Iterative classification for multiple target attributes

Guo, Hongyu; Létourneau, Sylvain

Publisher's version / Version de l'éditeur:
https://doi.org/10.1007/s10844-012-0224-5
Journal of Intelligent Information Systems, 40, 2, pp. 283-305, 2013-04-01

Iterative Classification for Multiple Target Attributes

Hongyu Guo · Sylvain Létourneau

Received: date / Accepted: date

Abstract Many real-world applications require the simultaneous prediction of multiple target attributes. The techniques currently available for these problems either employ a global model that simultaneously predicts all target attributes or rely on the aggregation of individual models, each predicting one target. This paper introduces a novel solution. Our approach employs an iterative classification strategy to exploit the relationships among multiple target attributes to achieve higher accuracy. The computation scheme is developed as a wrapper in which many standard single-target classification algorithms can be simply "plugged-in" to simultaneously predict multiple targets. An empirical evaluation using eight data sets shows that the proposed method outperforms 1) an approach that constructs independent classifiers for each target, 2) a multitask neural network method, and 3) ensembles of multi-objective decision trees in terms of simultaneously predicting all target attributes correctly.

Keywords Multi-target Learning · Multitask Learning · Iterative Classification · Data Mining

1 Introduction

In contrast to focusing on predicting a value of a single-target attribute (either single-label or multi-label classification tasks), many classification applications involve n ($n > 1$) target attributes (Zenko and Dzeroski, 2008) and require simultaneously determining the correct n target attributes for each instance. Such multi-target learning problems are common in many data mining applications. For instance, in medical applications one often needs to simultaneously settle on several symptoms in order to associate a patient with a certain disease (Lipkin et al, 1969; Gaag et al, 2001). Similarly, the models used to construct characters

Hongyu Guo and Sylvain Létourneau
National Research Council Canada
Address: 1200 Montreal Road, Ottawa, Ontario
E-mail: {hongyu.guo,sylvain.letourneau}@nrc-cnrc.gc.ca

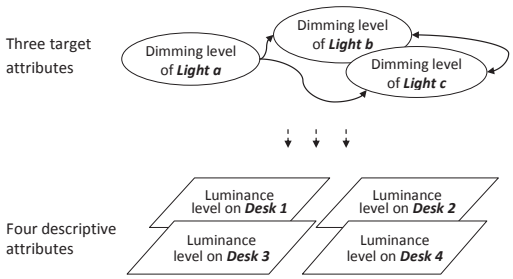

Fig. 1 An intelligent lighting system. The classification task here is to use the four descriptive features (namely the luminance levels on the four desks) to predict the three target attributes (i.e., the dimming levels of the three lights on the ceiling)

in computer-based animations have to determine several body characteristics such as shape, size, and pose (Xi et al, 2007).

The above multi-target problems can be very challenging for two reasons. First, the solution space, which is exponential in the number of target attributes, becomes enormous even with a limited number of target attributes. Second, the relationships between the target attributes can add a level of complexity that needs to be taken into account. To illustrate, let us consider an intelligent lighting system aiming to automate the control of lights in an open space office environment. We depict a simple case in Figure 1. The data mining task in this system is to use measured luminances on occupants' desks in a room to classify the correct dimming levels for all lights on the ceiling (the predicted dimming levels will be sent to the automation system to dim the lights). In this case, each light is a target attribute. The size of the solution space is given by m^n where n is the number of lights and m the number of distinct dimming levels. Finding an adequate solution within such a large space is going to be difficult even for limited numbers of lights and dimming levels. Importantly, the target attributes are also correlated since all lights contribute (at least to some degree) to the luminance observed across the room. Even an error on a single light could affect the luminance on all desks in the room and therefore result in dissatisfaction of all occupants.

A straightforward approach of learning and applying an independent classifier for each target attribute would be efficient. However, the overall accuracy is likely to suffer; because the classifiers are being developed independently, one is likely to observe a multiplicative effect of the errors from the individual classifiers, which would be detrimental even with a limited number of reasonably good classifiers. To guarantee success, the modeling needs to be able to account for the relationships among the targets. This paper proposes a new approach to take such relationships into account, thus improving the simultaneous prediction of multiple target attributes.

Our method was motivated by the following observations. Let us revisit the above-mentioned lighting example, as illustrated in Figure 1. In this example, knowing the dimming levels of *Lights b* and *c* does not benefit the predictions for *Light a*, but information about the dimming level of *Light a* helps improve the predictions of the dimming levels of *Lights b* and *c*. This observation suggests that the predictions for *Light a* are less affected by the target values of *Lights b* and *c*. In other words, in an iterative classification procedure, the predictions for *Light a*

will be insensitive to the estimated values for *Lights b* and *c*. Thus, the predictions for *Light a* can be *iteratively* used to help improve the estimates for *Lights b* and *c* through an iterative classification process.

