Social Media Text Classification under Negative Covariate Shift

Geli Fei and Bing Liu
University of Illinois at Chicago
Chicago, IL 60607, USA
gfei2@uic.edu, liub@cs.uic.edu

Abstract

In a typical social media content analysis task, the user is interested in analyzing posts of a particular topic. Identifying such posts is often formulated as a classification problem. However, this problem is challenging. One key issue is *covariate shift*. That is, the training data is not fully representative of the test data. We observed that the covariate shift mainly occurs in the negative data because topics discussed in social media are highly diverse and numerous, but the user-labeled negative training data may cover only a small number of topics. This paper proposes a novel technique to solve the problem. The key novelty of the technique is the transformation of document representation from the traditional *n-gram feature* space to a *center-based similarity* (CBS) space. In the CBS space, the covariate shift problem is significantly mitigated, which enables us to build much better classifiers. Experiment results show that the proposed approach markedly improves classification.

1 Introduction

Applications using social media data, such as reviews, discussion posts, and (micro) blogs are becoming increasingly popular. We observed from our collaborations with social science and health science researchers that in a typical application, the researcher first need to obtain a set of posts of a particular topic that he/she wants to study, e.g., a political issue. Keyword search is often used as the first step. However, that is not sufficient due to low precision and low recall. A post containing the keyword "politics" may not be a political post while a post that does not contain the keyword may be a political post. Thus, text classification is needed to make more sophisticated decisions to improve accuracy.

For classification, the user first manually labels a set of relevant posts (positive data) about the political issue and irrelevant posts (negative data) not about the political issue and then builds a classifier by running a learning algorithm, e.g. SVM or naïve Bayes. However, the resulting classifier may not be satisfactory. There may be many reasons. One key reason we observed is that the labeled negative training data is not fully representative of the negative test data.

Let the user-interested topic be P (positive), and the set of all other irrelevant topics discussed in a social media source be $T = \{T_1, T_2, \ldots, T_n\}$, which forms the negative data. n is usually large. However, due to the labor-intensive effort of manual labeling, the user can label only a certain number of training posts. Then the labeled negative training posts may cover only a small number of irrelevant topics S of T ($S \subseteq T$) as negative. Further, due to the highly dynamic nature of social media, it is probably impossible to label all possible negative topics. In testing, when posts of other negative topics in $T–S$ show up, their classification can be unpredictable. For example, in an application, the training data has no negative examples about *sports*. However, in testing, some sports posts show up. These unexpected sports posts may be classified arbitrarily, which results in low classification accuracy. In this paper, we aim to solve this problem.

In machine learning, this problem is called *covariate shift*, a type of *sample selection bias*. In classic machine learning, it is assumed that the training and testing data are drawn from the same distribution. However, this assumption may not hold in practice such as in our case above, i.e., the training and the test distributions are different (Heckman 1979; Shimodaira 2000; Zadrozny 2004; Huang et al. 2007; Sugiyama et al. 2008; Bickel et al. 2009). In general, the sample selection bias problem is not solvable because the two

Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 2347–2356,
Lisbon, Portugal, 17-21 September 2015.

distributions can be arbitrarily far apart from each other. Various assumptions were made to solve special cases of the problem. One main assumption was that the conditional distribution of the class given a data instance is the same in the training and test data sets (Shimodaira 2000; Huang et al. 2007; Bickel et al. 2009). This gives the *covariate shift* problem.

In this paper, we focus on a special case of the covariate shift problem. We assume that the covariate shift problem occurs mainly in the negative training and test data, and no or minimum covariate shift exists in the positive training and test data. This assumption is reasonable because the user knows the type of posts/documents that s/he is looking for and can label many of them.

Following the notations in (Bickel et al. 2009), our special case of the covariate shift problem can be stated formally as follows: let the set of training examples be $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_k, y_k)\}$, where $\mathbf{x}_i$ is the data/feature vector and y_i is the class label of $\mathbf{x}_i$. Let the set of test cases be $\{\mathbf{x}_{k+1}, \mathbf{x}_{k+2}, \ldots, \mathbf{x}_n\}$, which have no class labels. Since we are interested in binary classification, y_i is either 1 (positive class) or -1 (negative class). The labeled training data and the unseen test data have the same target conditional distribution $p(y|\mathbf{x})$ and the marginal distributions of the positive data in both the training and testing are also the same. But the marginal distributions of the negative data in the training and testing are different, i.e., $p_L(\mathbf{x}^-) \neq p_T(\mathbf{x}^-)$, where L, T, and $-$ represent the labeled training data, test data, and the negative class respectively.

Existing methods for addressing the covariate shift problem basically work as follows (see the Related Work section). First, they estimate the bias of the training data based on the given test data using some statistical techniques. Then, a classifier is trained on a weighted version of the original training set based on the estimated bias. Requiring the test data to be available in training is, however, a major weakness. In the social media post classification setting, the system needs to constantly classify the incoming data. It is infeasible to perform training constantly.

