2012 11th International Conference on Machine Learning and Applications

Semi-Supervised Learning on Single-View Datasets by Integration of Multiple Co-Trained Classifiers

Jelena Slivka[1*], Ping Zhang[2*], Aleksandar Kovačević[1], Zora Konjović[1], Zoran Obradović[2]
[1]Computing and Control Department, Faculty of Technical Sciences, University of Novi Sad, Serbia
[2]Center for Data Analytics and Biomedical Informatics, Temple University, Philadelphia, PA 19122, USA
[1]{slivkaje, kocha78, ftn_zora}@uns.ac.rs
[2]{ping, zoran.obradovic}@temple.edu
[*]These authors contributed equally to this work

***Abstract*—We propose a novel semi-supervised learning algorithm, called IMCC, designed for co-training classifiers on single-view datasets. Our method runs the co-training algorithm for a predefined number of times, each time using a different random split of features. Thus, a set of diverse co-training classifiers is created. Each of these classifiers then labels each of the examples for which we want to determine the class label. In this way, each example for classification is assigned multiple labels. We then treat this as a problem of learning from inconsistent and unreliable annotators in a multi-annotator problem setting and estimate the single hidden true label for each example. In experimental results obtained on 25 benchmark datasets of various properties IMCC outperformed five considered alternative methods for co-training on single-view datasets, and resulted in a statistical tie with a Naive Bayes classifier trained using a much larger set of labeled examples.**

Keywords- semi-supervised learning; ensemble methods; co-training; multiple annotation

I. INTRODUCTION

Semi-supervised learning is a machine learning technique that makes use of both labeled and unlabeled data with the goal of achieving higher accuracy while demanding less human effort.

A number of semi-supervised learning methods have been proposed, but their efficiency tends to depend on various circumstances. One of the major semi-supervised learning methods is co-training [1]. Successful co-training application is ensured when the dataset has a natural partitioning of the features in two disjoint subsets (views) where each view is sufficient for learning and conditionally independent of the other view given the class label. Co-training exploits the two views in order to train two classifiers using the available training examples. Then, iteratively, each classifier selects and labels some unlabeled examples in order to improve the accuracy of the other classifier by providing it with unknown information. However, in practice, co-training application is limited because the needed optimal feature split is usually unknown.

In this paper, we propose a novel semi-supervised learning algorithm based on **I**ntegration of **M**ultiple **C**o-trained **C**lassifiers, which we call IMCC, designed for co-training on single-view datasets. Our method runs the co-training algorithm for a predefined number of times, each time using a different random split of features. Thus, a set of diverse co-training classifiers is created. Each of these classifiers then labels each of the examples for which we want to determine the class label. In this fashion, each example for classification is assigned multiple labels. We then treat this as a problem of learning from inconsistent and unreliable annotators, and use GMM-MAPML [2], an unsupervised method designed for estimation of true labels when given only multi-annotator situations.

This paper is organized as follows. Section II presents the related work. Section III describes our IMCC algorithm. Section IV presents the conducted experiment and the achieved results. Finally, section V concludes the paper.

II. RELATED WORK

One approach to solving the problem of co-training on single view datasets is by designing a methodology for developing a good approximation of the optimal feature split. Some promising results have been achieved in this field [3-7], but this proved to be a difficult task, as the relation between the characteristics of the views and the performance of co-training has not been sufficiently understood. Moreover, research [4] indicates that given a small training dataset as in real-world situations where co-training is called for, the sufficiency and independence assumptions cannot be reliably verified, making the split methods unreliable and application of co-training uncertain.

Another approach to this problem is based on methodologies that combine co-training with ensemble learning. Such an approach is mainly focused on exploiting an ensemble of classifiers instead of two single classifiers in the co-training algorithm [8-10]. These methods can significantly boost the performance of co-training, but usually require a relatively large initial training set in order to build the initial ensemble of diverse and accurate classifiers.