Motivated by the above observations, our proposed approach first constructs independent classifiers for each target attribute. Next, it *iteratively* augments the classifiers against the "hard to learn" targets by expanding the initial instance's features with the predicted values of related targets. The computation scheme is developed as a wrapper in which many single-target classification algorithms can be simply "plugged-in" to simultaneously predict multiple target attributes. An empirical evaluation using eight data sets shows that the proposed method outperforms two benchmarking multi-task (-target) learning approaches, namely the multi-task neural networks (Caruana, 1997) and the ensembles of multi-objective decision trees (Kocev et al, 2007), as well as the approach that constructs independent single-target classifiers for each target attribute. Interestingly, our results also suggest that the iterative classification process improves the simultaneous prediction of the collection of independent classifiers by forcing these classifiers to make correct classifications on common instances instead of boosting their individual accuracies.

The paper is organized as follows. Section 2 introduces related work. Section 3 presents the proposed strategy in detail. Section 4 offers an experimental evaluation. Finally, Section 5 concludes the paper and outlines our future work.

2 Related Work

Related literature can be categorized into three groups: multi-task learning, iterative classification, and multi-label learning.

2.1 Multi-task Learning

Multitask learning (MTL) simultaneously learns multiple related tasks so that the tasks can benefit from each other (Caruana, 1997; Xue et al, 2007). Some theoretical work (Baxter, 2000; Ben-David and Schuller, 2003) have been conducted to help better understand the advantages of multitask learning. For instance, Baxter (2000) introduced the notion of the "extended VC dimension" to define the generalization bounds of the average error of the multiple tasks. Also, instead of focusing average error among tasks as considered by Baxter (2000), Ben-David and Schuller (2003) derive error bounds on each related task.

In addition, many traditional learning approaches have been extended to deal with multitask applications. A technique often used in such extensions is to introduce common parameters among the multiple tasks so that related information can be shared. For example, Caruana and Baxter allow multiple tasks to share hidden nodes when training a neural networks (Caruana, 1997; Baxter, 2000). Fang et al. introduce a set of relevance parameters to control the degree to which the data from other tasks are used in the expectation-maximization (EM) algorithm's parameters estimation (Fang et al, 2008). Also, some research adds a common prior in Bayesian modeling to capture the relations and similarities between the different tasks (Bakker and Heskes, 2003). Others allow Gaussian processes to share

parameters (Yu et al, 2005) or allow tasks to share a common structure on the predictor space (Ando and Zhang, 2005; Guo et al, 2011).

Continuing in the same trend, Evgeniou et al. extend existing kernel based learning methods for single task learning to tackle MTL problems (Evgeniou and Pontil, 2004). They assume that true models for the multiple tasks are all close to some existing ones. Also, van der Gaag and de Waal (2006) extends a Bayesian Network to model applications with more than one class. In their approach, the network structures over the features has bounded tree width. In addition, the set of feature variables connects with the set of class variables through a bi-partite directed graph. Following the same line of research, Suzuki et al (2001) extended a decision tree to simultaneously explain multiple labels; Last (2004) presented a Multi-objective Info-Fuzzy Network.

Predictive clustering (Blockeel et al, 1998) has also been introduced to address multitask problems. With this approach, examples are first clustered into subsets based on the target variables. Next, each cluster is associated with a predictive model using the descriptive attributes. These predictive models are then integrated and used to predict the values of the target variables. For example, Blockeel et al (1998) developed the Multi-objective decision trees (MODTs) approach for predicting examples with multiple target attributes at once. Their strategy extends an inductive tree learning algorithm. A decision tree is viewed as a hierarchy of clusters, each containing a subset of the instances. While constructing the trees, the sum of the entropies of the set of target variables is used to partition the examples for classification tasks. Following the same line of research but focusing on the well-known CN2 algorithm (Clark and Niblett, 1989), Zenko and Dzeroski (2008) proposed the Predictive Clustering Rules (PCRs) method for multitask problems. Inspired by the success of ensemble models, Kocev et al (2007) introduce the ensembles of multi-objective decision trees.

Instead of extending traditional learning algorithms, this paper attempts to establish a general framework for simultaneously predicting multiple target attributes. As a result, many state-of-the-art single-target learning techniques such as Decision Trees (Quinlan, 1993) and Neural Networks (Bishop, 1996) can be directly integrated into the computational scheme and applied to multi-target applications.

2.2 Iterative Classification

This research is also inspired by the iterative classification and inference strategies proposed for structured (relational) data (Chakrabarti et al, 1998; Lu and Getoor, 2003; Neville and Jensen, 2000; Oh et al, 2000; Taskar et al, 2001). In structured data, objects are assumed to be interconnected, such that, updating the category of one object can influence the inference about the classes of its related objects. For example, Chakrabarti et al (1998) employed an iterative approach to improve the predictive performance of a probabilistic model for hypertext categorization. Their method uses both local text in a document and the distribution of the estimated classes of neighboring documents to iteratively boost the classification accuracy. Similarly, Oh et al (2000) proposed a method for classifying a collection of encyclopedia articles. Their approach uses both links and incrementally available class information through an iterative labeling process. In recent years, iterative

inference has also been employed in collective classification algorithms to exploit the correlation among a set of related instances in relational data (Lu and Getoor, 2003; Neville and Jensen, 2000; Taskar et al, 2001; Macskassy and Provost, 2003). In these applications, the iterative classification algorithm is used to update the class labels or class conditional probabilities of some unseen instances, which are subsequently employed to derive information to help estimate the labels of other related instances.

