Pattern Recognition Letters 41 (2014) 53–64

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

On incrementally using a small portion of strong unlabeled data for semi-supervised learning algorithms ☆

Thanh-Binh Le, Sang-Woon Kim *

Department of Computer Engineering, Myongji University, Yongin 449-728, Republic of Korea

ARTICLE INFO

Article history:
Available online 6 September 2013

Keywords:
Semi-supervised learning
Semi-supervised MarginBoost
Incrementally reinforced selection strategy

ABSTRACT

The aim of this paper is to present an incremental selection strategy by which the classification accuracy of semi-supervised learning (SSL) algorithms can be improved. In SSL, both a limited number of labeled and a multitude of unlabeled data are utilized to learn a classification model. However, it is also well known that the utilization of the unlabeled data is not always helpful for SSL algorithms. To efficiently use them in learning the classification model, some of the unlabeled data that are deemed useful for the learning process are selected and given the correctly estimated labels. To address this problem, especially when dealing with semi-supervised MarginBoost (SSMB) algorithm (d'Alché-Buc et al., 2002), in this paper, two selection strategies, named simply recycled selection and incrementally reinforced selection, are considered and empirically compared. Our experimental results, obtained with well-known benchmark data sets, including SSL-type benchmarks and some UCI data sets, demonstrate that the latter, i.e., selecting only a *small* portion of *strong* examples from the available unlabeled data in an *incremental* fashion, can compensate for the shortcomings of the existing SSMB algorithm. Moreover, compared to the former, it generally achieves better classification accuracy results.

1. Introduction

In semi-supervised learning (SSL) approach, a large amount of unlabeled data, $U = \{(x_j)\}_{j=1}^{n_u}$, $x_j \in \mathbb{R}^d$, together with labeled data, $L = \{(x_i, y_i)\}_{i=1}^{n_l}$, $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$, are used to build better classifiers. That is, SSL exploits the examples of U in addition to the labeled counterparts to improve the performance on a classification task, leading to a performance improvement of supervised learning (SL) algorithms with a multitude of unlabeled data.

However, it is also well known that the utilization of U is not always helpful for SSL algorithms. In particular, it is not guaranteed that adding U to the training data, T, i.e., $T = L \cup U$, leads to a situation in which we can improve the classification performance (Ben-David et al., 2008; Lu, 2009; Zhu, 2006). Therefore, if we can know more about confidence levels involved in classifying U, we could easily choose some of the informative data and include it when training weak classifiers. This idea has been used in several SSL algorithms, including Self-training (McClosky et al., 2008; Rosenberg et al., 2005), Co-training (Blum and Mitchell, 1998; Du et al., 2011), Cluster-then-Label methods (Goldberg et al., 2009; Goldberg, 2010; Singh et al., 2008), and SemiBoost (Mallapragada et al., 2009). Specifically, in SemiBoost, Mallapragada et al. measured the pairwise similarity to guide the selection of a subset of U at each iteration and to assign (pseudo) labels to them. That is, they first computed the confidence of all of the examples of U based on the prediction made by an ensemble classifier and the similarity. They then selected a few examples of higher confidence to re-train the ensemble classifier together with L. The selecting-and-training step was repeated for some number of iterations or until some termination criterion was met.

From this consideration, to improve the classification performance of SSL algorithms further, especially when dealing with the semi-supervised MarginBoost (SSMB) algorithm (d'Alché-Buc et al., 2002), in this paper we propose a modified SSMB algorithm in which we use the discriminating U in an *incremental* fashion rather than in batch mode (Cesa-Bianchi et al., 2006; Wang et al., 2010). In both SemiBoost and the modified SSMB, some discriminative instances are first selected from U and are then used to train the classification model in addition to L. However, the two algorithms differ in how they construct the training data T. In the modified SSMB, the cardinality of T is increased incrementally as the

☆ This work was supported by the National Research Foundation of Korea funded by the Korean Government (NRF-2012R1A1A2041661). A preliminary *short* version of this paper (Le and Kim, 2012) was presented at ICPRAM2012, the 1st International Conference on Pattern Recognition Applications and Methods, Vilamoura, Portugal, in February 2012.