In this paper, we compare our proposed method to a method called MaxInd [3], designed for approximation of an optimal feature split for co-training. MaxInd is designed to create two maximally independent views given the class label, based on the conditional independence requirement [1]. Also, we compare our method to a recently proposed

978-0-7695-4913-2/12 $26.00 © 2012 IEEE
DOI 10.1109/ICMLA.2012.83

Random Split Statistic Algorithm (RSSalg) for co-training on single view datasets [11]. In RSSalg, co-training is applied for a predefined number of times, each time using a different random split of features. Each run of co-training produces a different enlarged training set, consisting of initial labeled data and data labeled in the co-training process. Examples from enlarged training sets are combined in a final training set, and pruned in order to keep only the most confidently labeled ones. The final classifier in RSSalg is obtained by training the base learner on a set created in this way. Pruning of the examples is done by selection of the most reliable and most informative cases. RSSalg has achieved some promising results [11], but its disadvantage was the introduction of threshold parameters to the co-training setting, which highly affect the performance of RSSalg and need to be carefully tuned for its successful application. To experiment with an alternative method of combining the co-training classifiers gained in the first step of RSSalg, we used a majority voting method (named MV in the experiment of this paper) to integrate the co-training classifiers. However, if most of the classifiers in the ensemble perform poorly, majority voting will perform badly as well.

In this paper, unlike in all of the previous studies, we combined co-training and multiple-annotation settings, and used an unsupervised algorithm to integrate multiple co-training based classifiers. We also performed systematic comparisons of the accuracy of 6 co-training methods as well as 2 additional Naive Bayes classifiers trained by different number of examples on 25 benchmark datasets of various properties.

III. METHOD

In this section we first briefly review the GMM-MAPML algorithm as it forms the basis of our method. Then we introduce our proposed IMCC method.

A. Review of the GMM-MAPML Algorithm

The GMM-MAPML algorithm [2] is developed to estimate the true labels for learning from multiple annotators of unknown quality. The algorithm takes into account that the annotators are not only unreliable, but may also be inconsistently accurate depending on the data. Given a dataset $D=\{x_i, y_i^1, \ldots, y_i^R\}$ (where x_i is an instance, $y_i^j \in \{0,1\}$ is the corresponding binary label assigned to the instance x_i by the j-th annotator and R is the number of the annotators), GMM-MAPML algorithm uses EM algorithm and Bayesian information criterion (BIC) to get parameters of the fittest Gaussian mixture model (GMM) and its mixture components' responsibilities (τ_{ik}) for each instance. Based on the intuition that real world annotators have different sensitivity and specificity for different groups of instances, the sensitivity α_k^j and specificity β_k^j are defined as:

$\alpha_k^j = \Pr(y_i^j = 1 \mid y_i = 1$, k-th Gaussian component generates $x_i)$

$\beta_k^j = \Pr(y_i^j = 0 \mid y_i = 0$, k-th Gaussian component generates $x_i)$

where $j=1,\ldots,R$; $k=1,\ldots,K$. Therefore, the algorithm models the annotators to generate labels as follows: given an instance x_i to label, the annotators find the Gaussian mixture component which most probably generates that instance. Then the annotators generate labels with their sensitivities and specificities at the most probable component.

By following the annotator model, GMM-MAPML uses majority voting to initialize the probabilistic labels z_i (i.e., the probability when the hidden true label is 1). Then, the algorithm alternately carries out the maximum-likelihood (ML) estimation and the maximum a posterior (MAP) estimation: given the current estimates of probabilistic labels z_i, the ML estimation measures annotators' performance (i.e., their sensitivity α and specificity β) at each mixture component and learns a classifier with parameter w; given the estimated sensitivity α, specificity β, and the prior probability which is provided by the learned classifier, the MAP estimation gets the updated probabilistic labels z_i based on the Bayesian rule. After the two estimations converge, the GMM-MAPML algorithm outputs both the probabilistic labels z_i and the model parameters $\phi = \{w, \alpha, \beta\}$. The GMM-MAPML estimations of the hidden true labels depend both on observations and on the labels from all annotators. A brief summary of the GMM-MAPML algorithm is shown at algorithm 1. For details please refer to [2].

