Learning SVM Ensemble Classifiers from Positive and Unlabeled Data

Marc Claesen MARC.CLAESEN@ESAT.KULEUVEN.BE
Department of Electrical Engineering-ESAT, SCD-SISTA – IBBT Future Health Department
KU Leuven, 3001 Leuven, Belgium

Frank De Smet
National Alliance of Christian Mutualities, 1030 Brussels, Belgium

Bart De Moor BART.DEMOOR@ESAT.KULEUVEN.BE
Department of Electrical Engineering-ESAT, SCD-SISTA – IBBT Future Health Department
KU Leuven, 3001 Leuven, Belgium

Editor: N/A

Abstract

We present a novel semi-supervised approach to train support vector machine (SVM) classifiers using positive and unlabeled data, called the positive and unlabeled ensemble (PUE). PUE consists of bagging biased SVMs, trained with randomized parameters on subsamples of the positive and unlabeled training sets. By randomizing SVM parameters and bootstrapping the training set, high diversity is induced between classifiers. This diversity is exploited by aggregating individual classifiers into an ensemble. Our approach has been benchmarked on two fully labeled data sets against other popular methods. PUE consistently yielded higher positive predictive values than other approaches, combined with competitive correct rates. Benchmarks additionally revealed that PUE is robust against false positives in the training set.

Keywords: classification, semi-supervised learning, ensemble learning, support vector machine

1. Introduction

Training binary classifiers on **p**ositive and **u**nlabeled data is referred to as PU learning. This form of semi-supervised learning differs from supervised classification in its lack of negative labels but the availability of unlabeled data. Practical applications of PU learning typically feature high-dimensional input spaces and/or large, imbalanced training sets with a small amount of labeled (positive) and a large amount of unlabeled training instances.

PU learning approaches are recommended when negative labels cannot be acquired or when the data contains a large amount of false negatives. PU learning is currently commonly used for document classification (Manevitz et al., 2001; Liu et al., 2003), web page classification (Yu et al., 2002) and intrusion detection (Lazarevic et al., 2003).

Other fields like bioinformatics and chemoinformatics also encounter PU learning problems, but specialized approaches are not always adopted. Such applications include gene prioritization (Aerts et al., 2006), biomarker discovery (Kitteringham et al., 2009) and virtual screening of drug compounds (Shoichet, 2004). In these applications, high positive

© 2012 M. Claesen, F. De Smet & B. De Moor.

predictive value (PPV[1]) of classifiers is of paramount importance, because further testing of selected candidates is expensive and/or time-consuming.

We present a novel approach for PU learning, which consists of bagging biased SVM classifiers with randomized training parameters. Our work was inspired by *bagging SVM* as introduced by Mordelet and Vert (2010), which we briefly discuss in Section 2.2. Our approach consistently obtains higher PPV than the current state-of-the-art biased SVM (Liu et al., 2003) in PU learning contexts, while maintaining reasonable recall. Our approach, called **p**ositive and **u**nlabeled **e**nsemble (PUE) is described in detail in Section 3.

PUE was compared with mapping convergence (MC) (Yu, 2005), biased SVM (BSVM) (Liu et al., 2003) and bagging SVM (BAG) (Mordelet and Vert, 2010). We have benchmarked the approaches using artificial and real data sets (see Section 4.1). All data sets used in our experiments are completely labeled. Knowledge of negative labels was not used during training but the presence of negative labels enabled us to compute standard performance measures on test sets to benchmark classifier performance.

1.1. PU Learning Characteristics

To formalize PU learning, we adopt the $< \mathbf{x}, y, s >$-triple notation used by Elkan and Noto (2008). Let $\mathbf{x}$ be an instance of the negative or positive class, $\mathbb{N} \sim P_-$ resp. $\mathbb{P} \sim P_+$ and let $y \in \{0,1\}$ be a binary label. Let $s = 1$ if the instance $\mathbf{x}$ is labeled, and let $s = 0$ otherwise. Formally, $\mathbf{x}$, y, and s can be considered random variables and some fixed unknown overall distribution $p(\mathbf{x}, y, s)$ over triples $< \mathbf{x}, y, s >$ exists. Since in PU learning only positive data is labeled, the following holds:

$$p(y = 0|\mathbf{x}, s = 1) = 0 \tag{1}$$

Labeled positive instances are commonly assumed to be selected completely at random from all positive instances. In that premises, Elkan and Noto (2008) have derived the following result:

$$p(y = 1|\mathbf{x}) = p(s = 1|\mathbf{x})/c \tag{2}$$

where $c = p(s = 1|y = 1)$ is an unknown constant. In other words, if we want to construct a classifier f to rank instances $\mathbf{x}$ according to the probability they belong to class $y = 1$ we could directly use a classifier g trained to distinguish positive and unlabeled instances.

Scott and Blanchard (2009) reach a similar conclusion by observing that the distribution of unlabeled data $\mathbb{U}$ can be written in function of P_+, P_- and the probability π of positives in $\mathbb{U}$: $P_u = (1-\pi)P_- + \pi P_+$. The optimal test for $h_+ : \mathbf{x} \sim P_+$ vs. $h_- : \mathbf{x} \sim P_-$ is identical to the optimal test for $h_+ : \mathbf{x} \sim P_+$ vs. $h_u : \mathbf{x} \sim P_u$ because the likelihood ratios are related by a monotone transformation (for $\pi \in (0,1]$):

$$\frac{h_u(\mathbf{x})}{h_+(\mathbf{x})} = (1-\pi) + \pi \frac{h_-(\mathbf{x})}{h_+(\mathbf{x})} \tag{3}$$

As such, Scott and Blanchard (2009) conclude that learning to discriminate between positive and unlabeled instances is a valid proxy for the true goal of learning to discriminate between positives and negatives.

[1] PPV is defined as $TP/(TP+FP)$, recall is defined as $TP/(TP+FN)$.

2. Related Work

PU learning approaches can be split into two categories: (1) approaches that try to infer likely negatives $\hat{\mathcal{N}}$ from $\mathcal{U}$ and then train supervised algorithms on $\mathcal{P}$ and $\hat{\mathcal{N}}$ and (2) approaches that consider $\mathcal{U}$ to be a negative set with noise on its labels. When *very* few labeled examples are available, the border between classification and clustering fades and semi-supervised clustering techniques are recommended (Alzate and Suykens, 2012).

Techniques that try to infer a negative set $\hat{\mathcal{N}}$ usually involve two stages. Examples include S-EM (Liu et al., 2002), mapping convergence (MC) (Yu, 2005), and ROC-SVM (Li and Liu, 2003). In our benchmarks, the MC algorithm was tested. It uses a one-class classifier (such as one-class SVM (Schölkopf et al., 2001) or support vector domain description (Tax and Duin, 1999)) to infer an initial set $\hat{\mathcal{N}}$. Subsequently, the algorithm iteratively trains supervised classifiers on $\mathcal{P}$ and $\hat{\mathcal{N}}$ and assigns extra negative labels.

The second type of approaches assign higher confidence to positive instances than unlabeled ones. A first way to achieve this is by weighting individual data points, such as in weighted logistic regression (Elkan and Noto, 2008; Lee and Liu, 2003). Another approach is by changing the penalties on misclassification during training, as is done in biased SVM (Liu et al., 2003), bagging SVM (Mordelet and Vert, 2010) and RT-SVM (Liu et al., 2005). Biased SVM is especially popular in this category and is considered the state-of-the-art (see Section 2.1). Bagging SVM was proposed by Mordelet and Vert (2010) to speed up biased SVM, but if configured slightly different it can achieve higher performance. Because our approach is similar to bagging SVM and biased SVM, we discuss them briefly in Sections 2.1 and 2.2.

Statistical approaches to estimate the probability of positive points in the unlabeled set have been reported in (Lee and Liu, 2003; Elkan and Noto, 2008; Scott and Blanchard, 2009). Once these probabilities have been estimated, these approaches typically proceed by employing weights for the entire unlabeled set or for each individual data point.

2.1. Biased SVM (BSVM)

Liu et al. (2003) first applied biased SVMs for PU learning. Biased SVM is a supervised technique in which the penalty for misclassification differs per class. When using biased SVM for PU learning, the unlabeled set is considered negative with noise on labels. During training, misclassification of positive points is penalized more than misclassification of unlabeled points. In the context of PU learning, the optimization problem for training BSVM can be written as:

$$\begin{aligned}
&\textbf{min}\ \frac{1}{2}||\mathbf{w}||_2^2 + C_+ \sum_{i\in\mathcal{P}} \xi_i + C_u \sum_{i\in\mathcal{U}} \xi_i \\
&\textbf{subject to}\ y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, && i = 1,\dots,N \\
&\qquad \xi_i \geq 0, && i = 1,\dots,N \\
&\qquad C_+ \geq C_u
\end{aligned}$$

The misclassification penalties C_+ and C_u require tuning. SVM formulations with unequal penalties across classes have been used previously to tackle imbalanced data sets (Osuna et al., 1997).

2.2. Bagging SVM (BAG)

Mordelet and Vert introduce bagging SVM as a meta-algorithm which consists of aggregating classifiers trained to discriminate $\mathcal{P}$ from a small, random subsample of $\mathcal{U}$ (Mordelet and Vert, 2010). They state that PU learning problems have a particular structure that leads to instability of classifiers, namely the sensitivity of classifiers to the empirical ratio of positive examples in the unlabeled training set. Bagging is a common technique used to improve the performance of instable classifiers (Breiman, 1996).

In bagging SVM, random subsamples of $\mathcal{U}$ are made and BSVM classifiers are trained to discriminate $\mathcal{P}$ from each subsample. By subsampling $\mathcal{U}$, the empirical ratio of positives is varied between subsamples. This induces variability in the classifiers which the aggregation procedure can then exploit. We will refer to bagging SVM using BAG_T where T is the amount of individual BSVM classifiers in the bag.

3. Positive and Unlabeled Ensemble (PUE)

The positive and unlabeled ensemble is a meta-algorithm, which consists of bagging randomly configured biased SVMs. It is inspired by and conceptually similar to bagging SVM as described in Section 2.2.

Bagging predictors is a method for generating multiple versions of a predictor and using these to get an aggregated predictor. The vital element is the instability of the prediction method. If perturbing the training set can cause significant changes in the predictor constructed, then bagging can improve accuracy (Breiman, 1996). When bagging classifiers, each constituent classifier is called a weak classifier.

In the remainder of this Section, we discuss how to induce diversity between weak classifiers in the ensemble. Later we discuss the aggregation procedure and finally we demonstrate some properties of PUE. Throughout this text we use $|\mathcal{X}|$ to denote the cardinality of the set $\mathcal{X}$ and $x : y : z$ to denote the series of numbers x to z with increments y.

3.1. Diversifying Weak Classifiers

When introducing bagging, Breiman (1996) induced variation exclusively by perturbing the training set (through bootstrapping). In bagging SVM, this is done by bootstrap subsampling $\mathcal{U}$ (Mordelet and Vert, 2010). In PUE, diversity between classifiers is promoted even more. First of all, perturbation of training sets is increased by varying the sizes of bootstrap samples from both positive and unlabeled training data per weak classifier. Subsequently training parameters are changed between weak classifiers. These measures lead to a very diverse set of classifiers which increases performance and robustness of the ensemble.

3.1.1. Perturbing Training Sets

In PUE, diverse training sets are constructed through bootstrap sampling of $\mathcal{P}$ and $\mathcal{U}$, where the size of samples $\mathcal{P}^t$ and $\mathcal{U}^t$ is varied between rounds $t = 1 : 1 : T$:

- $posratio^t$: the relative size of the bootstrap sample from $\mathcal{P}$. Randomly selected from $\{0.3 : 0.1 : 0.7\}$. The following holds: $|\mathcal{P}^t| = posratio^t \times |\mathcal{P}|$.

- B^t: used to determine $|\mathcal{U}^t|/|\mathcal{P}^t|$. Randomly selected from $\{2:2:20\}$. The following holds: $|\mathcal{U}^t| = K \times B^t \times |\mathcal{P}^t|$. The parameter K is discussed later.
- Draw bootstrap samples $\mathcal{P}_t$ and $\mathcal{U}_t$ from $\mathcal{P}$ resp. $\mathcal{U}$ with the correct size.

Using these bootstrap samples, T weak classifiers ψ_t are trained to discriminate $\mathcal{P}_t$ from $\mathcal{U}_t$ $(t = 1 : 1 : T)$ which are then aggregated into an ensemble.

3.1.2. Randomizing Biased SVM Parameters

In addition to perturbing the training sets used to obtain weak classifiers, which is standard for bagging, the training parameters of weak classifiers are randomized to induce additional diversity. The idea stems from the observation that even a broken clock is right twice a day. While poorly configured classifiers may not achieve high correct rates, they still capture *some* information in the data, which optimally tuned classifiers might not. By aggregating many (potentially suboptimal) classifiers, the ensemble achieves strong performance and robustness.

Since we used biased SVM, the parameters in question are C_+ and C_u (see Section 2.1). To ensure low bias, the penalty on misclassifications during training must be high. This increases model complexity and decreases stability. These properties are exploited through aggregation. Parameter randomization is performed as follows:

- C_u^t: the misclassification penalty for instances in $\mathcal{U}$. Randomly selected from $\{0.5 : 0.5 : 3\} \times numfeats$, where $numfeats$ is the amount of features.
- C_{ratio}^t: misclassification trade-off (discussed later). Randomly selected from $\{0.1 : 0.1 : 1\}$. The following holds: $C_+^t = K \times C_{ratio}^t \times B^t \times C_u^t$.

Evidently, parameters cannot be chosen completely at random. The sets of possible values contain sensible options per parameter. We used the same options in PUE as in the search grid we used to tune individual BSVM classifiers. The meaning of K and C_{ratio}^t can be deduced from the following equations:

$$|\mathcal{U}^t| = K \times B^t \times |\mathcal{P}^t| \tag{4}$$

$$C_+^t = K \times B^t \times C_u^t \times C_{ratio}^t \tag{5}$$

K is an input parameter (which can be tuned) that relates the average sizes of $\mathcal{P}^t$ and $\mathcal{U}^t$ in the ensemble. For 2Gauss (Section 4.1.1), $K = 1$ was optimal, for Reuters (Section 4.1.2) $K = 10$ was best. K can be considered the average trade-off between weak classifier performance and stability (similar to K for bagging SVM). The meaning of the misclassification trade-off C_{ratio}^t follows from rearranging Equations (4) and (5):

$$C_{ratio}^t = \frac{C_+^t \times |\mathcal{P}^t|}{C_u^t \times |\mathcal{U}^t|} \tag{6}$$

In bagging SVM, C_{ratio}^t always equals 1 (Mordelet and Vert, 2010).

3.2. Aggregation

In a classification context, weak classifiers are typically aggregated through majority voting. In PUE, the (binary) prediction of all weak classifiers in the ensemble is averaged:

$$f_{PUE}(\mathbf{x}) = \frac{1}{T} \sum_{t=1}^{T} \psi_t(\mathbf{x}) \tag{7}$$

The outcome lies between 0 and 1. A threshold $0 \leq \tau \leq 1$ is used to assign class labels. Higher thresholds yield more conservative classifiers. In general, the ensemble tends to be slightly overconservative when using majority votes ($\tau = 0.5$), which is symptomized by high PPV and low recall. Therefore, we consistently used $\tau = 0.4$. When PUE is used for ranking, thresholding is not required. Using τ, sensitivity and specificity can be controlled. By varying τ, ROC and PR curves can be computed.

3.3. Overview of the Positive and Unlabeled Ensemble

The PUE algorithm is listed in Algorithm 1. It consists of training T randomized biased SVM classifiers on T bootstrap samples from $\mathcal{P}$ and $\mathcal{U}$ with varying size.

```
input    : $\mathcal{U}$, $\mathcal{P}$, $K$=balance, $T$=number of bootstraps
output   : PUE classifier $f_{PUE} : \mathcal{I} \mapsto [0,\ 1]$
internal: posratio, B,Cratio, $\mathbf{C_u}$

for $t \leftarrow 1$ to $T$ do
    randomly select values $\mathsf{posratio}^t$, $\mathsf{B}^t$ from posratio, B;
    random sample $\mathcal{P}^t$, $\mathcal{U}^t$ from $\mathcal{P}$, $\mathcal{U}$ with replacement;
    • $|\mathcal{P}^t| = \mathsf{posratio}^t \times |\mathcal{P}|$;
    • $|\mathcal{U}^t| = K \times \mathsf{B}^t \times |\mathcal{P}^t|$;

    randomly select values $\mathsf{C}^t_{ratio}$, $\mathsf{C}^t_u$ from $\mathbf{C_{ratio}}$, $\mathbf{C_u}$;
    $\mathsf{C}^t_+ \leftarrow K \times \mathsf{B}^t \times \mathsf{C}^t_{ratio} \times \mathsf{C}^t_u$;
    train biased SVM $\psi_t$ using training data $\mathcal{P}^t$, $\mathcal{U}^t$ and parameters $\mathsf{C}^t_+$, $\mathsf{C}^t_u$ ;
    • $\psi_t : \mathcal{I} \mapsto \{0,\ 1\}$;
end
return
    $f_{PUE} = \frac{1}{T} \sum_{t=1}^{T} \psi_t$
```

Algorithm 1: The PUE meta-algorithm, which returns a PUE classifier that maps the input space $\mathcal{I}$ to the interval $[0, 1]$.