This paper also employs an iterative classification technique, but our iterative method aims to exploit the interplays between multiple target attributes of the same instance, rather than exploring relationships among multiple related single-target instances, as is the case with relational data. Consequently, a novel ordering strategy for the iterative process has been devised. In addition, the iterative approaches for relational data assume that nodes with unknown labels are linked to some nodes with known labels in a graph. The prior knowledge about these known labels is used as a starting point for the iterative inference procedure to infer the unknown label nodes. In our proposed method, the values of all target attributes for an instance are unknown. We use target attributes that are insensitive to other targets to initiate the iterative process.

2.3 Multi-label Learning

Another line of research that relates to our proposed strategy is the multi-label learning. Multi-label learning is referred to as learning problems where each object is assigned a subset of one pre-defined label set (Tsoumakas and Katakis, 2007; Ueda and Saito, 2003). Two major categories of approaches have been proposed: problem transformation and algorithm adaptation.

The first family of methods convert the presented task into multiple single-label classification problems, so that conventional single label learning algorithms can be applied. Many data transformation methods have been proposed. For example, the Label Powerset (LP) method regards each unique set of labels in the provided multi-label training data as a single-label binary classification task (Tsoumakas et al, 2009). To deal with challenge of generating a very large number of single-label classification task, the PPT method (Read, 2008) extends the LP strategy by pruning label sets that occur less frequent. Following the same line of though, the RAkEL method (Tsoumakas and Vlahavas, 2007) trains each LP binary classifier with different subset of the set of labels; other methods, such as the RPC (Hüllermeier et al, 2008), CLR (Fürnkranz et al, 2008), and INSDIF (Zhang and Zhou, 2007b) algorithms also tackle multi-label problems through transforming the provided data set. After the data transformation, a commonly used method for multi-label problems is to construct a binary classifier for each class independently. Subsequently, a test object is classified into the classes for which the corresponding classifier says "yes" or that rank above a threshold (Tsoumakas and Katakis, 2007). Some other methods leverage the relationship between multiple labels in the same label set. For example, approaches to find a low-dimensional subspace shared among multiple labels have been studied (Ji et al, 2008; Yang, 2001).

The second important category of multi-label strategies extends exist traditional single-label learning methods through taking multiple labels associated

with each instance into account. For example, Zhang and Zhou (2006) propose the Multi-label Neural Networks for multi-label learning. Through employing a novel error function to a back-propagation neural network, their approach aims to rank labels belonging to an instance higher than that of no associating with the instance. Elisseeff and Weston (2001) extends an SVM algorithm through minimizing the ranking loss when dealing with multi-label data. Ghamrawi and McCallum (2005) adapts a conditional random field classification model for multi-label learning. Other algorithms such as the MMP (Crammer and Singer, 2003), ML-KNN (Zhang and Zhou, 2007a), and MMAC (Thabtah et al, 2004) methods also fall in this family.

Using estimated labels to augment the feature space has also been studied in the multi-label literature. For instance, to tackle multi-label text classification problems, Godbole and Sarawagi (2004) introduce a two-layer stacked method. In the first layer, classifiers are trained using text tokens only. In the meta layer, classifiers are trained using both the original features along with the classification outputs from the first layer. Continuing in the same trend, Read et al (2009) present a classifier chains strategy for multi-label classification. In their approach, features are augmented by prior binary relevance predictions in the classifier chain. Unlike the method proposed by Godbole and Sarawagi, where predicted labels from all other classifiers are available, the classifier chains strategy allows a classifier take into account predicted labels generated only by previous classifiers in the chain. Hence, the number of features for each classifier in the chain is different.

In contrast to using predicted labels that will not be updated after generated, as is the case with the two above-mentioned multi-label approaches, our strategy keeps updating the estimated labels and augmenting the testing process with the refined label values through an iterative inference component. Specifically, in each iteration, the current target attribute estimates, resulting from the previous iteration, are used to enhance the learning models. In other words, in our approach, the predictions made by other classifiers are dynamically modified and refined in each iteration of the iterative inference process.

In general, multi-label classification copes with data being associated with only one label set. On the contrary, this paper deals with data with multiple label sets, where each instance associates with one label from each of the set of labels.

3 Iterative Inference for Multi-target Classification

3.1 Problem Definition And Evaluation Metrics

The objective of a single-label classification task is to learn a hypothesis function $F(x)$, which maps each instance x to a single-label y from a disjoint label set Y ($y \in Y$). In a multi-label learning problem, each instance belongs to a subset of labels L from the pre-defined label set $Y (L \subset Y)$, where $|L|$ is unknown beforehand. In contrast to the above single-target problems, this paper considers classification problems with multiple target attributes. That is, each such instance is assigned to n target attributes $\mathbf{y} = \{y_1, y_2, \cdots, y_n\}$ (with $n > 1$), and the objective is to correctly determine all of the n target attributes simultaneously.