Algorithm 1: GMM-MAPML Algorithm

1. Find the fittest K-mixture-component GMM for the instances, and get the corresponding GMM parameters and components responsibilities τ_{ik} for each instance.
2. Initialize $z_i = (1/R)\sum_{j=1}^{R} y_i^j$ based on majority voting.
3. (ML estimation) Given z_i, estimate the sensitivity and specificity of j-th annotator at k-th component as follows.

$$\alpha_k^j = \sum_{i=1}^{N} z_{ik} y_i^j \Big/ \sum_{i=1}^{N} z_{ik}$$

$$\beta_k^j = \sum_{i=1}^{N} (\tau_{ik} - z_{ik})(1 - y_i^j) \Big/ \sum_{i=1}^{N} (\tau_{ik} - z_{ik})$$

Also learn a logistic regression classifier by the Newton-Raphson update for optimizing w. Then we can calculate the prior probability p_i for the positive class as

$$p_i = \Pr[y_i = 1 \mid \boldsymbol{x}_i, \boldsymbol{w}] = \sigma(\boldsymbol{w}^T \boldsymbol{x}_i).$$

4. (MAP estimation) Given the sensitivity and specificity of each annotator at each component and the classifier parameter, update z_i as follows.

$$z_i = \frac{a_i p_i}{a_i p_i + b_i (1 - p_i)}$$

where

$$p_i = \Pr[y_i = 1 \mid \boldsymbol{x}_i, \boldsymbol{w}] = \sigma(\boldsymbol{w}^T \boldsymbol{x}_i)$$

$$a_i = \prod_{j=1}^{R} [\alpha_q^j]^{y_i^j} [1 - \alpha_q^j]^{1 - y_i^j}$$

$$b_i = \prod_{j=1}^{R} [1 - \beta_q^j]^{y_i^j} [\beta_q^j]^{1 - y_i^j}$$

$$q = \arg\max_{k=1,\ldots,K} (\tau_{ik})$$

Iterate steps 3 and 4 till convergence.

The GMM-MAPML algorithm can also be extended to multi-class data. Suppose that there are $C \geq 2$ classes. Let $y_i^j \in \{1,\ldots,C\}$ be the label assigned to the i-th instance by the j-th annotator, and let $y_i \in \{1,\ldots,C\}$ be the hidden true label. We model each annotator at the k-th component by the multinomial parameters $\boldsymbol{\alpha}_t^{j(k)} = (\alpha_{t1}^{j(k)},...,\alpha_{tC}^{j(k)})$, where

$$\alpha_{tc}^j = \Pr[y_i^j = c \mid y_i = t],\ \sum_{c=1}^{C} \alpha_{tc}^j = 1.$$

The term α_{tc}^j denotes that the probability that annotator j assigns class c to an instance given the true class is t. Then, algorithm 1 can be applied to the multi-class label settings.

B. The IMCC Algorithm

Given a small dataset of labeled examples $L = \{(x_i, y_i)\}$, where x_i is an instance and $y_i \in \{1,\ldots,C\}$ is the corresponding label of the instance x_i, and a sufficiently large set of unlabeled examples $U = \{u_i\}$, we want to estimate the unknown true labels of the instances in the given test set $T = \{t_i\}$.

The first step in our IMCC algorithm is to create a committee of m diverse co-training classifiers $\{CL\}_i^m$. This is achieved by running m iterations of co-training independently until they terminate for a given dataset. For each co-training run, a different random feature split is used. The random feature split is created by random selection of half of the features as the first view, and using the remaining half of the features as the second view.

In each run of the co-training algorithm, a different pair of base classifiers is trained (because each time a different feature split is used). Based on their confidence, each pair of base classifiers selects and labels diverse instances from the set of unlabeled instances. Thus, it may happen that in different runs of co-training different instances are selected for labeling, and it may also happen that different co-training classifiers label the same instance differently. Thus, by using different random splits we gain a set of diverse co-training classifiers.