In this paper, we propose a novel learning technique that does not need the test data to be available during training due to the specific nature of our problem, i.e., the positive training data does not have the covariate shift issue.

One obvious solution to this problem is one-class classification (Schölkopf et al. 1999; Tax and Duin, 1999a), i.e., one-class SVM. We simply discard the negative training posts/documents completely because they have the covariate shift problem. Although this is a valid solution, as we will see in the evaluation section, the models built based on one-class SVM perform poorly. Although it is conceivable to use an unsupervised method such clustering, SVD (Alter et al., 2000) or LDA (Blei et al., 2003), supervised learning usually give much higher accuracy.

In our proposed method, instead of performing supervised learning in the original document space based on n-grams, we perform learning in a similarity space. Thus, the key novelty of the method is the transformation from the original *document space* (DS) to a *center-based similarity space* (CBS). In the new space, the covariate shift problem is significantly mitigated, which enables us to build more accurate classifiers. The reason for this is that in CBS based learning the vectors in the similarity space enable SVM (which is the learning algorithm that we use) to find a good boundary of the positive class data based on similarity and to separate it from all possible negative class data, including those negative data that is not represented in training. We will explain this in greater detail in Section 3.5 after we present the proposed algorithm, which we call CBS-L (for CBS Learning).

This paper makes three contributions: First, it formulates a special case of the covariate shift problem. This case occurs frequently in social media data classification as we discussed above. Second, it proposes a novel CBS space based learning method, CBS-L, which avoids the covariate shift problem to a large extent because it is able to find a good similarity boundary of the positive data. Third, it experimentally demonstrates the effectiveness of the proposed method.

2 Related Work

Traditional supervised learning assumes that the training and test examples are drawn from the same distribution. However, this assumption can be violated in many applications. This is especially the case for social media data because of the high topic diversity and constant changes of topics. This problem is known as *covariate shift*, which is a form of sample selection bias.

Sample selection bias was first introduced in econometrics by Heckman (1979). It came into the field of machine learning through the work of Zadrozny (2004). The main approach in machine learning is to first estimate the distribution bias of the training data based on the test data, and then learn using weighted training examples to

compensate for the bias (Bickel et al. 2009).

For example, Shimodaira (2000) and Sugiyama and Muller (2005) proposed to estimate the training and test data distributions using kernel density estimation. The estimated density ratio is then used to generate weighted training examples. Dudik et al. (2005) and Bickel and Scheffer (2007) used maximum entropy density estimation, while Huang et al. (2007) proposed kernel mean matching. Sugiyama et al. (2008) and Tsuboi et al. (2008) estimated the weights for the training instances by minimizing the Kullback-Leibler divergence between the test and the weighted training distributions. Bickel et al. (2009) proposed an integrated model. As we discussed in the introduction, the need for the test data at the training time is a major weakness for social media data classification. The proposed technique CBS-L doesn't have this restriction.

As mentioned in the introduction, one-class classification is a suitable approach to solve the problem. Tax and Duin (1999a and 1999b) proposed a model for one-class classification called Support Vector Data Description (SVDD) to seek a hyper-sphere around the positive data that encompasses points in the data with the minimum radius. In order to balance between model over-fitting and under-fitting, Tax and Duin (2001) proposed a method that tries to use artificially generated outliers to optimize the model parameters. However, their experiments suggest that the procedure to generate artificial outliers in a hyper-sphere is only feasible for up to 30 dimensions. Also, as pointed out by (Khan and Madden, 2010; 2014), one drawback of their methods is that they often require a large dataset and the methods become very inefficient in high dimensional feature spaces. Since text documents are usually represented in a much higher dimensional space, these methods are less suitable for text applications. Manevitz and Yousef (2001) performed one-class text classification using one-class SVM as proposed by Schölkopf et al. (1999). The method is based on identifying outlier data that are representative of the second class. Instead of assuming the origin is the only member of the outlier class, it assumes those data points with few non-zero entries are also outliers. However, as reported in the paper, their methods produce quite weak results (Schölkopf et al., 1999; 2000). Li et al. (2003) presented an improved version of one-class SVM for detecting anomalies. Their idea is to consider all data points that are close to the origin as outliers. Both (Yang and Madden, 2007) and (Tian and Gu, 2010) tried to refine Schölkopf's models by searching optimal parameters. Luo et al., (2007) proposed a cost-sensitive one-class SVM algorithm for intrusion detection. We will see in the experiment section that one-class classification is far inferior to our proposed CBS-L method.

In this work, we propose to represent documents in the similarity space and thus it is related to works on document representation. Alternative document representations have been proposed in the past and have been shown to perform well in many applications (Radev et al., 2000; He et al., 2004; Lebanon 2006; Ranzato and Szummer, 2008, Wang and Domeniconi, 2008). In (Radev et al., 2000), although the centroid sentence/document vector was computed, it was not transformed to a similarity space vector representation. Wang and Domeniconi (2008) proposed to use external knowledge to build semantic kernels for documents in order to improve text classification. In our problem, the main difficulty is that testing negative documents cannot be well covered in training. It is not clear how the enriched document representations could help solve our problem.

