# Hybrid Method of Semi-supervised Learning and Feature Weighted Learning for Domain Adaptation of Document Classification

**Hiroyuki Shinnou, Liying Xiao, Minoru Sasaki, Kanako Komiya**
Ibaraki University, Department of Computer and Information Sciences
4-12-1 Nakanarusawa, Hitachi, Ibaraki JAPAN 316-8511
hiroyuki.shinnou.0828@vc.ibaraki.ac.jp,
{14nm721x, minoru.sasaki.01, kanako.komiya.nlp}
@vc.ibaraki.ac.jp

## Abstract

In regard to document classification, semi-supervised learning using the Naive Bayes method and EM algorithm was a great success, and we refer to this method as NBEM in this paper. Although NBEM is also effective for domain adaption of document classification, there is still room for improvement because NBEM does not employ valuable information for this task, that is the difference between source domain and target domain. Here, according to the similarity between the label distribution of the feature on source domain and the estimated label distribution of the feature on target domain, we set the weight on the features to reconstruct the training data. We use this reconstructed training data to perform document classification by NBEM. As a result of experiment by using a part of 20 Newsgroups, the effect of this method was confirmed.

## 1 Introduction

In this paper, for the domain adaption problems of document classification, we propose a hybrid method of semi-supervised learning and feature weighted learning. In many of the tasks of natural language processing, supervised learning has been a great success. However, if we want to use a supervised learning for real problems, there is often problems in domain adaptation. In general, the supervised learning is used to create a classifier which is usually using a learning algorithm such as support vector machine (SVM) by labeled training data, then it is possible to identify the label of the test data using this classifier. In this case, the problem is that the domain of training data and test data is different, so it is a problem of domain adaptation (Søgaard, 2013).

As a typical example, there is a sentiment analysis task to judge whether a review article for a commodity is positive or not (Blitzer et al., 2007). For example, if we use review articles for "book" as the training data to make a classifier, the classifier can not correctly identify the review articles for "movie" which is in another domain. In addition to the emotion analysis, supervised learning such as morphological analysis (Mori, 2012), parsing (Sagae and Tsujii, 2007), word sense disambiguation (Shinnou et al., 2015) (Komiya and Okumura, 2012) (Komiya and Okumura, 2011) is utilized in all tasks, it is possible that the domain adaptation problems come into being.

In general, the method of the domain adaptation can be divided into instance-based method and feature-based method (Pan and Yang, 2010). Instance-based method is a method of learning using weighted training data. Learning under covariate shift (Sugiyama and Kawanabe, 2011) is typical in this method. The covariate shift means the assumption that $P_S(\boldsymbol{x}) \neq P_T(\boldsymbol{x})$, $P_S(y|\boldsymbol{x}) = P_T(y|\boldsymbol{x})$. Learning under covariate shift is regarded as weighted learning, where the weight is set to the probability density ratio $P_T(\boldsymbol{x})/P_S(\boldsymbol{x})$. The feature-based method is a method that maps the source and target features spaces to a common features space to maintain important characteristics of both domains by reducing the difference between


*29th Pacific Asia Conference on Language, Information and Computation pages 500 - 507*
*Shanghai, China, October 30 - November 1, 2015*

domains. The paper (Blitzer et al., 2006) proposed the dimension reduction method called structural correspondence learning (SCL).

The paper (Daumé III, Hal, 2007) offered a weighting system for features . In this study, vector $\boldsymbol{x}_s$ of the training data in the source domain is mapped to an augmented input space $(\boldsymbol{x}_s, \boldsymbol{x}_s, \mathbf{0})$, and vector $\boldsymbol{x}_t$ of the training data in the target domain is mapped to an augmented input space $(\mathbf{0}, \boldsymbol{x}_t, \boldsymbol{x}_t)$. The classifier learned from the augmented vectors solves the classification problem. Daumé's method assumes that an effect can be determined by overlapping the characteristics that are common to the source and target domains.