In such applications, correctly classifying all target variables of an instance is required. We, therefore, consider a classifier has classified an instance correctly

only if all target variables of that instance are correctly determined (i.e., "exact match"). Thus, the overall accuracy in this paper refers to the "exact match" accuracy. The task aims to achieve higher "exact match" accuracy through learning a function F that maps each instance to a vector:

$$\mathbf{Y} = F(X)$$

Algorithm 1 The IAMC Algorithm

Input: Train data (D_{tr}), test data (D_{te}), the maximum number of iterations Γ, and a single target learning method f.
Output: Predictions of all target attributes for a test instance, using $\mathbf{Y} = F(X)$.

procedure TrainingPhase(D_{tr})
  IntrinsicLearner(D_{tr})
  RelationalLearner(D_{tr})
  Generate descending order of the target attributes (noted O), based on the prediction improvement obtained from the previous two steps, i.e., $\epsilon^r(i) - \epsilon^s(i)$
end procedure

procedure TestingPhase(D_{te}, Γ, O)
  for each instance t in the test set **do**
    obtain y_i using f_i^s;update Y_i in the test set
  end for
  repeat
    for each instance t in the test set **do**
      for each $Y_i \in O$ **do**
        compute y_i using f_i^r; update Y_i in the test set
      end for
    end for
  until reach Γ or predicted values for all target attributes have stabilized
end procedure

. . . . . .

procedure IntrinsicLearner(D_{tr})
  for each $Y_i \in Y$ **do**
    Build a model f_i^s using X only and instances from the train set; obtaining predictive accuracy $\epsilon^s(i)$
  end for
end procedure

procedure RelationalLearner(D_{tr})
  for each $Y_i \in Y$ **do**
    Build a model f_i^r using $X \cup (Y \backslash Y_i)$ and instances from the train set; obtaining predictive accuracy $\epsilon^r(i)$
  end for
end procedure

3.2 The Proposed Algorithm

The proposed Iterative Approach for Multi-target Classification problems (IAMC) includes two phases: training and inference. The training stage constructs two

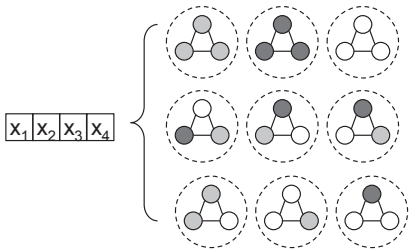

Fig. 2 Inputs (x_1, x_2, x_3, x_4) and some of the possible labels (dot-circles) for an intelligent lighting system.

collections of single-target classifiers, while the inference stage aims at exploiting the relationships among target attributes through these constructed classifiers.

The IAMC method firstly constructs two collections of classifiers: one utilizes the descriptive attributes only while the other is augmented with provided target attributes in the training data. Next, these two collections of classifiers are used for iterative inference, as follows. The first collection is used to initiate the iterative process, where all values of the target attributes in the test data set are unknown. The second one is then deployed to continue the inference procedure until the process stops. In each iteration, the current target attribute estimates, resulting from the previous iteration, are used to enhance the learning models. The details of the IAMC approach are discussed next.

3.2.1 Training Phase

At this stage, the IAMC method constructs two collections of single-target classifiers: intrinsic classifiers and relational classifiers. We will discuss these learning processes next.

Intrinsic Learners

The first collection of classifiers are trained using only the descriptive attributes, the so-called intrinsic features. Each classifier in this collection is constructed for one target attribute. That is, we build a set of functions

$$Y_i = f_i^s(X)$$

where i stands for the i^{th} target attribute. This collection of classifiers are used to infer the values of the target attributes of an instance when no class values about these target attributes are provided, which is the case in unseen data (i.e., the test data).

Example 1 To help describe the algorithm, let us consider the intelligent lighting application illustrated in Figure 1. Recall that there are three (3) lights and four (4) desks (occupants) in the office. The inputs to the intelligent lighting system are the measured luminances on occupants' desks (noted x_1, x_2, x_3, x_4), while the outputs correspond to the dimming levels of the light sources (noted y_1, y_2, y_3). For

simplicity, we assume that each light has the same three dimming levels. Figure 2 depicts the application. The nine (9) circles on the right side of Figure 2 show 9 of the 27 possible outputs. Within each circle, we denote three dots, each representing a light. The color of each dot indicates the dimming level for the corresponding light. Considering this simple scenario, this stage of the process will build three single-target classifiers, one for each light source. Also, we note that each classifier f_i^s is constructed using only the input attributes x_1, x_2, x_3, x_4. We denote these classifiers as *intrinsic learners*.

Relational Learners

The second collection of classifiers also includes a classifier for each target attribute. However, each of these classifiers is built using information from both the descriptive attributes and the related target attributes. Precisely, we have

$$Y_i = f_i^r(X, Y \backslash \{Y_i\})$$

Accordingly, these classifiers are not only able to exploit the relations between the input features and the target attributes to be determined but also the information among related target attributes. Since the classifiers here leverage the information among the related target attributes, i.e., the so-called relational attributes (Getoor and Taskar, 2007), we denote these classifiers as relational learners to indicate their relational learning context.

Example 2 Continuing with the example shown in Figure 2, at this stage we would build three classifiers, as follows: using inputs $\{x_1, x_2, x_3, x_4, y_2, y_3\}$ for f_1^r, using inputs $\{x_1, x_2, x_3, x_4, y_1, y_3\}$ for f_2^r, and using inputs $\{x_1, x_2, x_3, x_4, y_1, y_2\}$ for f_3^r.

In summary, the training phase generates two collections of single-target classifiers: one using descriptive attributes only and another being augmented by related target attributes. Each classifier in the collections focuses on one of the target attributes. After training, the IAMC algorithm starts the iterative inference process, as will be discussed next.

3.2.2 Testing Phase

This stage includes an iterative inference process and an ordering strategy. We will discuss these two components next.

Iterative Inference

The inference classification process aims to exploit the relationships among the multiple target attributes through the trained relational classifiers discussed above. Recall that the relational classifiers take into account not only the descriptive attributes of a test instance, but also the information from its target attributes. Obviously, in the real world, the new observations come without the true class values for the target attributes. However, these class values can be approximated

by the IAMC algorithm at the inference time through an iterative classification process, as follows.

In the first iteration, class values for all target attributes of the test data are obtained using the trained intrinsic learners. This is due to the fact that, at this stage, the only available information for classification is the descriptive attributes. In other words, the IAMC approach applies the collections of classifiers f_i^s to predict the i^{th} target attribute for the test instances.

Example 3 For the example shown in Figure 2, this iteration uses f_1^s, f_2^s, and f_3^s to independently predict the values for y_1, y_2, y_3, respectively. For illustrative purposes, we denote the resulting values for y_1, y_2, y_3 as y_1^1, y_2^1, y_3^1 respectively.

As a result, the intrinsic learners provide the currently estimated values for each target attribute of the testing instance, which in turn enables further inferences by the relational learners, which is discussed next.

After obtaining the initial estimated values for the target attributes of an instance, the trained relational classifiers are able to further infer the multiple target attributes of the instance, utilizing both the descriptive attributes and the currently estimated target attributes. That is, learner f_i^r will be applied to determine the i^{th} target attribute for each test instance. Following the second iteration, the relational learners iteratively infer all target attributes for an instance using the currently estimated values of the target attributes obtained in the previous iteration. By utilizing the best estimated target attributes currently available and leveraging the interdependence between those target attributes, the relational classifiers aim to augment their predictive performance iteratively.

Example 4 Running a second iteration through our example, we notice that the IAMC method uses x_1, x_2, x_3, x_4 and the estimated target attribute values resulting from the 1^{st} iteration, namely y_2^1, y_3^1, to determine the value for target attribute y_1. Let us denote the resulting value as y_1^2. In order to classify y_2^2, the values x_1, x_2, x_3, x_4 and the estimated target values resulting from the previous two(2) iterations (i.e., y_1^2, y_3^1) are used. Finally, to determine the value for target y_3, the IAMC algorithm uses the target values obtained from the previous iterations, namely y_1^2, y_2^2, along with the descriptive features x_1, x_2, x_3, x_4. This procedure is depicted in the middle of Figure 3, where the color dots indicate the estimated labels in the iterative inference process.

The above iterative process repeats until values for all target attributes have stabilized or a pre-set number of iterations has been reached. As stated in Algorithm 1, the IAMC outputs the estimated target values of the last iteration.

Ordering Strategy

In the above iterative process, which target attribute should be estimated first when processing a test instance? Clearly, this choice could have a significant impact on the performance of the IAMC strategy since the results of the previous iteration are used in the step immediately following it. That is, incorrect information from misclassification on a target attribute could directly propagate to related targets, leading to incorrect predictions on other target attributes.

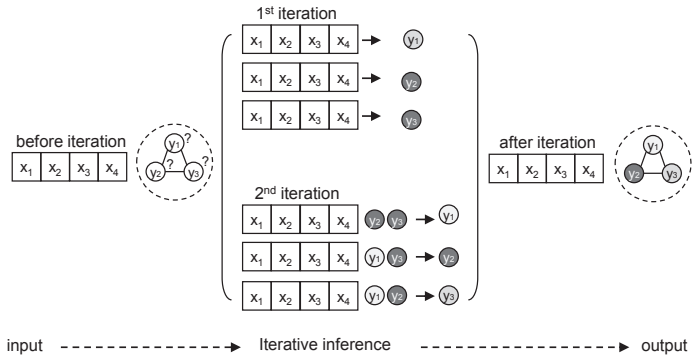

Fig. 3 An example of iterative inference in the IAMC approach.