Our work is also related to learning from positive and unlabeled examples, also known as PU learning (Denis, 1998; Yu et al. 2002; Liu et al. 2003; Lee and Liu, 2003; Elkan and Noto, 2008; Li et al. 2010). In this learning model, there is a set of labeled positive training data and a set of unlabeled data, but there is no labeled negative training data. Clearly, their setting is different from ours too. There is also no guarantee that the unlabeled data has the same distribution as the future test data.

Our problem is also very different from domain adaption as we work in the same domain. Due to the use of document similarity, our method has some resemblance to learning to rank (Li, 2011; Liu, 2011). However, CBS-L is very different because we perform supervised classification. Our similarity is also center-based rather than pair-wise document similarity, which is also used in (Qian and Liu 2013) for spam detection.

3 The Proposed CBS Learning

We now formulate the proposed supervised learning in the CBS space, called CSB-L. The key difference between CBS learning and the classic document space (DS) learning is in the document representation, which applies to both training and testing documents or posts. In the next subsection, we first give the intuitive idea

and a simple example. The detailed algorithm follows. In Section 3.5, we explain why CBS-L is better than DS-based learning when unexpected negative data appear in the test set.

3.1 Basic Idea

In the proposed CBS-L formulation, each document d is still represented as a feature vector, but the vector no longer represents the document d itself based on n-grams. Instead, it represents a set of similarity values between document d and the center of the positive documents. Specifically, the learning consists of the following steps:

1. Each document d (in the positive or negative class) is first represented with a set of document representations, i.e., *document space vectors* (*ds-vectors*) based on the document itself as in traditional text classification. Each vector denotes one representation of the document. For example, one representation may be based on only unigrams, and another representation may be based on only bigrams. For simplicity, we use only one representation/vector **x** (e.g., unigrams) here to represent d. Note that we use bold lower case letters to represent vectors. Each feature in a *ds*-vector is called a *ds-feature*.
2. A center vector **c** is then computed for each document representation for the positive class documents using the *ds-vectors* of all positive and negative documents of that representation. **c** is thus also a *ds-vector*.
3. Each document d in the positive and negative class is then transformed to a *center-based similarity space vector* $\mathbf{s}_d$ (called a *cbs-vector*). $\mathbf{s}_d$ consists of a set of similarity values between document d's set of ds-vectors, i.e., {**x**} in our case here (since we use only one representation), and the set of corresponding positive class center vectors, i.e., {**c**} in our case:

   $$\mathbf{s}_d = Sim(\{\mathbf{x}\}, \{\mathbf{c}\}),$$

   where *Sim* is a similarity function consisting of a set of similarity measures. Each feature in $\mathbf{s}_d$ is called an *cbs-feature*. $\mathbf{s}_d$ still has the same original class label as d. Let us see an actual example. We assume that our single center vector for the positive class has been computed (see Section 3.2) based on the unigram representation of documents:

   c: 1:1 2:1 6:2

   where *y*:*z* represents a *ds*-feature *y* (e.g., a word) and its feature value (e.g., term frequency, *tf*). We want to transform the following positive document d_1 and negative document d_2 (*ds-vectors*) to their *cbs*-vectors (the first number is the class):

   d_1: 1 1:2 2:1 3:1 d_2: -1 2:2 3:1 5:2

   If we use *cosine* as the first similarity measure in *Sim*, we can generate a *cbs*-feature 1:0.50 for d_1 (as cosine(c, d_1) = 0.50) and a *cbs*-feature 1:0.27 for d_2 (as cosine(c, d_2) = 0.27). If we have more similarity measures, more *cbs*-features will be produced. The resulting *cbs*-vectors for d_1 and d_2 with their class labels, 1 and -1, are:

   d_1: 1 1:0.50 … d_2: -1 1:0.27 …
4. We now have a binary classification problem in the CBS space. This step simply runs a classification algorithm, e.g., SVM, to build a classifier. We use SVM in our work.

3.2 CBS Based Learning

We are given a binary text classification problem. Let $D = \{(d_1, y_1), (d_2, y_2), \ldots, (d_n, y_n)\}$ be the set of training examples, where d_i is a document and $y_i \in \{1, -1\}$ is its class label. Traditional classification directly uses D to build a binary classifier. However, in the CBS space, we learn a classifier that returns 1 for documents that are "close enough" to the center of the training positive documents and -1 for documents elsewhere.

We now detail the proposed technique. As we mentioned above, instead of using one single *ds*-vector to represent a document $d_i \in D$, we use a set R_d of p *ds*-vectors $R_d = \{\mathbf{x}_1^d, \mathbf{x}_2^d, \ldots, \mathbf{x}_p^d\}$. Each vector $\mathbf{x}_i^d$ denotes one document space representation of the document, e.g., unigram representation. We then compute the center of positive training documents, which is represented as a set of p centroids $C = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_p\}$, each of which corresponds to one document space representation in R_d. The way to compute each center $\mathbf{c}_i$ is similar to that in the Rocchio relevance feedback method in information retrieval (Rocchio, 1971; Manning et al. 2008), which uses the corresponding *ds*-vectors of all training positive and negative documents. The detail will be given below. Based on R_d for document d and the center C, we can transform a document d from its document space representations R_d to one center-based similarity vector **cbs-v** by applying a similarity function *Sim* on each element $\mathbf{x}_i^d$ of R_d and its corresponding center $\mathbf{c}_i$. We now detail document transformation.