Although these methods for domain adaption often work well, while the differences between the domains is small, there may be counterproductive by such an method. When the difference between the domains is small, it is realistic that the problem of domain adaption is simply regarded as data sparseness problem. In that case, the method of conventional semi-supervised learning (Chapelle et al., 2006) and active learning (Settles, 2010) (Rai et al., 2010) is better.

In this paper, we are dealing with problems of the domain adaption in document classification. Here, as described above, semi-supervised learning is available for dealing with domain adaption that difference between domains is small. Especially as semi-supervised learning of document classification, the method using the EM algorithm based on Naive Bayes method is very famous (Nigam et al., 2000). In this paper, we refer to this method as NBEM. Here, we also use the NBEM. However, there is still room for improvement because NBEM does not employ valuable information for this task, that is the difference between source domain and target domain. Here, we use the method shown by Chen (Chen et al., 2011) which has improved the learning of weighting feature. This method is named as Self-Training Feature Weight, called STFW for short. STFW uses self-learning to estimate the label distribution of features on target domain, but we use NBEM to do it in STFW. The original STFW can be applied to only a binary classification task. For the multi-class classification, we improve STFW. Finally, we use the combination of NBEM and STFW. The domain adaption of document classification can perform more accurately by this. As for the experiment we used the 20 Newsgroups data[1] to construct the domain A and the domain B, and then domain adaption experiments were conducted from domain A to domain B and from domain B to domain A. As a result, NBEM was effective for our task. And the proposed method was able to improve NBEM.

## 2 Related works

There are some researches using NBEM for domain adaptation of document classification. The Naive Bayes Transfer Classifier (NBTC) modifies EM parts in NBEM to adapt to a target domain (Dai et al., 2007). NBTC needs the probability that a test document appears in the source domain. NBTC estimates this probability by using KL divergence between the source domain and the target domain, and empirical parameters. The Adapting Naive Bayes (ANB) also modifies EM parts in NBEM like NBEM (Tan et al., 2009). AMB uses the mixture distribution of the source domain and the target domain as the document generative model. The weight of the source domain is reduced according to EM iterations. As a result, both of NBEM and ANB gives weight to a feature through the class distribution of target domain. On the other hand, our method is based on the idea that the feature must be weighted if the class distribution of a feature in the target domain are similar.

## 3 Hybrid method of NBEM and STFW

### 3.1 NBEM

NBEM is one of the semi-supervised learning for learning a classifier from a little labeled training data and much unlabeled data. Generally speaking, it is an method that learn the classifier of Naive Bayes from labeled training data, and use a large amount of unlabeled data and EM algorithm to improve this classifier.

In a classification problem, let $C = \{c_1, c_2, \cdots, c_m\}$ be a set of classes. An instance $x$ is represented as a feature list

$$\boldsymbol{x} = (f_1, f_2, \cdots, f_n). \tag{1}$$

We can solve the classification problem by estimating the probability $P(c|x)$. Actually, the class

[1] tt http://qwone.com/~jason/20Newsgroups/

$c_x$ of $\boldsymbol{x}$, is given by

$$c_x = \arg\max_{c \in C} P(c|\boldsymbol{x}). \quad (2)$$

Bayes theorem shows that

$$P(c|\boldsymbol{x}) = \frac{P(c)P(\boldsymbol{x}|c)}{P(\boldsymbol{x})}. \quad (3)$$

As a result, we get

$$c_x = \arg\max_{c \in C} P(c)P(\boldsymbol{x}|c). \quad (4)$$

In the above equation, $P(c)$ is estimated easily; the question is how to estimate $P(\boldsymbol{x}|c)$. Naive Bayes models assume the following:

$$P(\boldsymbol{x}|c) = \prod_{i=1}^{n} P(f_i|c). \quad (5)$$

The estimation of $P(f_i|c)$ is easy, so we can estimate $P(\boldsymbol{x}|c)$.

We can use the EM method if we use Naive Bayes for classification problems. In this paper, we show only key equations and the key algorithm of this method (Nigam et al., 2000).