Co-training is highly sensitive to the two underlying assumptions on views [5, 12], and therefore sensitive to the feature split division. Therefore, the classifiers in our committee produced by running co-training with different splits of features should vary in performance and make errors independently of each other, which enable us to treat them as independent annotators when applying the GMM-MAPML method.

Each of the m classifiers from the created committee predicts a label for each instance from the dataset T. In this way, we get a prediction set P consisting of m labels: $P = \{(t, y_t^1,\ldots, y_t^m)\}$, where y_t^k is the label assigned to the instance t by the k-th classifier. Finally, we employ the GMM-MAPML algorithm in order to estimate the true label of each instance in T.

The original co-training algorithm labels examples with respect to their membership in one of two classes: positive and negative. We can extend co-training to work with multi-class data by applying inner classifiers, which can handle multi-class data. Also, we allow each inner classifier to label a predefined number of most confident examples for each class in each iteration.

The IMCC algorithm is summarized at algorithm 2.

Algorithm 2: IMCC Algorithm

Given:

- A small set L of labeled training examples, described by feature set X;
- A much larger set U of unlabeled examples;
- A set T of instances for which we want to estimate the unknown true labels;
- Co-training parameters: The number of co-training runs m; The number of examples $c_1, c_2,\ldots,c_C$ of each class $y_i \in \{1,\ldots,C\}$ to be added in each iteration of co-training to the initial training set; The size of the small unlabeled pool u; The number of iterations for co-training algorithm k;

Training:

for *i*=1..*m* iterations:

- Create two feature sets (views) X_1 and X_2 describing the examples by randomly splitting feature set X in half.
- Create a pool U' of examples by choosing u examples at random from U.
- **Loop for *k* iterations:**
  - Use L to train a classifier h_1^i that considers only the X_1 portion of X.
  - Use L to train a classifier h_2^i that considers only the X_2 portion of X.
  - For each class $y_j \in \{1,\ldots,C\}$:
    - Allow h_1^i to label c_j examples from U' which are most confidently predicted as y_j.
    - Allow h_2^i to label c_j examples from U' which are most confidently predicted as y_j.
  - Add these self-labeled examples to L.
  - Randomly choose $2{\cdot}\Sigma_{j=1}^{C} c_j$ examples from U to replenish U'

  end loop

end for;

Classification of new examples:

- Form a prediction set P from instances from T ($P = T$).
- **for *i*=1..*m***
  - For each instance t from T, compute the probability of each possible label by multiplying the probabilities output by h_1^i and h_2^i for that label and assign the instance its most probable label y_t^i. Assign this label to the instance t in the prediction set P.

  end for;
- Apply GMM-MAPML algorithm to the prediction set $P = \{(t, y_t^1,\ldots, y_t^m)\}$ in order to estimate the true label y_t of each instance t. Assign y_t to instance t in the set T.

Output:

- Set $T = \{(t, y_t)\}$ of instances and their estimated labels.

IV. EXPERIMENTS

In this section we report the experimental procedures and the validation results of the proposed method versus alternatives obtained on 25 datasets.

A. Datasets and Configuration

We have evaluated our method on 3 natural language datasets used in previous studies to test the performance of co-training: the WebKB-Course dataset [1, 5], News2x2 [3] and LingSpam corpus [3, 7]. In addition, we performed experiments on 14 binary and 8 multi-class UCI datasets also previously used for evaluating co-training [4, 8, 13]. The benchmark datasets of various properties were selected to give us a better insight of how effective our method is on datasets of various dimensionality, size and redundancy. For natural language datasets we have adopted the preprocessing technique used in [3] in order to compare our methodology to the performance of co-training run with the MaxInd feature split. For the dimensionality reduction, document frequency was used in order to choose the 200 most important features for each view. After dimensionality reduction, the dataset is represented using the bag-of-words model with tf-idf measurement [14]. Both document frequency and tf-idf measurement are class-independent and therefore do not violate the co-training setting, where labels are available for just a few training examples.