The IAMC strategy orders the target attributes based on the increase in accuracy between the corresponding *intrinsic* and *relational* learners. That is, the IAMC approach first computes the accuracy improvements between f_i^r and f_i^s for each target attribute i. Next, it orders these values in a descending order and then proceeds from the beginning of the list. In this way, the target attribute for which maximal improvement has been obtained is processed first. In other words, through using target attributes that are less sensitive to other targets, the iterative process first attempts to improve the target attribute whose accuracy is much lower without than with the knowledge of other target attributes.

The rationale for the IAMC method's ordering strategy is as follows. Firstly, by using targets that are less sensitive to other targets to start the inference process, the IAMC strategy aims to use inferences made with high confidence to infer subsequent inferences of related target attributes. When a target attribute is less sensitive to related target attributes, one may consider that the predicted values made to this target are more reliable in an iterative classification process, because these predictions are less affected by the uncertainties introduced by the estimated values of other target attributes. The IAMC algorithm aims to *repeatedly* use these confident predictions to infer other related targets.

Secondly, to first infer targets that benefit most from the knowledge of related targets helps reduce the error propagation of the inference procedure. The IAMC strategy intends to improve the target attribute with worse accuracy first so that the improved predictions then can provide more accurate information for inferring other targets. In addition, a model that does not improve much by the addition of the extra knowledge (about its related target attributes) can be seen as a strong model that is less likely to be influenced by potential noise that may be introduced by the inference procedure. By keeping strong models at the end of the iterative process, the IAMC's *cautious* ordering strategy tends to minimize error propagations.

4 Experimental Study

4.1 Methodology

We implemented the IAMC algorithm using Weka (Witten and Frank, 2000). In our experiments, two single-target learning methods–J48 decision trees (Witten and Frank, 2000) and Artificial Neural Networks (Bishop, 1996)–were used in order to evaluate the impact of different single-target learning approaches on the IAMC framework. The decision tree learner was used due to its de facto standard for empirical comparisons. In addition, Artificial Neural Networks were chosen because they have been proved surprisingly successful in many real-world knowledge discovery applications (Mitchell, 1996).

We compared the overall performance of the IAMC method with three benchmarking algorithms developed for multitask and multi-target problems. The Multitask Neural Networks (MANNs) (Caruana, 1997) is probably the most influential work in this area. The algorithm trains a multilayer perceptrons (MLP) neural network with multiple output nodes, each focusing on learning a different target. The multiple tasks share inter-task information through a shared hidden layer trained in parallel on all the tasks.

The second tested method is a state-of-the-art multi-target classification system, namely the ensembles of Multi-Objective Decision Trees (Kocev et al, 2007). Kocev et al (2007) study the performance of the popular Multi-Objective Decision Trees (MODTs) algorithm and conclude that ensembles of MODTs improve the predictive accuracy of the MODTs method. We obtained the settings of the ensembles of MODTs from their authors. That is, in our experiments, a random forest strategy was applied to combine 100 individual multi-objective decision trees.

In addition, we compared the IAMC strategy with an algorithm that trains independent classifiers for each target attribute and then combines the results. That is, this approach treats the target attributes for each instance as mutually independent. For description purpose, we note this method as ExcLa.

All the results discussed were obtained through 10-fold cross validation with the default settings of Weka on a 2.93GHz PC with 64 bit Windows Vista installed. In addition, the maximal number of iterations for the inference process of the IAMC approach was set to 20 in all experiments.

4.2 Data Sets

In order to empirically evaluate the performance of the IAMC algorithm, we used eight data sets from various application domains: a multi-target application data set, six publically available data sets, which can be regarded as multiple target problems, and a synthetic data set where target attributes have different numbers of class values. The details of the data sets are as follows.

The Light dataset has been collected to experiment with the development of an intelligent lighting system as described in introduction. This application data used here is composed of six descriptive attributes corresponding to the luminances on six desks and four target attributes corresponding to four light sources in an open-space office. The learning task (denoted as “Light”) is to predict the dimming

Table 1 Descriptions of Data Sets Used

Data Set	Number of instances	Number of attributes	Number of target attr.	Number of classes in each target attr.	Domain
Yeast	2417	103	14	2	biology
Emotions	593	72	6	2	audio
Scene	2407	294	6	2	image
Mediamill	43907	120	10	10 or 11	video
Monks	432	6	3	2	robot
Caesar	2064	2	2	2	anthropometry
Light	19683	6	4	3	lighting
Opt-digits	5620	64	10	2	handwritten

levels (low, median, or high) for all four target attributes, given the luminance values on the six desks.

The "Emotions" data (Wieczorkowska et al, 2006) consists of six target attributes, 72 descriptive attributes, and 593 instances. The classes indicate whether a piece of music is associated with a certain emotion group. The "Scene" data describes a problem of semantic scene classification (Boutell et al, 2004). In this application, each still image may be associated with multiple target attributes. The data set consists of 2407 images, each described by 294 attributes and associated with six target attributes. The "Yeast" data set relates to protein classification (Elisseeff and Weston, 2001). The data consists of 103 attributes and 14 target attributes. The Monks problems are based on an artificial robots domain (Thrun et al, 1991). Each example is a robot described with six attributes. Each of the three monks' tasks is a binary classification task with three target attributes. The Opt-digits data is from the application of optical recognition of handwritten digits (Alimoglu and Alpaydin, 1997). The dataset consists of 10 tasks: the two class classification problems for hand-written digits. The CAESAR [1] data set was created by an international anthropometric project. In our experiment, we used all of the residents from Italy and the Netherlands, forming a total of 2,064 subjects. The learning task is to use a subject's stature and weight to simultaneously predict the subject's nationality and gender.

In order to obtain a data set where target attributes have different numbers of classes, we create a new data set from the Mediamill data (Snoek et al, 2006), as follows. The original Mediamill data describe a challenge that aims to label 85 hours of video data with 101 labels, each representing a binary classification problem. We consider 10 label intervals as one multi-target attribute, and its value is the number of classes the instance is associated with within the interval. The transformed data set therefore consists of nine target attributes where each has 10 class values and one target attribute that contains 11 class values.

Table 1 summarizes these eight data sets. For each data set, it lists the number of instances, descriptive attributes, target attributes, and the number of classes in each target attribute, along with the application domain of the data set.

4.3 Experimental Results

In this section, we evaluate the performance of the four tested algorithms, namely the IAMC, ExcLa, MANNs, and ensembles of MODTs (we denote this algorithm

[1] http://store.sae.org/caesar/

Table 2 "Exact match" accuracy obtained by the IAMC, ExcLa, MANNs, and En-MODTs methods. "†" and "−" respectively denote a statistically significant improvement or degradation of the IAMC algorithm when compared to the ExcLa, MANNs, and En-MODTs strategies, with a paired t-test with significance level of 0.05. The best accuracy achieved for each data sets is highlighted in bold. DTrees and ANNs respectively denote the J48 Decision Trees method and the Neural Networks approach as the single-target learning algorithm of the IAMC and ExcLa methods.

Data Set Used	Learning Method	IAMC(%)	ExcLa(%)	Ensembles of MODTs(%)	MANNs(%)
Yeast	DTrees	**13.73**	6.827 †	12.40 †	13.36
	ANNs	**18.53**	11.33 †	12.40 †	13.36 †
Emotions	DTrees	**25.62**	18.38 †	24.44	24.96
	ANNs	**28.17**	24.28 †	24.44	24.96
Scene	DTrees	**55.71**	42.66 †	55.29	55.46
	ANNs	**65.32**	54.66 †	55.29 †	55.46 †
Mediamill	DTrees	**12.55**	8.627†	10.47†	10.48†
	ANNs	**14.25**	12.13†	10.47†	10.48†
Monks	DTrees	**70.56**	63.64	57.83 †	64.56
	ANNs	**100.0**	**100.0**	57.83 †	64.56 †
Caesar	DTrees	**53.77**	52.17	52.76	52.23
	ANNs	**55.76**	54.74	52.76†	52.23 †
Light	DTrees	**74.19**	66.29 †	54.32 †	70.08 †
	ANNs	**100.0**	77.36 †	54.32 †	70.08 †
Opt-digits	DTrees	89.80	85.96 †	87.15 †	**95.96** −
	ANNs	**97.79**	96.60 †	87.15 †	95.96 †

as En-MODTs in this paper) methods, in terms of accuracy and execution time. We also empirically study the IAMC method's convergence properties.

4.3.1 " Exact Match" Accuracy Obtained

We present the overall predictive accuracy over all target attributes, namely the "exact match" accuracy, obtained by the four tested methods for each of the eight learning tasks, in Table 2. In this table, DTrees and ANNs respectively denote the J48 Decision Trees method and the Neural Networks approach as the single-target learning algorithm of the IAMC and ExcLa methods. Also, the signs "†" and "−" respectively denote a statistically significant improvement or degradation of the IAMC algorithm, when compared to the ExcLa, MANNs, and En-MODTs strategies, with a paired t-test with a significance level of 0.05. The best accuracy achieved for each data sets is highlighted in bold.

The predictive performance results obtained show that the IAMC algorithm appears to consistently improve the accuracy in most of the test cases when compared with the ExcLa, En-MODTs, and MANNs approaches. Statistically significant performance improvements, as indicated by the "†" sign in Table 2, were observed regardless of the single-target learning methods used.

When compared with the ExcLa method, the IAMC algorithm appears to consistently improve the accuracy for all of the 16 test cases. Also, in 12 of the 16 cases, the prediction improvements were statistically significant with 95% confidence. In particular, in many cases, we observed a very large accuracy improvement. For examples, the IAMC strategies improved the accuracy by 22.64%, 10.66%, and 7.20% against the Light, Scene, and Yeast data sets respectively when ANNs methods were applied. In addition, the results also show that large improvements were achieved when decision trees were deployed as single-target learners. For example, against the Scene, Light, Emotions, Yeast, and Opt-digits data sets, we notice accuracy improvements of 13.05%, 7.90%, 7.24%, 6.9%, and 3.84%, respectively.