* Corresponding author. Tel.: +82 31 330 6437; fax: +82 31 335 9998.
E-mail address: kimsw@mju.ac.kr (S.-W. Kim).

0167-8655/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.patrec.2013.08.026

iterations are repeated, while, in SemiBoost, the cardinality is always the same when executing the learning iterations. In this context,[1] the problem to be solved is the correct manner of choosing them incrementally; how to measure the confidence and/or how to choose strong examples from U.

Several strategies have been investigated to address this incremental learning task, which include incremental principal component analysis (IPCA) (Hall and Martin, 1998; Hall et al., 2000; Zhao et al., 2006), incremental linear discriminant analysis (ILDA) (Pang et al., 2005; Wang et al., 2010), incremental learning in non-stationary environments (Mulhlbaier and Polikar, 2007), Learn++ (Polikar et al., 2004), etc. Incremental learning algorithms are of learning new information from additional data that may later become available, after a classifier has already been trained using a previously available database (Cesa-Bianchi et al., 2006; Polikar et al., 2001; Polikar et al., 2004). In (Polikar et al., 2004), for example, the incremental algorithm, named Learn++, takes advantage of synergistic generalization performance of an ensemble of classifiers, in which each classifier is trained with a strategically chosen data. The ensemble of classifiers then is combined through a weighted majority voting procedure.

As mentioned previously, in this paper we study an improvement of SSMB (d'Alché-Buc et al., 2002) by sampling a subset of the strong examples from the available unlabeled data at each iteration in the incremental fashion, named *incrementally reinforced selection*. This sampling process for selection is a modification of the *simply recycled selection* scheme employed for SemiBoost (Mallapragada et al., 2009). The difference between both schemes is twofold: first, for the simply recycled selection scheme of SemiBoost the amount of the selected examples is fixed over boosting iterations, whereas for the incrementally reinforced selection, we use a decreased amount of examples, but keeping the previously selected examples in the training set. The second difference is the method of labeling the examples of U. For the simply recycled selection, SemiBoost uses the confidence (i.e., $sign(p-q)$, which is described in Section 2.2), while for the incrementally reinforced selection we use the nearest neighbor (NN) rule.

The main contribution of this paper is that it demonstrates that the classification accuracy of a SSL approach, i.e., SSMB, can be improved by utilizing a small portion of U, selected by the incrementally reinforced scheme, as well as the labeled training data. Also, a comparison of the simply recycled selection and the incrementally reinforced selection schemes has been performed in two fashions: traditional feature-based classification (Fukunaga, 1990) and recently developed dissimilarity-based classification (Pekalska and Duin, 2005). In particular, some of the critical questions concerning the strategies employed in the present work are: *why is NN used to label the examples selected from U, instead of the confidence measure, knowing that the latter has a theoretical justification? what are the drawbacks of original SSMB and SemiBoost, leading to the lower classification accuracy? why is the present modified SSMB better than the original SSMB and the SemiBoost?*, etc.

The remainder of the paper is organized as follows. In Section 2, we first provide a brief introduction to SSMB and SemiBoost algorithms. Then, in Section 3, we present a method of improving SSMB by incrementally utilizing a small amount of strong unlabeled examples for each training stage. In Sections 4 and 5, we continuously present the experimental setup and the results obtained with well-known, real-life benchmark data sets. Finally, in Section 6, we present our concluding remarks as well as some feature works that deserve further study.

[1] When solving the problem of using a large unlabeled data and, especially, the existence of two different, somewhat redundant sources of information about examples to boost performance of a SSL algorithm, a co-training strategy (Blum and Mitchell, 1998; Du et al., 2011) can be considered. This discussion on the availability of the co-training strategy is beyond the scope of this paper.