Training document transformation: The train-

ing data transformation from *ds*-vectors to *cbs*-vectors performs the following two steps:

Step 1: Compute the set C of centroids for the positive class. Each centroid vector $\mathbf{c}_i \in C$ is for one document representation $\mathbf{x}_i^d$. And it is computed by applying the Rocchio method to the corresponding *ds*-vectors of all documents in both positive and negative training data.

$$\mathbf{c}_i = \frac{\alpha}{|D_+|} \sum_{\mathbf{ds}_i^d \in D_+} \frac{\mathbf{x}_i^d}{\|\mathbf{x}_i^d\|} - \frac{\beta}{|D - D_+|} \sum_{\mathbf{x}_j^d \in D - D_+} \frac{\mathbf{x}_j^d}{\|\mathbf{x}_j^d\|}$$

where D_+ is the set of documents in the positive class and |.| is the size function. α and β are parameters, which are usually set empirically. It is reported that using *tf-idf* representation, $\alpha = 16$ and $\beta = 4$ usually work quite well (Buckley et al. 1994). The subtraction is used to reduce the influence of those terms that are not discriminative (i.e., terms appearing in both positive and negative documents).

Step 2: Compute the similarity vector **cbs-v**$_d$ (center-based similarity space vector) for each document $d \in D$ based on its set of document space vectors R_d and the corresponding centroids C of the positive documents.

$$\textbf{cbs-v}_d = Sim(R_d, C)$$

Sim has a set of similarity measures, and each measure m_j is applied to p document representations $\mathbf{x}_i^d$ in R_d and their corresponding centers $\mathbf{c}_i$ in C to generate p similarity features (*cbs*-features) in **cbs-v**$_d$. We discuss the *ds*-features and similarity measures for computing *cbs*-features in the next two subsections.

Complexity: The data transformation step is clearly linear in the number of examples, i.e., n.

Test document transformation: For each test document d, we can use step 2 above to produce a *cbs*-vector for d.

3.3 DS-Features

In order to compute *cbs*-features (center-based similarity space features) for each document, we need to have the *ds*-features of a document and the center of the positive class. We discuss *ds*-features first, which are extracted from each document itself.

Since our task is document classification, we use the popular unigram, bigram and trigram with *tf-idf* weighting as the *ds*-features for a document. These three types of *ds*-features also give us three different document representations.

3.4 CBS-Features

Ds-vectors are transformed into *cbs*-vectors by applying a set of similarity measures on each document space vector and the corresponding center vector. In this work, we employed five similarity measures from (Cha, 2007) to gauge the similarity of two vectors. Based on these measures, we produce 15 CBS features using the unigram, bigram, and trigrams representations of each document. The similarity measures we used are listed in Table 1, where P and Q are two vectors and d represents the dimension of P and Q.

$s_{Cos} = \frac{\sum_{i=1}^{d} P_i Q_i}{\sqrt{\sum_{i=1}^{d} {P_i}^2} \sqrt{\sum_{i=1}^{d} {Q_i}^2}}$
$s_{gow} = 1 - \frac{1}{d} \sum_{i=1}^{d} \left\| \frac{P_i}{\sqrt{\sum_{i=1}^{d} {P_i}^2}} - \frac{Q_i}{\sqrt{\sum_{i=1}^{d} {Q_i}^2}} \right\|$
$s_{Lor} = 1 - \sum_{i=1}^{d} ln(1 + \|P_i - Q_i\|)$
$s_{Dice} = \frac{2 \sum_{i=1}^{d} P_i Q_i}{\sum_{i=1}^{d} {P_i}^2 + \sum_{i=1}^{d} {Q_i}^2}$
$s_{jac} = \frac{\sum_{i=1}^{d} P_i Q_i}{\sum_{i=1}^{d} {P_i}^2 + \sum_{i=1}^{d} {Q_i}^2 - \sum_{i=1}^{d} P_i Q_i}$

Table 1: similarity measures for CBS-Features

3.5 Why Does CBS Space Learning Work?

We now try to explain why CBS learning (CBS-L) can deal with the covariate shift problem, and thus can perform better than document space learning. The reason is that due to the use of similarity features, CBS-L is essentially trying to generate a boundary for the positive training data because similarity is not directional and thus covers all directions in a spherical shape in the space. In classification, the negative data from anywhere or direction outside the spherical shape can be detected. The covariate shift problem will not affect the classification much. Many types of documents that are not represented in the negative training data will still be detected due to their low similarity. For example, in Figure 1, we want to build a SVM classifier to separate positive data represented as black squares and negative data represented as empty circles. The constructed CBS-L classifier would look like a circle (in dashed line) in the original document space

covering the positive data. The size of this (boundary) circle depends on the separation margin between the two classes. Although data points represented by empty triangles are not represented in the negative training data (which has only empty circles) in building the classifier, our classifier is able to identify them as *not positive* at the test time because they are outside the boundary circle.