Figure 1 shows the evolution of a PUE classifier in function of the amount of weak classifiers in the ensemble T. Performance increases with the size of the ensemble but stagnates for large values of T. The relative change in predictions $\Delta_f(T)$ can be used to assess whether the ensemble is large enough. $\Delta_f(T)$ values for the training and testing sets are very similar.

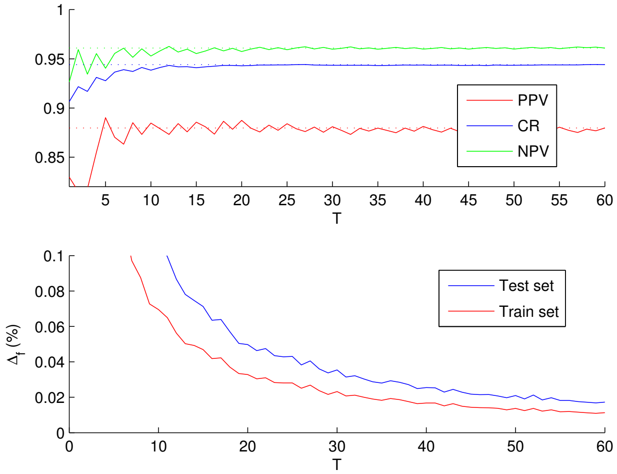

Figure 1: Evolution of the PUE classifier for increasing T. Top: correct rate, positive predictive value and negative predictive value. Bottom: relative change in predictions on the given set $\mathcal{T}$ between f_T and f_{T-1}: $\Delta_f(T) = \mathbf{d}^T\mathbf{d}/|\mathcal{T}|$ with $\mathbf{d} = f_T(\mathbf{x}) - f_{T-1}(\mathbf{x})$. Values shown were computed using all labels.

4. Benchmarking Setups

We used the SVM implementations in LIBSVM 3.11 in all experiments (Chang and Lin, 2011). Two data sets were used to benchmark the performance of PUE in comparison to mapping convergence, biased SVM and bagging SVM. For each algorithm, parameters were tuned in every individual experiment. Parameters were evaluated via 10-fold cross-validation, optimal values were selected by maximizing the following score function introduced by Lee and Liu (2003):

$$\texttt{pu\_score} = \frac{\texttt{PPV} \times \texttt{recall}}{Pr(y=1)} = \frac{\texttt{recall}^2}{Pr\big(f(\mathbf{x})=1\big)} \tag{8}$$

Where $Pr(f(\mathbf{x}) = 1)$ is the probability for the classifier f to assign a positive label. This score can be computed in PU learning contexts, in contrast to the correct rate. Our experimental results show that `pu_score` is a worthy surrogate, which consistently yields classifiers with good performance.

In practical settings, erroneous labeling may occur. Therefore we consider equation (1) an assumption about the labeled data rather than dogma. It is known that when training a SVM, a small number of false positive training data could be detrimental (Yu et al., 2002). The impact of incorrect labels is exacerbated in semi-supervised learning. For every PU learning approach, we have investigated the impact of breaking the rule in Equation (1) by introducing false positives in the training data. In our experiments, the configuration parameter β controls the amount of false positives:

$$\beta = p(y=0|x, s=1) \tag{9}$$

In our implementation of mapping convergence, we used 1-SVM (Schölkopf et al., 2001) as initial classifier in the mapping stage, followed by C-SVC as defined by Cortes and Vapnik (1995) in the convergence stage.

4.1. Data Sets

We used two data sets to benchmark the performance of our algorithm in comparison to other popular methods. Our data sets contain positive and negative labels, which enables us to assess performance objectively and exactly.

The first data set we have used is an artificial one in two dimensions, called 2Gauss. The second benchmark data set is a subset of Reuters-21578, which is used often to compare document classification algorithms.

4.1.1. 2Gauss

This data set consists of random samples from two overlapping, 2D normal distributions $\mathcal{N}(\mu, \Sigma)$, denoted as $\mathbb{P}$ (positives) and $\mathbb{N}$ (negatives):

- Properties of $\mathbb{P}$: $\mu = [0, 0]$ and Σ is the 2×2 unity matrix.
- Properties of $\mathbb{N}$: $\mu = [3, 0]$ and Σ is the 2×2 unity matrix.

We used RBF kernels for all experiments on this data set. Benchmarks were conducted for three configurations, in which we varied the ratio of positive instances in the unlabeled set using $\alpha = p(y = 1|x, s = 0)$ and the ratio of false positives in the training set ($\mathcal{P}$), using $\beta = p(y = 0|x, s = 1)$.

In all benchmark configurations, the size of positive training sets $\mathcal{P}$, unlabeled training sets $\mathcal{U}$ and test sets $\mathcal{T}$ were 70, 1050 and 480, respectively. By varying α and β, the amount of positives in $\mathcal{P}$, $\mathcal{U}$ and $\mathcal{T}$ was changed. The three benchmark configurations we used are summarized in Table 1.

config	α	β	n_+ in $\mathcal{P}$	n_+ in $\mathcal{U}$	n_+ in $\mathcal{T}$
1	0%	0%	70	0	30
2	20%	0%	70	210	90
3	20%	10%	63	210	87

Table 1: Summary of the three benchmark configurations on 2Gauss, with varying amount of positive instances in positive training set $\mathcal{P}$, unlabeled training set $\mathcal{U}$ and test set $\mathcal{T}$.