Basically the method computes $P(f_i|c_j)$ where $f_i$ is a feature and $c_j$ is a class. This probability is given by[2]

$$P(f_i|c_j) = \frac{1 + \sum_{k=1}^{|D|} N(f_i, d_k)P(c_j|d_k)}{|F| + \sum_{m=1}^{|F|}\sum_{k=1}^{|D|} N(f_m, d_k)P(c_j|d_k)}. \quad (6)$$

$D$**:** all data consisting of labeled data and unlabeled data
$d_k$**:** an element in $D$
$F$**:** the set of all features
$f_m$**:** an element in $F$
$N(f_i, d_k)$**:** the number of $f_i$ in the instance $d_k$.

In our problem, $N(f_i, d_k)$ is 0 or 1, and almost all of them are 0. If $d_k$ is labeled, $P(c_j|d_k)$ is 0 or 1. If $d_k$ is unlabeled, $P(c_j|d_k)$ is initially 0, and is updated to an appropriate value step by step in proportion to the iteration of the EM algorithm.

[2]This equation is smoothed by taking into account the frequency 0.

By using equation 6, the following classifier is constructed:

$$P(c_j|d_i) = \frac{P(c_j)\prod_{f_n \in K_{d_i}} P(f_n|c_j)}{\sum_{r=1}^{|C|} P(c_r)\prod_{f_n \in K_{d_i}} P(f_n|c_r)}. \quad (7)$$

In this equation, $K_{d_i}$ is the set of features in the instance $d_i$.

$P(c_j)$ is computed by

$$P(c_j) = \frac{1 + \sum_{k=1}^{|D|} P(c_j|d_k)}{|C| + |D|}. \quad (8)$$

The EM algorithm computes $P(c_j|d_i)$ by using equation 7 (E-step). Next, by using equation 6, $P(f_i|c_j)$ is computed (M-step). By iterating E-step and M-step, $P(f_i|c_j)$ and $P(c_j|d_i)$ converge. In our experiment, when the difference between the current $P(f_i|c_j)$ and the updated $P(f_i|c_j)$ comes to less than $8 \cdot 10^{-6}$ or the iteration number reaches 10 times, we judge that the algorithm has converged.

### 3.2 STFW

In this paper, we improved STFW proposed by Chen. STFW is a feature-based method which is effective in domain adaption. In essence, feature-based method can be regarded as a method which maps the common space of feature between the space of target domain and the source domain. As for the operation, we corresponds to weighting the feature, so intuitively, it is also considered as an method that set a weight to feature that is effective to identification in both domains of the source domain and the target domain. Chen set weight to the feature in the following ways. First, we set the value of feature $f$ of data $\boldsymbol{x}$ to $x_f$, set the class of data $\boldsymbol{x}$ to $y_x$. We regard the correlation coefficient of $x_f$ and $y_x$ as $\rho_S(x_f, y_x)$ for labeled data in source domain. About the data $\boldsymbol{x}$ in target domain, its class is substituted for the class which estimated by self-learning $y'_x$, and we obtain the correlation coefficient $\rho_T(x_f, y'_x)$ of $x_f$ and $y'_x$. Then the weight $w(f)$ of feature $f$ is defined as the following.

$$w(f) = \frac{1 + \rho_S(x_f, y_x)\rho_T(x_f, y'_x)}{2} \quad (9)$$

A new value $v_{new}$ of the feature come to be obtained by multiplying the weight:

$$v_{new} = w(f) \cdot v_{old} \quad (10)$$

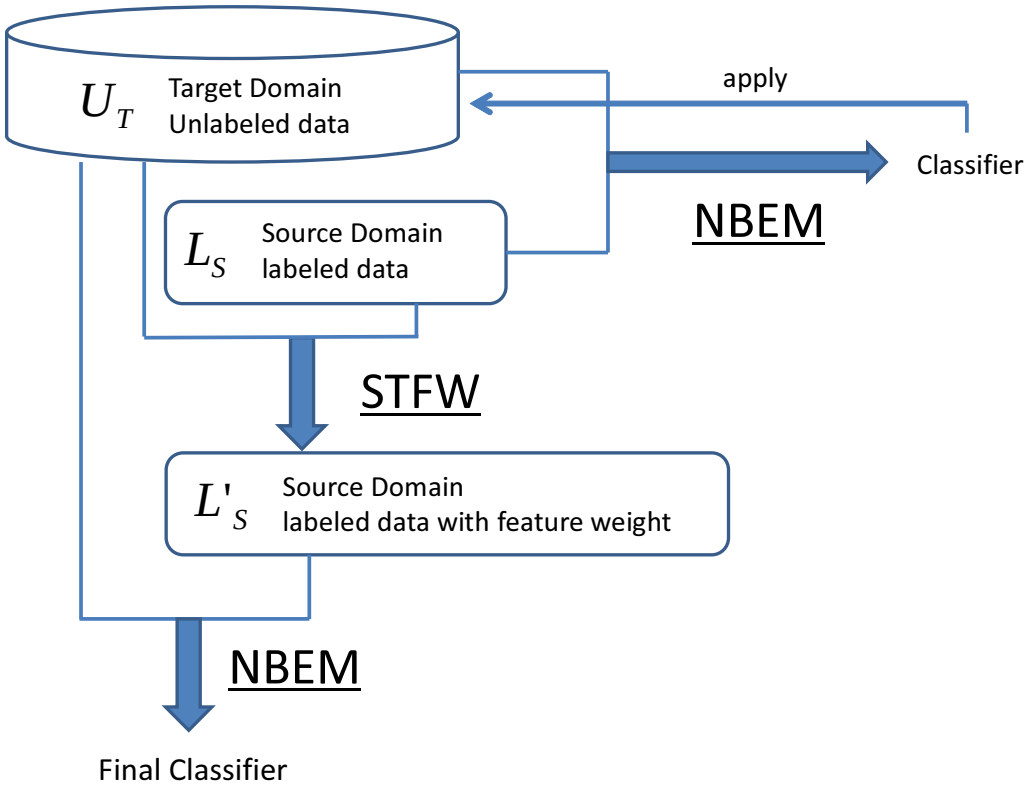


Figure 1: Hybrid method of NBEM and STFW

Note $v_{new} = 0$ if $v_{old} = 0$ in the equation 10.

Chen's method uses a correlation coefficient $\rho_S(x_f, y_x)$ と $\rho_T(x_f, y'_x)$ to define the weight. Because the label is a categorical value, in fact, only binary classification can be targeted. Based on Chen's method here, it is defined of weighting that it also can be used in the multi-class classification. The weight Chen defined can be regarded that measured the similarity of the label distribution $P_s$ of feature $f$ in source domain and label distribution $P_t$ of feature $f$ in target domain. The $P_s$ is the distribution of the following set:

$$\{y_x | x \ in \ Source \ data \ set, x_f > 0\}. \quad (11)$$

The $P_t$ can be defined by the same way.

Therefore, in this paper, first, define the distance $d(f)$ between $P_s$ and $P_t$ as following:

$$d(f) = |P_s - P_t|. \quad (12)$$

Then set the weight by using $d(f)$. However, our task is document classification. We use Naive Bayes as a learning algorithm, so the value of feature becomes frequency. Therefore, the value of feature (i.e. the weight) is desirably an integer of 0 or more. As a result, we define the new value $v_{new}$ of the feature as follows:

$$v_{new} = \begin{cases} v_{old} + 1 & if \quad d(f) < \theta_1, v_{old} > 0 \\ v_{old} - 1 & if \quad d(f) > \theta_2, v_{old} > 0 \\ v_{old} & if \quad others \end{cases}$$

However, if $v_{new}$ is a negative number after minus 1, $v_{new} = 0$. In the experiments of this paper, the parameter $\theta_1$ and $\theta_2$ was set to 0.2 and 1.5 respectively. These values were obtained through some experiments [3].

Also because there is no label of the data in target domain, $P_t$ can not simply obtained. Chen labeled the data in target domain by self-learning, and

[3]The parameter $\theta_1$ and $\theta_2$ depend on the number of classes. In the experiments of this paper, all of the number of classes are three.

seeking $P_t$ only on reliable data. In this paper, we do not use self-learning, but the classifier learned by NBEM. And it is not only limited to those reliable data, all of the data will be used to estimate $P_t$.

### 3.3 Combination of NBEM and STFW

In this paper we propose an method that uses a combination of NBEM and STFW, referring to Figure 1.

First, we learn a classifier by using the NBEM against labeled training data $L_S$ of the source domain and unlabeled data $U_T$ of the target domain. Use this classifier to estimate the label of $U_T$.

Using this label estimated, we set a weight to the feature of $L_s$ by STFW, and construct new training data $L'_S$.

## 4 Experiment

It took out a 20 Newsgroups data set [4] from the document group of following six categories in our experiment. Symbols in parentheses refer to the class name.

```
A: comp.sys.ibm.pc.hardware (comp)
B: rec.sport.baseball  (rec)
C: sci.electronics (sci)
D: comp.sys.mac.hardware  (comp)
E: rec.sport.hockey  (rec)
F: sci.med  (sci)
```

We suppose the dataset of `(A,B,C)` to domain X, and the dataset of `(D,E,F)` to domain Y. Each domain has become a dataset of the document classification that `L = {comp, rec, sci}` is the class label set.

The document number (the number of data) of each document group is shown in Table 2. Although the class distribution of labeled training data is uniform in each domain, Class distribution of the test data which can fit the problem of reality was set to be different in each domain.

On the one hand, in domain adaption which is from domain X to domain Y, labeled data of A, B, C becomes training data (a total of 300 documents), and the unlabeled data of D, E, F is unlabeled data(a total of 900 documents) which can be used. Then the test data of D,E,F is used as test data (a total of 600 documents). On the other hand, in domain adaption which is from domain Y to domain X, labeled data of D, E, F becomes training data (a total of 300 documents), and the unlabeled data of A, B, C is unlabeled data(a total of 900 documents) which can be used. Then the test data of A, B, C is used as test data (a total of 600 documents).

[4] http://qwone.com/~jason/20Newsgroups/

Table 2: Number of data of each document group

| | Labeled data | Unlabeled data | Test data |
|---|---|---|---|
| A | 100 | 400 | 300 |
| B | 100 | 300 | 200 |
| C | 100 | 200 | 100 |
| D | 100 | 200 | 100 |
| E | 100 | 400 | 300 |
| F | 100 | 300 | 200 |

The results of the experiment is shown in table1.

The column of NB (S-Only) learns the classifier only from the training data of the source domain by Naive Bayes, has been written of the accuracy rate of test data identified. The column of NBEM is the accuracy rate using the training data and unlabeled data by NBEM, the column of NBEM+STFW is accuracy rate by hybrid method of NBEM and STFW proposed in this paper. The effect of the method proposed in Table1 can be confirmed. Also as reference accuracy rate that it learn the classifier from training data of target domain by Naive Bayes is shown in NB (T-Only). These values have shown the accuracy rate of supervised learning in the case of the usual problems of domain adaption have not occurred.

## 5 Discussion

### 5.1 Comparison with transductive method

Like semi-supervised learning, transductive learning is another method using unlabeled data in order to improve the classifier learned through labeled data. And then as a representative method of transductive learning, there is Transductive-SVM (TSVM) (Joachims, 1999).

In this paper, although we use NBEM of semi-supervised learning, it is also possible to use the TSVM instead of NBEM.

Table 1: Experimental results (%)

| | NB (S-only) | NBEM | NBEM+STFW | NB (T-only) |
|---|---|---|---|---|
| X → Y | 72.83 | 90.00 | **92.33** | 94.67 |
| Y → X | 81.17 | 82.67 | **82.83** | 90.00 |

Table 3: Another method using unlabeled data

| | NB | NBEM | SVM | TSVM |
|---|---|---|---|---|
| X → Y | 72.83 | 90.00 | 75.83 | 66.50 |
| Y → X | 81.17 | 82.67 | 71.16 | 70.83 |

Table 4: Other domain adaptation methods

| | NBEM+STFW | SVM | SCL | uLSIF |
|---|---|---|---|---|
| X → Y | **92.33** | 75.83 | 74.33 | 73.67 |
| Y → X | **82.83** | 71.16 | 71.83 | 72.17 |

Generally SVM has a higher accuracy than NB. However, NB sometimes has high accuracy in the case of document classification. In fact, in domain adaption of Y → X, NB is better than SVM. When using NB for document classification, it is better that documents simply represent by a bag of words. Thus, using SVM, it becomes necessary to make some processing. In the experiment using SVM above, we set the vector value by TF*IDF, and finally normalize the size of the vector to 1.

TSVM does not improve the accuracy of the SVM, conversely the accuracy become lower. It is because that TSVM assumes that the class distribution of test data and training data is the same, but this assumption is not satisfied in our experiments.

### 5.2 Comparison with other methods of domain adaption

The method of domain adaption can be classified to feature-based method and instance-based method. In this section we apply a feature-based method and an instance-based method, and compare them with our proposed method.

As a feature-based method, we use the structural correspondence learning (SCL) (Blitzer et al., 2006). This is the representative feature-base method. On the other hand, the typical instance-based method is learning by covariate shift. In learning by covariate shift, the calculation of the probability density ratio become the key point. Here we use a density calculation method named Unconstrained Least Squares Importance Fitting (uLSIF) (Kanamori et al., 2009).

The result of experiment is shown in Table 4. NBEM+STFW in the table is the our proposed method.

As a result of SCL and uLSIF has not changed a lot that both of them is based of SVM, there is a high overwhelmingly accuracy toward NBEM+STFW. Here we can see the great difference of the results is because that whether the base of the learning algorithm is SVM or NB. NB made a higher accuracy than SVM just in our task. Both of SCL and uLSIF are transductive method, although the test data in target domain is used in the process of learning, the unlabeled data are not used. On the other hand, NBEM+STFW does not use test data, but unlabeled data. Test data is also unlabeled data, but the former is smaller than the latter. In this experiment, the amount of unlabeled data is 1.5 times of the amount of test data. Therefore it can be considered one reason that NBEM+STFW is better than SCL and uLSIF.

### 5.3 Weighting to feature

In this paper we give a weight to the feature likely to be valid for identification in domain adaption, subtract the weight of the feature likely to make an adverse effect on identification.

Here we examined the points following:

- Weighting to Test Data
- Size of the Added Weight
- Negative Weights

We show results of the experiment in turn below.

**Weighting to Test Data**

In this paper we set the weight to features of training data only, but it is also conceivable to the

test data. The result of the experiment is shown in Table 5.

Table 5: Weighting to Test Data (TW)

| | NBEM+STFW (without TW) - our method - | NBEM+STFW (with TW) |
|---|---|---|
| X → Y | **92.33** | 91.17 |
| Y → X | 82.83 | **83.00** |

Weighting to the test data is effective to domain adaption of Y → X, but it is not effective of X → Y.

**Size of the Added Weight**

In this paper, giving a weight means to plus 1, here we change it to plus 2, and the result of the experiment is shown in Table 6.

Table 6: Change the Size of the Added Weight

| | NBEM+STFW (+1) - our method - | NBEM+STFW (+2) |
|---|---|---|
| X → Y | 92.33 | **93.33** |
| Y → X | 82.83 | 82.83 |

While we make the twice of the weight, it is effective in domain adaption of X → Y, but it is not effective in Y → X.

**Negative Weights**

In domain adaption, there may be some labeled data which creates an adverse result in learning. This is called 'negative transfer' (Rosenstein et al., 2005). Our method is designed on the based on 'negative transfer.' That is, if the difference between class distributions of feature on the source domain and the target domain is quite big, we assign the feature negative weight ($-1$), In order to investigate the effect of negative weights here, we make an experiment which did not assign negative weight. And its result is shown in Table7.

Table 7: The Effect of Negative Weight (NW)

| | NBEM+STFW (with NW) - our method - | NBEM+STFW (without NW) |
|---|---|---|
| X → Y | 92.33 | **93.00** |
| Y → X | **82.83** | 82.67 |

Without negative weight, although it is effective in domain adaption of X → Y, it is not effective of Y → X.