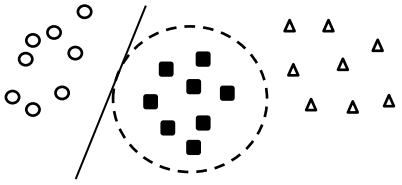

Figure 1: CBS learning vs. DS learning.

If we had used the document space (DS) features to build a SVM classifier, the classifier would be a line (see Figure 1) between the positive data (black squares) and the negative data (empty circles). This line unfortunately will not be able to identify data points represented as empty triangles as not positive because the triangles actually lie on the positive side and would be classified as positive, which is clearly wrong.

4 Experiments

In this section, we evaluate the proposed learning in the center-based similarity space (CBS-L) and compare it with baselines.

4.1 Experimental Dataset

As stated at the beginning of the paper, this work was motivated by the real-life problem of identifying the right social media posts or documents for specific applications. For an effective evaluation, we need a large number of classes in the data to reflect the topic richness and diversity of the social media. The whole data also has to be labeled for evaluation. Using online reviews of a large number of products is a natural choice because there are many types of products and services and there is no need to do manual labeling, which is very labor intensive, time consuming, and error prone. We obtained the Amazon review database from the authors of (Jindal and Liu 2008), and constructed a dataset with reviews of 50 types of products, which we also call 50 topics. Each topic (a type of products) have 1000 reviews. For each topic, we randomly sampled 700 reviews/documents for training and the remaining 300 reviews for testing. Note that although we use this product review collection, we do not perform sentiment classification. Instead, we still perform the traditional topic based classification. That is, given a review, the system decides what type of product the review is about. In our experiments, we use every topic as the positive class. This gives us 50 classification results.

4.2 Baselines

We use three baselines in our evaluation.

Document space one-class SVM (ds-osvm): As we discussed earlier, due to the covariate shift problem in the negative training data, one solution is to drop the negative training data completely to build a one-class classifier. One-class SVM is the state-of-the-art one-class classification algorithm. We apply one-class SVM to the documents in the document space as one of the baselines. One-class SVM was first introduced by Schölkopf et al. (1999; 2000), which is based on the assumption that the origin is the only member of the second class. The data is first mapped into a transformed feature space via a kernel and then standard two-class SVM is employed to construct a hyper-plane that separates the data and the original with maximum margin. As mentioned earlier, there is also the support vector data description (SVDD) formulation for one-class classification proposed by Tax and Duin (1999a; 1999b). SVDD seeks to distinguish the positive class from all other possible data in space. It basically finds a hyper-sphere around the positive class data that contains almost all points in the data set with the minimum radius. It has been shown that the use of Gaussian kernel makes SVDD and One-class SVM equivalent, and the results reported in (Khan and Madden, 2014) demonstrate that SVDD and One-class SVM are comparable when the Gaussian kernel is applied. Thus in this paper, we just use one-class SVM, which is one of the SVM-based classification tools in the LIBSVM[1] library (version 3.20) (Chang and Lin, 2011).

Center-based similarity space one-class SVM (cbs-osvm): Instead of applying one-class SVM to documents in the original document space, this baseline applies it to the CBS space after the documents are transformed to CBS vectors.

SVM: This baseline is the SVM applied in the original document space. Although in this case, there is covariate shift problem, we want to see how serious the problem might be, and how the proposed CBS-L technique can deal with the

[1] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/

	In-training			Out-of-training			Combined		
	precision	recall	F1-score	precision	recall	F1-score	precision	recall	F1-score
	10 topics are used in the training negative class								
ds-osvm							0.154	0.498	0.205
cbs-osvm	0.664	0.453	0.514	0.357	0.442	0.339	0.343	0.452	0.330
SVM	0.678	0.811	0.736	0.176	0.803	0.282	0.160	0.819	0.262
CBS-L	0.796	0.766	0.776	0.384	0.768	0.491	0.368	0.754	0.481
	20 topics are used in the training negative class								
ds-osvm							0.154	0.498	0.205
cbs-osvm	0.561	0.477	0.466	0.430	0.445	0.390	0.364	0.457	0.344
SVM	0.566	0.753	0.643	0.304	0.753	0.422	0.254	0.758	0.371
CBS-L	0.761	0.700	0.723	0.557	0.702	0.608	0.485	0.693	0.558
	30 topics are used in the training negative class								
ds-osvm							0.154	0.498	0.205
cbs-osvm	0.451	0.491	0.393	0.488	0.524	0.407	0.378	0.487	0.355
SVM	0.508	0.721	0.591	0.450	0.722	0.547	0.323	0.726	0.439
CBS-L	0.723	0.650	0.678	0.721	0.644	0.667	0.569	0.649	0.598
	40 topics are used in the training negative class								
ds-osvm							0.154	0.498	0.205
cbs-osvm	0.423	0.482	0.379	0.590	0.511	0.444	0.372	0.486	0.347
SVM	0.456	0.689	0.544	0.641	0.685	0.658	0.374	0.695	0.481
CBS-L	0.697	0.613	0.644	0.848	0.616	0.699	0.639	0.613	0.619

Table 2: Summary results of the 50 topics

problem. We use the SVM tool in LIBSVM.