To ensure reproducibility of our results, we conducted 50 experiment rounds per benchmark configuration. In each round r, new sets $\mathcal{P}_r$, $\mathcal{U}_r$ and $\mathcal{T}_r$ were generated using which classifiers were tuned, trained and evaluated. Performance measures listed in Table 1 are the means of performance measures over all rounds.

4.1.2. Reuters

The Reuters-21578 data set[2] is a commonly used benchmark for text mining algorithms, particularly for document classification (Manevitz et al., 2001; Li and Liu, 2003). We have

[2] The Reuters-21578, Distribution 1.0 test collection is available from http://www.daviddlewis.com/resources/testcollections/reuters21578

used a preprocessed subset[3], also used by Cai et al. (2011), which will subsequently be referred to as Reuters. Documents belonging to several topic categories were removed.

Reuters contains 8293 documents, vectorized in tf-idf format, of which 5946 were used in training. The documents contain 18933 distinct terms (features) and are categorized into one of 65 topic categories. In our experiments we used 8 topic categories to represent the positive class. Figure 2 shows the frequency of these topics in the training and test sets.

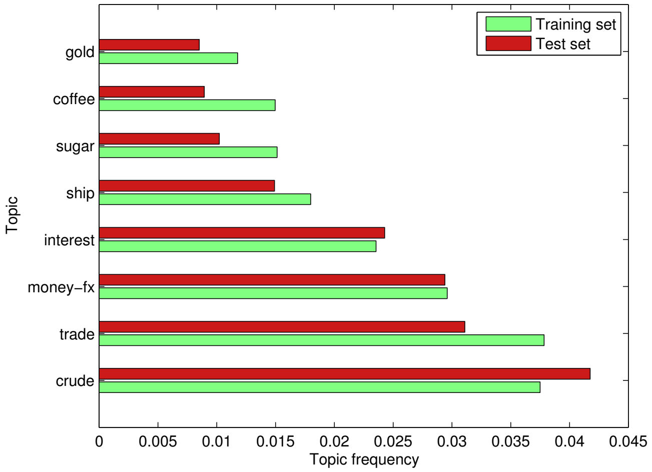

Figure 2: Frequency of selected topics in the Reuters data set.

Per experiment, one topic category is selected as the positive class and documents belonging to the other 64 topics form the negative class. In our experiments, κ represents the fraction of positive training instances which were moved to the unlabeled set. Please note the contrast between κ and α which was used in experiments on 2Gauss. The linear kernel was used in all experiments on Reuters.

5. Results and discussion

We will refer to BSVM, BAG and PUE as the *BSVM family*, since all of them share traits in terms of performance. Recall that both BAG and PUE form ensembles of individual BSVM classifiers.

Our experimental results are summarized below. For most experiments, all approaches obtained high negative predictive values and correct rates. This follows directly from the imbalance of all data sets ($n_{neg} \gg n_{pos}$). Differences between approaches become apparent in terms of PPV and recall. For both data sets and all PU configurations, the PUE classifier consistently yields highest positive predictive values. In terms of recall, bagging SVM appears to be best.

[3] Available at http://www.cad.zju.edu.cn/home/dengcai/Data/TextData.html

5.1. 2Gauss

Figure 3 shows ROC curves for PUE and BAG on the 2Gauss data set in configuration 3 (see Section 4.1.1). PUE obtains higher AUC than BAG and BSVM (96.9% vs. 95.1% and 93.5%, respectively), which testifies to the merit of diversity in ensemble learning.

Davis and Goadrich (2006) proved that if a curve dominates in ROC space, it also dominates in Precision-Recall space[4]. Since the ROC curve of PUE dominates all others for false positive rates (FPR) below ±0.7, its PR curve will also dominate. For this data set, a false positive rate of 0.7 translates into a PPV below 0.25. Therefor we can conclude that the PUE classifier has higher recall than all others for any desired PPV above 0.25.

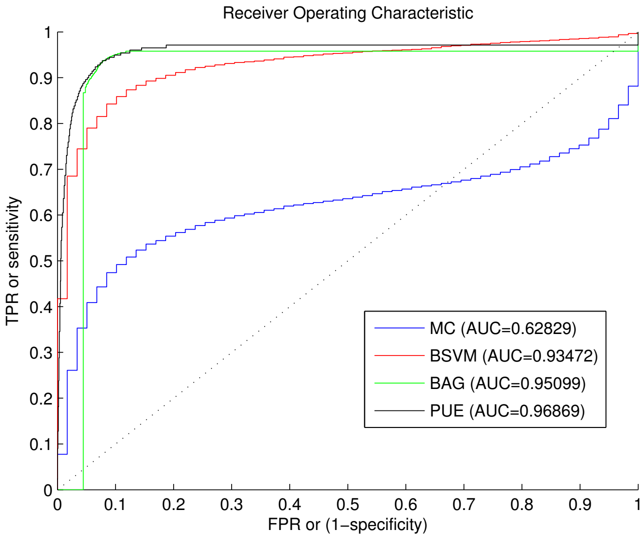

Figure 3: Receiver Operating Characteristic (ROC) curves of several approaches on 2Gauss with $\alpha = 0.2$ and $\beta = 0.1$ (see Section 4.1.1).