For evaluation, we have adopted the stratified 10-fold-cross validation described in [3]. Data is divided into 10 stratified folds. In each round of the 10-fold-cross validation process, a different fold is selected for random selection of the required number of labeled training examples. The remaining data from that fold, as well as from 5 adjacent folds, are used as unlabeled training examples, and finally, the remaining 4 folds are used as testing examples. In this way, in each round, 60% of the data is used for training and the remaining 40% is used for testing. Each fold is used exactly once for the selection of labeled data, five times as unlabeled data and four times as a part of the testing set. In the experiment, we used accuracy as the measure of performance.

For co-training configuration, the number of examples for each class in the initial training set L was proportional to the class distribution in the dataset (except for the LingSpam dataset, where we use the same settings as in [3]). The numbers of examples per each class labeled by co-training inner classifiers at each iteration (i.e., the parameters $c_1, c_2, \ldots, c_C$ in algorithm 2) were also chosen proportionally to the class distribution and the dataset size. By following [3], the size of the unlabeled pool u was 50 and the number of iterations used in the co-training algorithm (in each of the settings) was 20. The number of co-training classifiers in the ensemble m, used by the IMCC algorithm, was 50 for the natural language datasets and 30 for the UCI datasets. Other than the $\text{MaxInd}_{\text{best}}$ discussed in the section IV.B, the base classifier used in the co-training algorithms was Naive Bayes. This classifier was chosen both for its speed and for the high accuracy it achieves on benchmark datasets.

The datasets and their basic properties are listed in Table I. In this table, the first three datasets (WebKB, LingSpam, and News2x2) are natural language datasets, rows 4-17 are binary UCI datasets (Hepatitis to Diabetes) and the remaining 8 are multi-class UCI datasets. All groups are sorted by the parameter *Gap*.

TABLE I. A SUMMARY OF THE DATASETS USED IN THE EXPERIMENT

Datasets	Dim	|L|	L_{acc}	|All|	All_{acc}	Gap
WebKB (2)	400	10	78.6	630	96.4	17.8
LingSpam (2)	400	10	80.1	1735	88.9	8.8
News2x2 (2)	400	10	81.1	1200	89.6	8.5
Hepatitis (2)	19	2	61.7	92	84.8	23.1
Kr-vs-kp (2)	36	11	65.6	1917	87.2	21.6
Credit-g (2)	20	2	53.6	600	74.1	20.5
Heart-statlog (2)	13	5	65.7	162	80.5	14.8
Cylinder-bands (2)	39	5	58.4	323	72.9	14.5
Sonar (2)	60	2	55.5	124	68.8	13.3
Ionosphere (2)	34	8	70.1	210	83.1	13.0
Breast-cancer (2)	9	2	59.0	171	71.7	12.7
Credit-a (2)	15	9	69.3	413	81.5	12.2
Tic-tac-toe (2)	9	9	58.8	574	70.7	11.9
Spambase (2)	57	3	67.7	2759	79.6	11.9
Breast-w (2)	9	3	86.3	418	97.4	11.1
Mushroom (2)	22	6	84.7	4873	95.3	10.6
Diabetes (2)	8	3	64.8	460	75.0	10.2
Splice (3)	62	24	66.2	1914	95.3	29.1
Wine (3)	14	3	71.9	107	96.2	24.3
OptDigits (10)	65	60	72.6	3372	91.7	19.1
SyntheticControl (6)	62	24	76.8	360	94.1	17.3
Waveform5000 (3)	41	15	64.1	3000	80.0	15.9
Dermatology (6)	35	19	81.8	219	97.1	15.3
Segment (7)	20	21	67.2	1386	80.8	13.6
CMC (3)	10	5	41.1	884	48.8	7.7

Notation - **Dataset:** the number in parentheses following the dataset names denotes the number of classes; **Dim:** the number of features describing the dataset; **|L|:** the size of the initial training set L; **L_{acc}:** accuracy achieved by a supervised Naive Bayes classifier trained on the initial set L; **|All|:** the size of the entire training set *All* (i.e., the sum of numbers of labeled and unlabeled examples); **All_{acc}:** accuracy achieved by supervised Naive Bayes classifier trained on the entire training set *All* (i.e., labeled examples and unlabeled examples with correct label); **Gap:** performance gap (also called the optimal gain in [4]) computed as $All_{acc} - L_{acc}$.

B. Experimental Results

The following co-training methods were considered in our experiments:

1. Co-training using a natural feature split (in datasets WebKB, News2x2, and LingSpam the natural feature split is known), hereinafter referred to as **Natural**.
2. Co-training using a random feature split, hereinafter referred to as **Random**.
3. The classifier obtained by a majority voting of the classifiers created in the same way as the committee of classifiers in IMCC, hereinafter referred to as **MV**.
4. Co-training using a MaxInd feature split introduced in [3]. In [3] it is reported that MaxInd is not always successful in combination with Naive Bayes while it might be more successful when using other base classifiers, such as RBF

Nets and SVM. Because of this, we report the best performance of MaxInd achieved when using one of these three classifiers for each dataset. Hereinafter we refer to such a co-training method as **MaxInd$_{best}$**.

5. The RSSalg method introduced in section II always selects the most reliable and most informative cases in the co-training process. Hereinafter we refer it as **RSSalg**.
6. The new **IMCC** method proposed in this paper.

In table II we report the accuracies obtained by each co-training method on all datasets. In most datasets (19 out of 25), our IMCC method was the most accurate.

To further characterize the proposed IMCC algorithm, we also conducted a Friedman test in order to detect whether there are differences considering the global set of classifiers, followed by post-hoc Bergmann-Hommel's test in order to find the concrete pairwise comparisons that produce the differences as recommended in [15, 16]. The significance level of α=0.05 is used for all tests. Using Friedman's test, we compared 5 co-training methods (Natural was not included in the tests as there are too few datasets on which we can measure its performance) as well as NB_L (i.e., Naive Bayes classifier trained on the initial set *L*) and NB_All (i.e., Naive Bayes classifier trained on the entire training set *All* by using labeled examples and unlabeled examples with correct labels) on all datasets (for lack of space, results are reported in a supplement found at http://astro.temple.edu/~tua87106/ICMLA_Suppl.pdf).

In our conducted experiments and statistical tests, the proposed method IMCC was more accurate than all alternative co-training methods and the difference was statistically significant versus all alternatives except for RSSalg. It should be noted that the reported accuracy of RSSalg is optimistic (it only represents the upper bound for that method) as its thresholds are tuned manually according to its performance on the test set. In co-training applications, we lack the labels of the test data needed for tuning of these thresholds. Thus, the results presented here for RSSalg only reflect its upper bound performance on this particular test set. Finally, IMCC resulted in a statistical tie with Naive Bayes classifier trained using a much larger set of labeled data *All.*

TABLE II. COMPARISONS OF IMCC VS. FIVE ALTERNATIVE CO-TRAINING METHODS. THE PERCENT ACCURACY AND STANDARD DEVIATION ARE REPORTED BASED ON 10-FOLD STRATIFIED CROSS-VALIDATION ON EACH DATASET. NATURAL IS APPLICABLE ONLY ON THE FIRST THREE DATASETS (WEBKB, LINGSPAM, AND NEWS2X2) THAT HAVE KNOWN NATURAL FEATURE SPLITS. THE HIGHEST ACCURACIES IN EACH DATASET ARE BOLDED.

Datasets	**Natural**	**Random**	**MV**	**MaxInd$_{best}$**	**RSSalg**	**IMCC**
WebKB	87.2±6.7	84.2±7.6	87.7±3.5	78.3±9.1	**90.7±3.3**	88.6±1.0
LingSpam	70.3±13.3	76.6±8.5	81.1±7.4	83.9±1.1	91.1±5.9	**95.3±0.5**
News2x2	82.9±4.6	80.0±7.0	86.3±2.4	76.2±12.8	**90.6±1.8**	85.7±0.7
Hepatitis		80.3±8.0	83.3±4.3	80.8±7.9	**86.5±3.4**	85.8±5.1
Kr-vs-kp		54.4±5.0	55.3±4.5	60.1±6.2	67.1±4.2	**79.1±1.0**
Credit-g		62.0±5.5	64.4±5.5	68.1±1.8	70.2±0.7	**70.7±1.8**
Heart-statlog		79.4±8.2	81.8±2.0	80.8±4.5	83.3±2.2	**85.2±2.6**
Cylinder-bands		52.5±5.2	52.9±6.5	56.3±5.7	61.6±2.5	**65.9±1.9**
Sonar		54.9±6.0	56.5±5.5	56.7±9.1	61.2±5.8	**62.4±3.8**
Ionosphere		69.4±12.6	73.1±4.9	78.3±7.7	**79.6±5.8**	75.2±3.0
Breast-cancer		66.7±6.1	68.2±4.5	67.5±5.4	70.4±5.3	**73.9±2.2**
Credit-a		69.2±15.0	73.4±11.0	76.1±2.6	77.6±4.8	**79.6±1.2**
Tic-tac-toe		61.5±3.2	63.2±2.5	62.0±1.7	64.1±2.9	**70.5±1.5**
Spambase		67.8±15.8	78.4±4.6	68.9±8.2	81.5±4.1	**81.5±0.5**
Breast-w		96.8±0.8	96.9±0.7	96.7±0.7	97.5±0.4	**97.6±0.4**
Mushroom		88.2±3.2	89.1±1.0	88.4±1.3	89.2±0.9	**89.9±0.7**
Diabetes		61.4±7.3	64.1±3.3	65.3±1.1	67.7±1.8	**71.7±2.1**
Splice		81.1±6.9	84.1±3.0	77.3±8.5	86.2±3.2	**93.1±0.3**
Wine		92.5±6.8	94.5±2.6	92.3±3.5	96.8±1.9	**97.8±0.9**
OptDigits		77.4±3.2	82.3±2.0	**88.3±1.7**	83.4±1.7	87.9±0.3
SyntheticControl		84.5±4.1	85.0±2.3	87.8±4.1	**87.9±2.8**	86.7±1.1
Waveform5000		63.0±7.6	67.1±6.8	64.0±4.4	72.0±6.1	**79.8±0.6**
Dermatology		86.9±4.3	87.1±4.2	83.6±2.2	87.6±3.9	**97.3±1.7**
Segment		59.6±4.7	63.2±4.1	62.6±3.9	72.6±2.9	**76.6±1.7**
CMC		37.3±3.5	38.2±3.1	38.6±3.0	45.0±3.3	**47.5±1.4**

V. CONCLUSION

In this paper we have proposed an IMCC algorithm designed to boost the performance of co-training and to enable its successful application on single-view datasets of varying size, dimensionality and redundancy. In each iteration of IMCC, a different random split of features is used for co-training to create a set of diverse classifiers. The predictions of those classifiers on the test data are treated as noisy labels and the GMM-MAPML method, designed for estimation of true labels in a multi-annotator problem setting, is applied to estimate the true labels.

The experiments were performed on 3 natural language datasets, as well on 22 UCI datasets, using a 10-fold-cross validation procedure. In all experiments, the proposed algorithm succeeded in improving the accuracy of the initial Naive Bayes classifier trained on the same small-labeled set, and it outperformed all five alternative co-training methods considered in our study. Also, IMCC resulted in a statistical tie with Naive Bayes classifier trained using a much larger set of labeled examples.

We are in the process of integrating our IMCC algorithm into the information system for monitoring the scientific research activity of the University of Novi Sad (CRIS UNS). The newly proposed algorithm is an additional support of the system for automatic extraction of metadata from scientific publications, with the goal to overcome the problem of manual annotation of a large number of scientific papers [17]. Finally, we are working on overcoming the bottleneck of manual annotation in the case of automatically mining methodologies from scientific articles [18].

ACKNOWLEDGEMENT

Results presented in this paper are part of the research conducted within the Grant No. III-47003 financed by the Ministry of Education and Science of the Republic of Serbia.

This project was also funded in part under a grant with the GlaxoSmithKline LLC.

REFERENCES

[1] A. Blum and T. Mitchell, “Combining labeled and unlabeled data with co-training,” Proc. 11th Annual Conf. Computational Learning Theory, 92-100, 1998.

[2] P. Zhang and Z. Obradovic, “Learning from Inconsistent and Unreliable Annotators by a Gaussian Mixture Model and Bayesian Information Criteria,” Proc. European Conf. Machine Learning and Principles and Practice of Knowledge Discovery in Databases, 553-568, 2011.

[3] F. Feger and I. Koprinska, “Co-training Using RBF Nets and Different Feature Splits,” Proc Int’l Joint Conf. Neural Network, 1878-1885, 2006.

[4] J. Du, C. Ling and Z.H. Zhou, “When Does Co-Training Work in Real Data?,” IEEE Trans. Knowledge and Data Engineering, 23(5), 788-799, 2010.

[5] K. Nigam and R. Ghani, “Understanding the behavior of co-training,” Proc. Workshop on Text Mining at KDD, 2000.

[6] K. Nigam and R. Ghani, "Analyzing the Effectiveness and Applicability of Co-training," Proc. Int’l Conf. on Information and Knowledge Management, 86-93, 2000.

[7] J. Chan, I. Koprinska and J. Poon, “Co-training with a single natural feature set applied to email classification,” Proc. IEEE/WIC/ACM Int’s Conf. Web Intelligence, 586-589, 2004.

[8] Z.H. Zhou and M. Li, “Tri-training: Exploiting unlabeled data using three classifiers,” IEEE Trans. Knowledge and Data Engineering, 17, 1529-1541, 2005.

[9] M.F. Hady and F. Schwenker, “Co-training by Committee: A New Semi-supervised Learning Framework,” Proc. IEEE Int’l Conf. Data Mining Workshops, 563-572, 2008.

[10] M. Li and Z.H. Zhou. “Improve computer-aided diagnosis with machine learning techniques using undiagnosed samples,” IEEE Trans. Systems, Man and Cybernetics, 37(6), 1088-1098, 2007.

[11] J. Slivka, A. Kovacevic and Z. Konjovic, “Co-Training Based Algorithm for Datasets without the Natural Feature Split,” Proc. IEEE Int’l Symp. Intelligent Systems and Informatics, 279-284, 2010.

[12] I. Muslea, S. Minton and C. Knoblock, “Active + Semi-Supervised Learning = Robust Multi-View Learning,” Proc. Int’l Conf. Machine Learning, 2002.

[13] J. Huang, J. Sayyad-Shirabad, S. Matwin and J. Su, “Improving Co-training with Agreement-Based Sampling,” Proc. Rough Sets and Current Trends in Computing, 197-206, 2010.

[14] G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” Information Processing and Management, 24(5), 513-523, 1988.

[15] J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” Journal of Machine Learning Research, 7:1-30, 2006.

[16] S. García and F. Herrera, ”An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons,” Journal of Machine Learning Research, 9:2677-2694, 2008.

[17] A. Kovačević, D. Ivanović, B. Milosavljević, Z. Konjović, D. Surla, "Automatic extraction of metadata from scientific publications for CRIS systems," Program: Electronic library and information systems, 45(4), 376-396, 2011.

[18] A. Kovačević, Z. Konjović, B. Milosavljević, G. Nenadic, "Mining methodologies from NLP publications: A case study in automatic terminology recognition," Computer Speech & Language, 26(2), 105-126, 2011.