4.3 Kernels and Parameters

As Khan and Madden (2014) pointed out that one-class SVM performs the best when Gaussian kernel is used, we use Gaussian kernel as well. Manevitz and Yousef (2001) applied one-class SVM to text classification, and the authors reported that one-class SVM works the best with binary feature weighting scheme compared to *tf* or *tf-idf* weighting schemes. Also, they reported that a small number of features (10) with highest document frequency performed the best with Gaussian kernel. We also use binary representation, but found that 10 features are already too many in our case. In fact, 5 features give the best results. Using a small number of features is intuitive because to find the boundary of a very high dimensional space is very difficult. We also tried more features but they were poorer.

For SVM classification in the document space, we use the linear kernel as it has been shown by many researchers that the linear kernel performs the best (e.g., Joachims, 1998; Colas and Brazdil, 2006). We experimented with RBF kernels extensively, but they did not perform well with the traditional document representation. The term weighting scheme is *tf-idf* (Colas and Brazdil, 2006) with no feature selection.

For our proposed method CBS-L, we use *tf-idf* values of unigram, bigram and trigram to represent a document in three ways in the document space. As mentioned earlier, five document similarity functions are used to transform document space vectors to CBS space vectors. And in order to filter out less useful features for the center vector of the positive class, we performed feature selection in the document space using the classic information gain method (Yang and Pedersen, 1997) to empirically choose the most effective 100 features for the positive class.

For all the kernels, we use the default parameter settings in the LIBSVM systems. We tried to tune the parameters, but did not get better results.

4.4 Results

We now present the experiment results. As mentioned above, we treat each topic as the positive class. This gives 50 tests. To test the effect of covariate shift, we also vary the number of topics in the negative class. We used 10, 20, 30, and 40 topics in the training negative class. The test set always has 49 topics of negative data.

For each setting, we give three sets of results for the positive class, which is the target topic data that we are interested in obtaining through classification. Each set of results includes the standard measures of precision, recall, and F1-score for the positive class. The three sets are:

1. *In-training*: In this case, the test negative data

contains only data from those topics used in training. This is the classical supervised learning setting where the training and test data are randomly drawn from the same distribution.

2. *Not-in-training*: In this case, the test negative set contains only data from the other topics not used in training. The classical setting of supervised learning does not deal with this problem. This represents covariate shift.
3. Combined: In this case, the test data contains both in-training and not-in-training negative topics. Due to the use of not-in-training test data, this is also not the classical setting.

Due to a large number of experiment results, we cannot report all the details. Table 2 summarizes the results. Notice that for ds-osvm, it does not make sense to have in-training and not-in-training results because it does not use any training negative data. Thus, there is only one set of results for "Combined," which is duplicated in the table for easy comparison. However, note that cbs-osvm uses negative data for training in order to compute the center for the positive class.

From the table, we can make the following observations (since there are many numbers, we only focus on F1-scores).

1. The proposed CBS-L method performs markedly better than all baselines. For the results of in-training, not-in-training, and combined, CBS-L is consistently better in all cases than all baselines. Even for in-training, CBS-L perform better than SVM. This clearly shows the superiority of the proposed CBS-L method.
2. ds-osvm performs poorly. cbs-osvm is much better because it uses the negative data in feature selection and center computation.
3. SVM in the document space performed poorly (Combined) when only a small number of negative topics are used in training. It gets better than both one-class SVM baselines when more negative topics are used in training (see the reason in the next point).
4. Finally, we can also see that with the number of training negative topics increases, the results of the combined case of both SVM and CBS-L improve. This is expected because with the increased number of negative topics for training, the number of not-in-training negative topics for testing decreases and the covariate shift problem gets smaller. We can also see that cbs-osvm, SVM and CBS-L's F1-scores for not-in-training improve with the increased training negative topics due to the same reason. However, their F1-scores drop for in-training because with more negative

topic	ds-osvm	cbs-osvm	SVM	CBS-L
Amplifier	0.125	0.360	0.406	0.597
Automotive	0.041	0.031	0.240	0.383
Battery	0.266	0.425	0.433	0.656
Beauty	0.079	0.401	0.470	0.618
Cable	0.131	0.028	0.231	0.500
Camera	0.376	0.361	0.433	0.523
CD Player	0.154	0.274	0.344	0.585
Clothing	0.046	0.234	0.292	0.486
Computer	0.117	0.225	0.328	0.455
Conditioner	0.075	0.195	0.381	0.519
Fan	0.408	0.581	0.581	0.724
Flashlight	0.273	0.487	0.528	0.744
Graphics Card	0.419	0.473	0.552	0.631
Headphone	0.298	0.338	0.432	0.533
Home Improvement	0.039	0.032	0.178	0.233
Jewelry	0.362	0.579	0.632	0.800
Kindle	0.107	0.387	0.416	0.685
Kitchen	0.042	0.118	0.197	0.261
Lamp	0.091	0.249	0.374	0.487
Luggage	0.105	0.482	0.506	0.482
Magazine Subscriptions	0.406	0.597	0.796	0.858
Mattress	0.435	0.562	0.603	0.702
Memory Card	0.134	0.256	0.367	0.508
Microphone	0.103	0.223	0.25	0.417
Microwave	0.378	0.577	0.637	0.735
Monitor	0.136	0.345	0.312	0.513
Mouse	0.493	0.580	0.552	0.779
Movies TV	0.146	0.507	0.641	0.682
Musical Instruments	0.073	0.241	0.446	0.575
Network Adapter	0.164	0.483	0.481	0.596
Office Products	0.040	0.193	0.327	0.346
Patio Lawn Garden	0.043	0.226	0.295	0.483
Pet Supplies	0.098	0.447	0.524	0.584
Pillow	0.491	0.640	0.781	0.888
Printer	0.549	0.557	0.624	0.859
Projector	0.230	0.459	0.482	0.805
Rice Cooker	0.571	0.616	0.692	0.942
Shoes	0.224	0.524	0.585	0.793
Speaker	0.241	0.251	0.253	0.410
Subwoofer	0.147	0.268	0.346	0.401
Table Chair	0.141	0.496	0.571	0.703
Tablet	0.069	0.234	0.142	0.424
Telephone	0.099	0.034	0.144	0.167
Tent	0.289	0.465	0.428	0.764
Toys	0.088	0.029	0.331	0.449
Video Games	0.424	0.387	0.508	0.705
Vitamin Supplement	0.052	0.026	0.341	0.527
Wall Clock	0.401	0.582	0.607	0.777
Watch	0.362	0.553	0.543	0.775
Webcam	0.155	0.304	0.372	0.645
Average	0.205	0.355	0.439	0.598

Table 3: F1-score for each positive topic or class in the combined case

topics, the data becomes more skewed, which hurts in-training classification.

To give a flavor of the detailed results for each topic (product), we give the full results for one setting with 30 randomly selected topics as the training negative data (Table 3). The results in the table are F1-scores of the combined case.

5 Conclusion

The ability to get relevant posts accurately about a topic from social media is a challenging problem. This paper attempted to solve this problem by identifying and dealing with the technical issue of covariate shift. The key idea of our technique is to transform document representation from the traditional n-gram feature space to a similarity based space. Our experimental results show that the proposed method CBS-L outperformed strong baselines by large margins.

Acknowledgments

This research was partially supported by the NSF grants IIS-1111092 and IIS-1407927, and a Google faculty award.

Reference

Alter, O., Brown, P.O. and Bostein, D. 2000. Singular Value Decomposition for Genome-Wide Expression Data Processing and Modeling. *Proc. Nat',l Academy of Science,* vol. 97, no. 18, pp. 10101-10106, Aug.

Blei, D. Ng, A. and Jordan, M., 2003. Latent dirichlet allocation, *The Journal of Machine Learning Research*, 3, p.993-1022, 3/1/2003

Buckley, C., Salton, G., Allan, J. 1994. The Effect of Adding Relevance Information in a Relevance Feedback Environment, *Proceedings of SIGIR Conference*.

Bickel, S., Bruckner, M., and Scheffer. 2009. T. Discriminative learning under covariate shift. *Journal of Machine Learning Research*.

Bickel S. and Scheffer T. 2007. Dirichlet-enhanced spam filtering based on biased samples. *Advances in Neural Information Processing Systems*.

Cha, S.-H. 2007. Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, 1(4):300--307.

Chang, C-C. and Lin, C-J. 2011. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1--27:27, http://www.csie.ntu.edu. tw/~cjlin/libsvm

Colas, F. and Brazdil. P. 2006. Comparison of SVM and some older classification algorithms in text classification tasks. *Artificial Intelligence in Theory and Practice*. IFIP International Federation for Information Processing, pp. 169-178.

Denis, F., PAC learning from positive statistical queries. *ALT*, 1998.

Radev, D., Jing, H. and Budzikowska, M. 2000. Centroid-based summarization of multiple documents: Sentence extraction, utility-based evaluation, and user studies. In *ANLP/NAACL Workshop on Summarization*, Seattle, April.

Dudik, M., Schapire, R., and Phillips, S. 2005. Correcting sample selection bias in maximum entropy density estimation. *Advances in Neural Information Processing Systems*.

Elkan, C. and Noto, K. 2008. Learning classifiers from only positive and unlabeled data. *KDD*, 213-220.

He, X., Cai, D., Liu, H. and Ma, W.-Y. 2004. Locality Preserving Indexing for Document Representation. *Proc. Of SIGIR*.

Heckman, J. 1979. Sample selection bias as a specification error. *Econometrica*, 47:153–161.

Huang, J., Smola, A. and Gretton, A., Borgwardt K., and Scholkopf B. 2007. Correcting sample selection bias by unlabeled data. *Advances in Neural Information Processing Systems*.

Joachims, T. 1998. Text categorization with support vector machines: Learning with many relevant features. *ECML*.

Jindal, N. and Liu, B. 2008. Opinion Spam and Analysis. *Proceedings of the ACM Conference on Web Search and Data Mining*.

Khan, S., and Madden, M. 2010. A survey of recent trends in one class classification. *Artificial Intelligence and Cognitive Science*, volume 6206 of Lecture Notes in Computer Science. 188–197.

Khan, S. and Madden, M. 2014. One-Class Classification: Taxonomy of Study and Review of Techniques. *The Knowledge Engineering Review*, 1-30.

Lebanon, G. 2006. Sequential document repre-

sentations and simplicial curves. *UAI*.

Lee, W. S. and Liu, B. 2003. Learning with Positive and Unlabeled Examples Using Weighted Logistic Regression. *ICML*.

Li, H. 2011. *Learning to Rank for Information Retrieval and Natural Language Processing*. Morgan & Claypool publishers.

Li, K., Huang, H., Tian, S. and Xu, W. 2003. Improving One-class SVM for anomaly detection. *Proc. of the Second International conference on Machine Learning and Cybernetics*, volume 5, pages 3077–3081.

Li, X., Liu, B. and Ng. S.-K. 2010. Negative Training Data can be Harmful to Text Classification. *EMNLP*.

Liu, B, Dai, Y., Li, X., Lee, W-S. and Yu. P. 2003. Building text classifiers using positive and unlabeled examples. *ICDM*.

Liu. T. 2011. *Learning to Rank for Information Retrieval*. Springer.

Luo, J., Ding, L., Pan, Z., Ni, G. and Hu, G. 2007. Research on cost-sensitive learning in one-class anomaly detection algorithms. *Autonomic and Trusted Computing*, volume 4610 of Lecture Notes in Computer Science.

Manevitz, L. and Yousef. M. 2001. One-class SVMs for document classification. *Journal of Machine Learning research*.

Manning, C. D., Prabhakar R., and Hinrich, S. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

Qian, T. and Liu, B. 2013. Identifying Multiple Userids of the Same Author. *EMNLP*.

Ranzato, M. and Szummer, M. 2008. Semi-supervised learning of compact document representations with deep networks. *ICML*.

Rocchio, J. 1971. Relevant feedback in information retrieval. In G. Salton (ed.). *The smart retrieval system: experiments in automatic document processing*.

Schölkopf, B., Williamson, R., Smola, A., Taylor, J. and Platt, J. 2000. Support vector method for novelty detection. *Neural Information Processing Systems*, pages 582–588.

Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, A. and Williamson, R. 1999. Estimating the support of a high-dimensional distribution. *Technical Report*, Microsoft Research, MSR-TR-99-87.

Shimodaira, H. 2000. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90:227–244.

Sugiyama, M. and Muller, K.-R. 2005. Input-dependent estimation of generalization error under covariate shift. *Statistics and Decision*, 23(4):249–279.

Sugiyama, M., Nakajima, S., Kashima, H., von Bunau P., and Kawanabe M. 2008. Direct importance estimation with model selection and its application to covariate shift adaptation. *Advances in Neural Information Processing Systems*.

Tax, D. and Duin, R. 1999a. Data domain description using support vectors. *Proceedings ESAN99*, Brussels. 251-256

Tax, D. and Duin, R. 1999b. Support vector domain description. *Pattern Recognition Letters* 20. 1191-1199

Tax, D. and Duin, R. 2001. Uniform object generation for optimizing one-class classifiers. *J. of Machine Learning Research*, 2:155–173.

Tian, J. and Gu, H. 2010. Anomaly detection combining one-class SVMs and particle swarm optimization algorithms. *Nonlinear Dynamics*, 61(1-2): 303–310.

Tsuboi, J., Kashima, H., Hido, S., Bickel, S., and Sugiyama, M. 2008. Direct density ratio estimation for large-scale covariate shift adaptation. *Proceedings of the SIAM International Conference on Data Mining* (*SDM*).

Wang, P. and Domeniconi, C. 2008. Building semantic kernels for text classification using Wikipedia, *KDD*.

Yang, Y. and Pedersen, J. O. 1997. A comparative study on feature selection in text categorization. *ICML*.

Yang, L., and Madden, M. 2007. One-class support vector machine calibration using particle swarm optimization. *AICS*, Dublin.

Yu, H., Han, J. and Chang, K. 2002. PEBL: Positive example based learning for Web page classification using SVM. *KDD*, 239-248.

Zadrozny, B. 2004. Learning and evaluating classifiers under s ample selection bias, *ICML*.