Table 2 summarizes important performance measures per configuration on 2Gauss, averaged over all rounds. Results show that classifiers in the BSVM family end up with comparable correct rates and outperform MC. PUE outperforms other approaches in terms of PPV in both PU learning configurations. Large variations exist between approaches in terms of recall. Bagging SVM obtains maximum recall across the board, followed closely by PUE. In terms of NPV and recall, MC performs very bad in configurations 2 and 3. This means that MC does not obtain a good surrogate negative set, for reasons unknown to us.

Figure 4 shows PUE predictions per configuration. Adding positives to $\mathcal{U}$ results in more conservative ensembles. This is also apparent in Table 2, where recall drops while PPV increases between configuration 1 and 2. Additionally, introducing false positives into $\mathcal{P}$ does *not* make the classifier less conservative. This robustness against false positives is obtained by bootstrapping $\mathcal{P}$ when forming a PUE classifier and is a desirable trait.

[4] Precision is a synonym for positive predictive value. PR space is suited for imbalanced data sets.

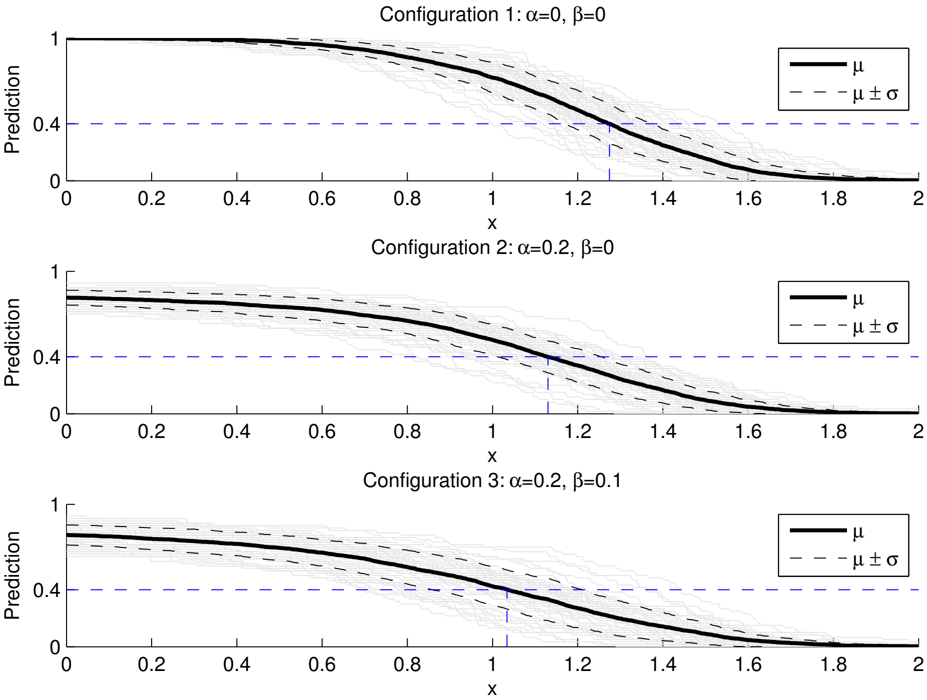

Figure 4: Predictions by PUE_{60} classifiers trained on 2Gauss over all rounds of the three configurations for test points on the line segment $[x, 0]$, $0 < x < 2$. The solid black line represents the mean μ, black dashed lines represent $\mu\pm$ standard deviation σ over all rounds. Gray lines show the predictions of individual classifiers. Positive prediction threshold was set at 0.4.

config	**algorithm**	**CR**	**PPV**	**NPV**	**recall**	**ratio**
$\alpha = 0\%$	MC	**0.9727**	**0.8740**	0.9779	0.6627	0.1241
	BSVM	0.9640	0.7021	0.9866	0.7987	0.5313
$\beta = 0\%$	BAG_{60}	0.9375	0.5062	**0.9933**	**0.9047**	6.2139
	PUE_{60}	0.9541	0.5963	0.9920	0.8833	2.8432
$\alpha = 20\%$	MC	0.7912	0.8572	0.7896	0.0630	0.6384
	BSVM	0.9119	0.8493	0.9279	0.7117	0.5104
$\beta = 0\%$	BAG_{60}	0.9336	0.8139	**0.9737**	**0.9103**	10.8291
	PUE_{60}	**0.9394**	**0.8626**	0.9621	0.8660	6.3687
$\alpha = 20\%$	MC	0.8000	0.7981	0.7988	0.0596	0.5165
	BSVM	0.9297	0.8495	0.9456	0.7783	0.5056
$\beta = 10\%$	BAG_{60}	0.9328	0.7977	**0.9747**	**0.9062**	12.7407
	PUE_{60}	**0.9431**	**0.8764**	0.9606	0.8514	7.2987

Table 2: Average results on 2Gauss with $\alpha = p(y = 1|x, s = 0)$ the fraction of positives in $\mathcal{U}$ and $\beta = p(y = 0|x, s = 1)$ the amount of false positives. Listing correct rate (CR), PPV, NPV and SV ratio (n_{SV}/n_{train}).

5.2. Reuters

Results are summarized in Tables 3 to 5. In Tables 3 and 4, baseline is a (supervised) C-SVC which was trained using the unlabeled set as negatives. These tables show the importance of using the proper approaches in PU learning since both the supervised SVM and one-class SVM (1-SVM) are outperformed by far in a PU learning context (Table 4). In Table 4 the characteristics of PUE are clear. While other approaches struggle to maintain consistently high PPV, PUE yields excellent results. The PPV of one-class SVM is quite low in both configurations. This indicates that one-class SVM is not conservative enough for this problem. Note, however, that one-class SVM was never designed to optimize PPV.