2. Related work

In this section, we briefly review the semi-supervised MarginBoost (SSMB) and SemiBoost algorithms. The details of these algorithms can be found in the related literature, including (d'Alché-Buc et al., 2002; Mallapragada et al., 2009; and Mason et al., 2000).

2.1. SSMB

As mentioned previously, using a training dataset T consisting of L and U, SSMB determines an ensemble classifier $g_t(x)=\sum_{\tau=1}^{t}\bar{\alpha}_\tau h_\tau(x)$, where $\bar{\alpha}_\tau=\frac{\alpha_\tau}{|\alpha_\tau|}$ and $h_\tau\in\mathcal{H}$ is the weak learner with output in $[-1,1]$, such that the probability of $sign(g_t(x_i))\neq y_i$ is minimized. That is, SSMB minimizes the cost function $\mathcal{C}$, which is defined with any scalar decreasing function c of the margin ρ:

$$\mathcal{C}(g_t)=\sum_{i=1}^{L}c(\rho_L(g_t(x_i),y_i))+\sum_{i=1}^{U}c(\rho_U(g_t(x_i))), \tag{1}$$

where $\rho_L(g_t(x_i),y_i)=y_i g_t(x_i)$ and $\rho_U(g_t(x_i))=g_t(x_i)^2$.

With the definition of the ensemble classifier $g_t(x)$ for L, we can easily obtain the increment in the values of the labeled margins, as follows:

$$\rho_L=\sum_{\tau=1}^{t}\alpha_\tau y_i h_\tau(x_i), \tag{2}$$

where the multiplication $y_i h_\tau(x_i)$ denotes whether the classifier is good or bad. When the classifier is good, the margin ρ_L increases when $(h_\tau(x_i),y_i)=(1,1)$ or $(-1,-1)$. When the classifier is bad, however, the margin decreases as $(h_\tau(x_i),y_i)=(-1,1)$ or $(1,-1)$. On the other hand, the increasing/decreasing of unlabeled margins depends on the output of the weak classifier h_{t+1} at each t. Because the weak classifier can be 1 or -1, U-margin at the $t+1$ iteration will be greater than U-margin at t by the square of α_t:

$$\rho_U=\left(\sum_{\tau=1}^{t}\alpha_\tau h_\tau(x)\right)^2. \tag{3}$$

In SSMB, the criterion quantity, J_t^T, where $T=L\cup U$, has two terms: $J_t^T=J_t^L+J_t^U$. First, instead of using the minimized $\mathcal{C}(g_t)$ exactly at each base classifier, h_{t+1} can be sought to maximize $-\langle\nabla\mathcal{C}(g_t),h_{t+1}\rangle$ of the inner product (d'Alché-Buc et al., 2002). Therefore, the weighted cost function of L to be maximized can be expressed as:

$$J_t^L=\sum_{x_i\in L}w_t(i)y_i h_{t+1}(x_i), \tag{4}$$

where $w_t(i)=\frac{c'(\rho_L(g_t(x_i),y_i))}{\sum_{x_j\in L}c'(\rho_L(g_t(x_j),y_j))}$.

The second term can then be computed as follows:

$$J_t^U=\sum_{x_i\in U}w_t(i)\frac{\partial\rho_U(g_t(x_i))}{\partial g_t(x_i)}h_{t+1}(x_i), \tag{5}$$

where $\partial\rho_U(g_t(x_i))/\partial g_t(x_i)=2g(x_i)$ and the two instances of $w_t(i)$ in Eqs. (4) and (5) are computed as follows:

$$w_t(i)=\begin{cases}\frac{c'(\rho_L(g_t(x_i),y_i))}{\sum_{x_j\in T}w_{t-1}(j)}, & \text{if } x_i\in L,\\ \frac{c'(\rho_U(g_t(x_i)))}{\sum_{x_j\in T}w_{t-1}(j)}, & \text{if } x_i\in U.\end{cases} \tag{6}$$

On the basis of what we have briefly discussed, an algorithm for SSMB is formalized as follows: