a basic learner in the experiments. We compared our method, TABoost with four baselines: (1) TABoost with the same-period training data only (TAB_s), (2) SVM, (3) TrAdaBoost (Dai et al., 2007), and (4) biased-SVM (Liu et al., 2003) by SVM-light (Joachims, 1998). TAB_s is the same behavior as boosting. TrAdaBoost (TrAdaB) is presented by (Dai et al., 2007). Biased-SVM (b-SVM) is known as the state-of-the-art SVMs method, and often used for comparison (Elkan and Noto, 2008). Similar to SVM, for biased-SVM, we used the first two folds as a training data, and classified test documents directly, *i.e.* we used closed data. We empirically selected values of two parameters, "c" (trade-off between training error and margin) and "j", *i.e.* cost (cost-factor, by which training errors on positive instances) that optimized result obtained by classification of test documents. Similar to (Liu et al., 2003), "c" is searched in steps of 0.02 from 0.01 to 0.61. "j" is searched in steps of 5 from 1 to 200. As a result, we set c and j to 0.01 and 10, respectively. To make comparisons fair, all five methods including our method are based on linear kernel. Throughout the experiments, the number of iterations is set to 100. We used error rate as an evaluation measure (Dai et al., 2007).

4.2 Results

Categorization results for 8 categories (48% of the test documents, *i.e.* 378,008 documents) are shown in Table 1. Each value in Table 1 shows macro-averaged error rate across 22 years. "M-Avg" refers to macro-averaged error rate across categories. The results obtained by biased-SVM show minimum error rate obtained by varying the parameters, "c" and "j".

As can be seen clearly from Table 1, the overall performance obtained by TAB was the best among the five methods. The macro average error rates with TrAdaB and TAB were lower to those obtained by b-SVM, although b-SVM in Table 1 was the result obtained by using the closed data. In contrast, SVM did not work well. This demonstrates that once the training data drive from a different time period from the test data, the distributions of terms between training and test documents are not identical. The results obtained by TAB_s were worse than those obtained by TrAdaBoost, b-SVM, and TAB. This shows that (i) the same-period training data we used is not sufficient to train a model alone, and (ii) TAB demonstrates a good transfer ability.

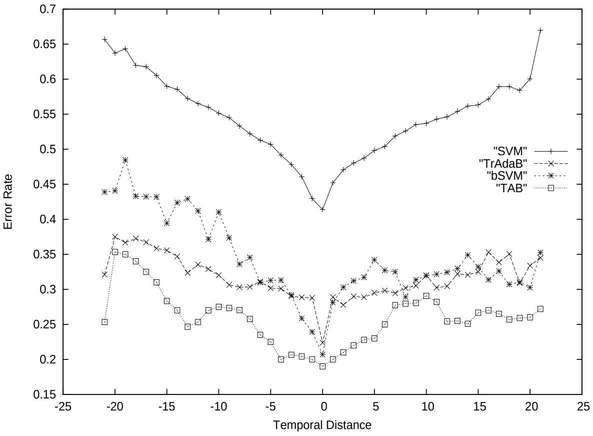

Figure 2: Performance against temporal distance

Figure 2 illustrates error rate against the temporal difference between diff-period training and test data. Both training and test data are the documents from 1991 to 2012. For instance, "10" of the x-axis in Figure 2 indicates that the test documents are created 10 years later than the training documents. We can see from Figure 2 that the result obtained by TAB was the best in all of the temporal distances. There are no significant differences among three methods, bSVM, TrAdaB, and TAB when the test and training data are the same time period. The performance of these methods including SVM drops when the creation time of the test data differs greatly from the diff-period training data. However, the performance of TAB was still better to those obtained by other methods. This demonstrates that the algorithm with continuous timeline model works well for categorization.

Figure 3 shows the error rate against the number of iterations. Each curve shows averaged error rate under time period between same-period and diff-

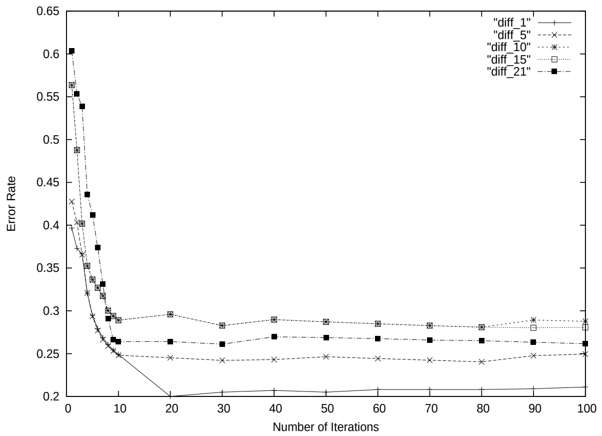

Figure 3: Iteration curves by temporal distance

period training data. For example, "diff_10" indicates that the difference time period between same and diff training data is ± 10 years. We can see from Figure 3 that all curves except for "diff_20" drop rapidly and converge around 10 iterations. "diff_20" converges around 20 iterations. This was the same behaviour as TrAdaBoost, *i.e.* TrAdaBoost converges around 20 iterations. The fast convergence is not particularly surprising because we used a small number of same-period (2%) and a large number of diff-period (50%) training data. It is necessary to examine how the ratio between same-period and diff-period training data affects overall performance for further quantitative evaluation, although a main contribution of TAB is in situations using by both of a small amount of labeled new data which is not sufficient to train a model alone, and a large amount of old data.

5 Conclusion

We have presented a method for text categorizaiton that minimizes the impact of temporal effects. The results using Japanese Mainichi Newspaper corpus show that it works well for categorization, especially when the creation time of the test data differs greatly from the training data. There are a number of interesting directions for future work. The rate of convergence of TAB ($O(\sqrt{\text{In}n/N})$) is slow which can also be found in (Dai et al., 2007). Here, n is the number of training data, and N is the number of iterations. In the future, we will try to extend the framework to address this issue. We used Japanese newspaper documents in the experiments. For quantitative evaluation, we need to apply our mehtod to other data such as ACM-DL and a large, heterogeneous collection of web content in addition to the experiment to examine the performance agasint the ratio between same-period and diff-period training data.

References

J. Blitzer, R. McDonald, and F. Pereira. 2006. Domain Adaptation with Structural Correspondence Learning. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 120–128.

S. K. Crow. 1988. Log-normal Distributions: Theory and Application. *NewYork: Dekker*.

W. Dai, Q. Yang, G.R. Xue, and Y. Yu. 2007. Boosting for Transfer Learning. In *Proc. of the 24th International Conference on Machine Learning*, pages 193–200.

H. Daumé III. 2007. Frustratingly Easy Domain Adaptation. In *Proc. of the 45th Annual Meeting of the Association of computational Linguistics*, pages 256–263.

C. Elkan and K. Noto. 2008. Learning Classifiers from Only Positive and Unlabeled Data. In *Proc. of the KDD'08*, pages 213–220.

F. E. Fan, K. W. Chang, C. J. Hsieh, X. R. Wang, and C. J. Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. *Machine Learning*, 9:1871–1874.

G. Folino, C. Pizzuti, and G. Spezzano. 2007. An Adaptive Distributed Ensemble Approach to Mine Concept-drifting Data Streams. In *Proc. of the 19th IEEE International Conference on Tools with Artificial Intelligence*, pages 183–188.

Y. Freund and R. E. Schapire. 1997. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1):119–139.

X. Glorot, A. Bordes, and Y. Bengio. 2011. Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach. In *Proc. of the 28th International Conference on Machine Learning*, pages 97–110.

S. Gopal and Y. Yang. 2010. Multilabel Classification with Meta-level Features. In *Proc. of the 33rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 315–322.

S. Hassan, R. Mihalcea, and C. Nanea. 2007. Random-Walk Term Weighting for Improved Text Classification. In *Proc. of the IEEE International Conference on Semantic Computing*, pages 242–249.

D. He and D. S. Parker. 2010. Topic Dynamics: An Alternative Model of Bursts in Streams of Topics. In *Proc. of the 16th ACM SIGKDD Conference on Knowledge discovery and Data Mining*, pages 443–452.

T. Joachims. 1998. SVM Light Support Vector Machine. In *Dept. of Computer Science Cornell University*.

M. Kleinberg. 2002. Bursty and Hierarchical Structure in Streams. In *Proc. of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 91–101.

R. Klinkenberg and T. Joachims. 2000. Detecting Concept Drift with Support Vector Machines. In *Proc. of the 17th International Conference on Machine Learning*, pages 487–494.

M. M. Lazarescu, S. Venkatesh, and H. H. Bui. 2004. Using Multiple Windows to Track Concept Drift. *Intelligent Data Analysis*, 8(1):29–59.

B. Liu, Y. dai, X. Li, W. S. Lee, and P. S. Yu. 2003. Building Text Classifiers using Positive and Unlabeled Examples. In *Proc. of the ICDM'03*, pages 179–188.

Y. Matsumoto, A. Kitauchi, T. Yamashita, Y. Hirano, Y. Matsuda, K. Takaoka, and M. Asahara. 2000. *Japanese Morphological Analysis System Chasen Version 2.2.1.* In Naist Technical Report.

F. Mourao, L. Rocha, R. Araujo, T. Couto, M. Goncalves, and W. M. Jr. 2008. Understanding Temporal Aspects in Document Classification. In *Proc. of the 1st ACM International Conference on Web Search and Data Mining*, pages 159–169.

J. Murphy. 1999. *Technical Analysis of the Financial Markets*. Prentice Hall.

R. Raina, A. Y. Ng, and D. Koller. 2006. Constructing Informative Priors using Transfer Learning. In *Proc. of the 23rd International Conference on Machine Learning*, pages 713–720.

T. Salles, L. Rocha, and G. L. Pappa. 2010. Temporally-aware Algorithms for Document Classification. In *Proc. of the 33rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 307–314.

M. Siao and Y. Guo. 2013. Domain Adaptation for Sequence Labeling Tasks with a Probabilistic Language Adaptation Model. In *Proc. of the 30th International Conference on Machine Learning*, pages 293–301.

M. Song, G. E. Heo, and S. Y. Kim. 2014. Analyzing topic evolution in bioinformatics: Investigation of dynamics of the field with conference data in dblp. *Scientometrics*, 101(1):397–428.

K. Sparinnapakorn and M. Kubat. 2007. Combining Subclassifiers in Text Categorization: A DST-based Solution and a Case Study. In *Proc. of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 210–219.

C. Wang, D. Blei, and D. Heckerman. 2008. Continuous Time Dynamic Topic Models. In *Proc. of the 24th Conference on Uncertainty in Artificial Intelligence*, pages 579–586.

G. R. Xue, W. Dai, Q. Yang, and Y. Yu. 2008. Topic-bridged PLSA for Cross-Domain Text Classification. In *Proc. of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 627–634.

Y. Yang and J. O. Pedersen. 1997. A Comparative Study on Feature Selection in Text Categorization. In *Proc. of the 14th International Conference on Machine Learning*, pages 412–420.