The main effect we want to emphasize is the change in PPV per approach between configurations. Table 5 shows that PUE is the *only* classifier whose PPV is robust against false positives in the training set ($\beta \neq 0$). While other approaches' PPV drop below 80%, PUE still sports an impressive 88.4%. Unfortunately, there is no such thing as a free lunch, the price paid is in terms of loss in recall. This shows that PUE contrasts to other approaches when false positives are introduced in the training set: PUE is the only approach that becomes more conservative and by doing so safeguards its PPV.

In Tables 3 and 4, we observe that the PPV of PUE across topics is clearly better than any other approach, both in value and consistency. In terms of PPV, PUE is never beaten by more than 4% while it outperforms all others by up to 19% for some topics. This makes PUE an ideal approach when high positive predictive value is desired.

5.3. Model Size

SVM prediction speed scales linearly with the model size, expressed by the amount of support vectors n_{SV}. Our results concerning model size are consistent across all test configurations (see ratio in Tables 2 and 5):

$$n_{SV}^{MC} \approx n_{SV}^{BSVM} \ll n_{SV}^{PUE} < n_{SV}^{BAG}$$

BSVM models are generally larger than MC models because misclassifications are penalized more during BSVM training (through C_+ and C_u). The ensemble classifiers obtained using BAG and PUE are significantly larger than MC and BSVM.

Because the individual classifiers of BAG and PUE share many identical support vectors, n_{SV}^{BAG} and n_{SV}^{PUE} are large. Model size is directly proportional to the amount of weak classifiers T. Ensemble models can be shrunk by reducing the amount of classifiers that make up the ensemble by pruning some weak classifiers in order to end up with *support models* as described by Hamers et al. (2003), using techniques like the lasso (Tibshirani, 1994). Optimized implementations solve the problem of duplicate support vectors.

Prediction speed could be improved significantly by caching intermediate results of computations with common support vectors, but standard SVM libraries do not offer this functionality. This only works when a single kernel is used in all weak classifiers of the ensemble, so kernels may not be diversified when we cache intermediate results. In our current approach, kernel parameters were not diversified anyway.

topic	**baseline**	**1-SVM**	**MC**	**BSVM**	**BAG$_{60}$**	**PUE$_{60}$**
crude	0.9111	0.2050	0.9111	0.9111	0.9121	**0.9250**
trade	0.8310	0.2582	0.8310	0.8310	0.8310	**0.8772**
money-fx	0.6567	0.0858	**0.7241**	0.6716	0.6515	0.6863
interest	0.8462	0.1546	0.8113	0.8462	0.8148	**0.8667**
ship	0.8148	0.0317	0.8148	0.8148	0.8148	**0.8571**
sugar	0.9091	0.2055	**0.9130**	**0.9130**	**0.9130**	0.9000
coffee	**0.9524**	0.1613	**0.9524**	**0.9524**	**0.9524**	0.9444
gold	0.8668	0.1250	**1.0000**	**1.0000**	0.9412	**1.0000**

Table 3: PPV of algorithms on 8 topics in Reuters, $\beta = 0\%$ and $\kappa = 0\%$.

topic	**baseline**	**1-SVM**	**MC**	**BSVM**	**BAG$_{60}$**	**PUE$_{60}$**
crude	0.8592	0.0610	0.8316	0.7531	0.7763	**0.9259**
trade	0.7885	0.0585	0.6344	0.7586	0.7593	**0.9487**
money-fx	0.5000	0.0357	**0.6744**	0.5686	0.4909	0.6316
interest	0.8000	0.0549	0.7381	0.7347	0.7708	**0.8718**
ship	0.5909	0.0133	0.7368	0.8667	0.6500	**1.0000**
sugar	0.8824	0.0171	**0.9000**	0.7826	0.8947	0.8824
coffee	0.8889	0.0197	0.8421	**0.9375**	0.8000	0.9333
gold	0.7500	0.0164	0.6923	0.8462	0.8000	**0.8750**

Table 4: PPV of algorithms on 8 topics in Reuters, $\beta = 10\%$ and $\kappa = 20\%$.

config	**algorithm**	**CR**	**PPV**	**NPV**	**recall**	**ratio**
$\kappa = 0\%$	MC	**0.9922**	0.8697	0.9949	0.7767	0.0558
	BSVM	**0.9922**	0.8675	**0.9951**	**0.7844**	0.0555
$\beta = 0\%$	BAG$_{100}$	0.9920	0.8539	**0.9951**	0.7883	3.8831
	PUE$_{60}$	0.9911	**0.8750**	0.9933	0.6866	2.2453
$\kappa = 20\%$	MC	0.9874	0.7972	0.9908	0.5797	0.0577
	BSVM	0.9877	0.8560	0.9901	0.5459	0.0601
$\beta = 0\%$	BAG$_{100}$	**0.9893**	0.8385	**0.9919**	**0.6281**	4.0246
	PUE$_{60}$	0.9887	**0.8993**	0.9901	0.5476	2.0704
$\kappa = 20\%$	MC	**0.9877**	0.7562	**0.9923**	**0.6175**	0.0765
	BSVM	0.9867	0.7810	0.9907	0.5828	0.0811
$\beta = 10\%$	BAG$_{100}$	0.9860	0.7428	0.9903	0.5557	6.1782
	PUE$_{60}$	0.9875	**0.8836**	0.9889	0.4897	2.7951

Table 5: Average results on Reuters with κ the fraction of positives that were moved to the unlabeled set and $\beta = p(y = 0|x, s = 1)$ the amount of false positives. Listing correct rate (CR), positive predictive value (PPV), negative predictive value (NPV) and SV ratio (n_{SV}/n_{train}).

6. Conclusion

Experimental results show that the PUE approach yields excellent results. By forming ensembles with high diversity between weak classifiers, state-of-the-art biased SVMs were improved upon in PU learning scenarios. The ensembles sport higher correct rates and positive predictive value than using a single BSVM classifiers. Additionally, our ensembles are more robust against false positives in the training set than other approaches.