It can be confirmed that the accuracy is subtly changed by the way of setting weight and its value.

## 6 Conclusion

In this paper, for the domain adaption problems of document classification, we proposed a hybrid method of semi-supervised learning and feature weighted learning. NBEM is used to learn a classifier, and then the learned classifier and SFTW reconstruct training data, and then the final classifier is learned by using the reconstruct training data and NBEM again. As a result of experiment by using a part of 20 Newsgroups, the effect of our method was confirmed. As for challenges in the future, we need to discover an more appropriate setting way and a better size of weight.

## References

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *EMNLP-2006*, pages 120–128.

John Blitzer, Mark Dredze, Fernando Pereira, et al. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, volume 7, pages 440–447.

Olivier Chapelle, Bernhard Schölkopf, Alexander Zien, et al. 2006. *Semi-supervised learning*, volume 2. MIT press Cambridge.

Minmin Chen, Kilian Q Weinberger, and John Blitzer. 2011. Co-training for domain adaptation. In *NIPS-2014*, pages 2456–2464.

Wenyuan Dai, Gui-Rong Xue, Qiang Yang, and Yong Yu. 2007. Transferring Naive Bayes Classifiers for Text Classification. In *AAAI-2007*.

Daumé III, Hal. 2007. Frustratingly Easy Domain Adaptation. In *ACL-2007*, pages 256–263.

Thorsten Joachims. 1999. Transductive inference for text classification using support vector machines. In *ICML*, volume 99, pages 200–209.

Takafumi Kanamori, Shohei Hido, and Masashi Sugiyama. 2009. A least-squares approach to direct importance estimation. *The Journal of Machine Learning Research*, 10:1391–1445.

Kanako Komiya and Manabu Okumura. 2011. Automatic Determination of a Domain Adaptation Method for Word Sense Disambiguation using Decision Tree Learning. In *IJCNLP-2011*, pages 1107–1115.

Kanako Komiya and Manabu Okumura. 2012. Automatic Domain Adaptation for Word Sense Disambiguation Based on Comparison of Multiple Classifiers. In *PACLIC-2012*, pages 75–85.

Shinsuke Mori. 2012. Domain adaptation in natural language processing (in japanese). *The Japanese Society for Artificial Intelligence*, 27(4):365–372.

Kamal Nigam, Andrew McCallum, Sebastian Thrun, and Tom Mitchell. 2000. Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning*, 39(2/3):103–134.

Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359.

Piyush Rai, Avishek Saha, Hal Daumé III, and Suresh Venkatasubramanian. 2010. Domain adaptation meets active learning. In *NAACL HLT 2010 Workshop on Active Learning for Natural Language Processing*, pages 27–32.

Michael T Rosenstein, Zvika Marx, Leslie Pack Kaelbling, and Thomas G Dietterich. 2005. To transfer or not to transfer. In *NIPS 2005 Workshop on Transfer Learning*, volume 898.

Kenji Sagae and Jun'ichi Tsujii. 2007. Dependency parsing and domain adaptation with lr models and parser ensembles. In *EMNLP-CoNLL-2007*, pages 1044–1050.

Burr Settles. 2010. Active learning literature survey. *University of Wisconsin, Madison*.

Hiroyuki Shinnou, Yoshiyuki Onodera, Minoru Sasaki, and Kanako Komiya. 2015. Active Learning to Remove Source Instances for Domain Adaptation for Word Sense Disambiguation. In *PACLING-2015*, pages 156–162.

Anders Søgaard. 2013. *Semi-Supervised Learning and Domain Adaptation in Natural Language Processing*. Morgan & Claypool.

Masashi Sugiyama and Motoaki Kawanabe. 2011. *Machine Learning in Non-Stationary Environments: Introduction to Covariate Shift Adaptation* . MIT Press.

Songbo Tan, Xueqi Cheng, Yuefen Wang, and Hongbo Xu. 2009. Adapting Naive Bayes to Domain Adaptation for Sentiment Analysis. In *Advances in Information Retrieval*, pages 337–349.