An added benefit of PUE is the fact that it is easy to tune with only one parameter (K). All of these characteristics make our approach valuable in many PU learning settings, specifically when positive predictive value is critical. Another advantage of our approach is that training the weak classifiers in the ensemble is straightforward to parallellize, which can significantly speed up training time for PUE models when resources are available. When dealing with huge data sets, this is certainly worth consideration.

Due to the large amount of (duplicate) support vectors retained in a PUE model, its use may be prohibited for very large data sets due to the total size of the model. To tackle this problem, we have implemented the ensembles more efficiently, such that duplicate support vectors are not stored or evaluated more than once. These implementations will be made publicly available.

References

Stein Aerts, Diether Lambrechts, Sunit Maity, Peter Van Loo, Bert Coessens, Frederik De Smet, Leon-Charles Tranchevent, Bart De Moor, Peter Marynen, Bassem Hassan, Peter Carmeliet, and Yves Moreau. Gene prioritization through genomic data fusion. *Nat Biotech*, 24(5):537–544, May 2006. ISSN 1087-0156.

Carlos Alzate and Johan A. K. Suykens. A semi-supervised formulation to binary kernel spectral clustering. In *2012 IEEE World Congress on Computational Intelligence (IEEE WCCI/IJCNN 2012)*, Brisbane, Australia, June 2012.

Leo Breiman. Bagging predictors. *Mach. Learn.*, 24(2):123–140, August 1996. ISSN 0885-6125.

Deng Cai, Xiaofei He, and Jiawei Han. Locally consistent concept factorization for document clustering. *IEEE Trans. on Knowl. and Data Eng.*, 23(6):902–913, June 2011. ISSN 1041-4347.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, September 1995. ISSN 0885-6125.

Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 233–240, New York, NY, USA, 2006. ACM. ISBN 1-59593-383-2.

Charles Elkan and Keith Noto. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 213–220, New York, NY, USA, 2008. ACM.

Bart Hamers, Johan A. K. Suykens, and Bart De Moor. Coupled transductive ensemble learning of kernel models. Internal Report 03-172, ESAT-SISTA, K.U. Leuven, Leuven, Belgium, 2003.

Neil R. Kitteringham, Rosalind E. Jenkins, Catherine S. Lane, Victoria L. Elliott, and B. Kevin Park. Multiple reaction monitoring for quantitative biomarker analysis in proteomics and metabolomics. *Journal of chromatography. B, Analytical technologies in the biomedical and life sciences*, 877(13):1229–1239, May 2009. ISSN 1873-376X.

Ar Lazarevic, Aysel Ozgur, Levent Ertoz, Jaideep Srivastava, and Vipin Kumar. A comparative study of anomaly detection schemes in network intrusion detection. In *In Proceedings of the Third SIAM International Conference on Data Mining*, 2003.

Wee Sun Lee and Bing Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, page 2003, 2003.

Xiaoli Li and Bing Liu. Learning to classify texts using positive and unlabeled data. In *IJCAI'03: Proceedings of the 18th international joint conference on Artificial intelligence*, pages 587–592, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc.

Bing Liu, Wee Sun Lee, Philip S. Yu, and Xiaoli Li. Partially supervised classification of text documents. In *ICML '02: Proceedings of the Nineteenth International Conference on Machine Learning*, pages 387–394, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc. ISBN 1-55860-873-7.

Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S. Yu. Building text classifiers using positive and unlabeled examples. In *Proceedings of the Third IEEE International Conference on Data Mining*, ICDM '03, pages 179–, Washington, DC, USA, 2003. IEEE Computer Society. ISBN 0-7695-1978-4.

Zhigang Liu, Wenzhong Shi, Deren Li, and Qianqing Qin. Partially supervised classification – based on weighted unlabeled samples support vector machine. In *Proceedings of the First international conference on Advanced Data Mining and Applications*, ADMA'05, pages 118–129, Berlin, Heidelberg, 2005. Springer-Verlag. ISBN 3-540-27894-X, 978-3-540-27894-8.

Larry M. Manevitz, Malik Yousef, Nello Cristianini, John Shawe-taylor, and Bob Williamson. One-class SVMs for document classification. *Journal of Machine Learning Research*, 2:139–154, 2001.

Fantine Mordelet and Jean-Philippe Vert. A bagging SVM to learn from positive and unlabeled examples. July 2010. URL http://hal.archives-ouvertes.fr/hal-00523336.

Edgar Osuna, Robert Freund, and Federico Girosi. Support Vector Machines: Training and Applications. Technical Report AIM-1602, 1997.

Bernhard Schölkopf, John C. Platt, John C. Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Comput.*, 13(7):1443–1471, July 2001. ISSN 0899-7667.

C. Scott and G. Blanchard. Novelty detection: Unlabeled data definitely help. In *AISTATS: Artificial Intelligence and Statistics, JMLR: W&CP 5*, 2009. URL http://jmlr.csail.mit.edu/proceedings/papers/v5/scott09a.html.

Brian K. Shoichet. Virtual screening of chemical libraries. *Nature*, 432(7019):862–865, December 2004. ISSN 0028-0836.

David M. J. Tax and Robert P. W. Duin. Support vector domain description. *Pattern Recognition Letters*, 20:1191–1199, 1999.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

Hwanjo Yu. Single-class classification with mapping convergence. *Mach. Learn.*, 61(1-3): 49–69, November 2005. ISSN 0885-6125.

Hwanjo Yu, Jiawei Han, and Kevin C. Chang. PEBL: positive example based learning for web page classification using svm. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 239–248, New York, NY, USA, 2002. ACM Press. ISBN 158113567X.