When considering the comparison with the En-MODTs method, we notice that the IAMC algorithm was also able to increase the accuracy for all of the 16 cases, regardless of the single-target learning method for the IAMC approach. Amongst the 16 tested cases, 12 of them were statistically significant. In addition, the results also show that large accuracy improvements were achieved by the IAMC method when compared with the En-MODTs algorithm. For example, against the Monks, Opt-digits , Scene, and Mediamill data sets, the larger (of the J48 and ANNs as single-target learners) error rate reductions were 42.17%, 10.64%, 10.03%, and 3.78%, respectively.

Promising outcomes have also been observed when comparing the IAMC method with the MANNs approach. For example, as shown in Table 2, the IAMC method improved the accuracy in 15 of the 16 tested cases. The only exception was against the Opt-digits data set with J48 as single-target learners. In particular, when compared with the MANNs methods, large accuracy improvements have been achieved by the IAMC methods. For example, for the Monks, Light, Scene, Yeast, Caesar, and Emotions data sets, accuracy improvements obtained by the IAMC method were 35.44%, 29.92%, 9.86%, 5.17%, 3.53%, and 3.21% respectively.

In addition to the paired t-test, we also conducted the Friedman (1937, 1940) test with the corresponding post-hoc tests. These tests aim to further validate the statistical differences of the four tested approaches. Friedman tests enable the evaluation of statistical differences of multiple classifiers when considering their performances on multiple data sets as a whole (Demšar, 2006). In our experiments, each method's predictive accuracy was obtained by averaging the accuracies of the 10 folds. Also, the Wilcoxon-Nemenyi-McDonald-Thompson test (Hollander and Wolfe, 1999) was employed as the post-hoc tests. The p-values of the Friedman test as well as the corresponding post-hoc tests on the eight learning tasks are provided in Tables 3 and 4. Table 3 depicts the Friedman test results of the IAMC_DTrees, En_MODTs, ExcLa_DTrees, and MANNs methods, while Table 4 presents the results of the IAMC_ANNs, En_MODTs, ExcLa_ANNs, and MANNs strategies. Here, _DTrees and _ANNs respectively denote the J48 Decision Trees method and the Neural Networks approach as the single-target learning method of the IAMC and ExcLa algorithms.

Table 3 The resulting p-values of the Friedman test with the corresponding post-hoc tests (i.e., the Wilcoxon-Nemenyi-McDonald-Thompson test), when comparing the IAMC_DTrees, En_MODTs, ExcLa_DTrees, and MANNs methods on all the eight data sets.

Friedman test	p-value = 0.0002			
post-hoc tests		En_MODTs	ExcLa_DTrees	MANNs
	IAMC_DTrees	**0.0100**	**0.0003**	**0.5274**
	En_MODTs		0.7675	0.3014
	ExcLa_DTrees			0.0339

When comparing the IAMC_DTrees method with the other three tested approaches, namely the En_MODTs, ExcLa_DTrees, and MANNs algorithms, the results shown in the first row of Table 3 indicated that the p-value of the Friedman test was 0.0002. This number implied a statistically significant difference between at least two of the four tested methods. Our subsequent post-hoc tests results, depicted in Table 3, show that the pairwise p-values between the IAMC_DTrees

and the other three tested methods, namely the En_MODTs, ExcLa_DTrees, and MANNs, were 0.0100, 0.0003, and 0.5274, respectively (highlighted in bold in Table 3). These results suggest that, when considering the predictive performance on all the eight data sets, the IAMC strategy with decision trees as base learner significantly outperformed the other tested methods with a significance level of 0.05, except for the MANNs approach.

Table 4 The resulting p-values of the Friedman test with the corresponding post-hoc tests (i.e., the Wilcoxon-Nemenyi-McDonald-Thompson test), when comparing the IAMC_ANNs, En_MODTs, ExcLa_ANNs, and MANNs methods on all the eight data sets.

Friedman test	p-value = 0.0015			
post-hoc tests		En_MODTs	ExcLa_ANNs	MANNs
	IAMC_ANNs	**0.0007**	**0.0550**	**0.0427**
	En_MODTs		0.5842	0.6463
	ExcLa_ANNs			0.9996

When considering the comparison of the IAMC_ANNs, En_MODTs, ExcLa_ANNs, and MANNs algorithms on all the eight data sets (as a whole), the Friedman test resulted in a p-value of 0.0015, as shown in Table 4. This low p-value implies that at least one of the classifier significantly differs from at least one other classifier (Siegel and Castellan, 1988). Further post-hoc test results, as depicted in Table 4, confirmed that the IAMC_ANNs method statistically outperformed the En_MODTs and MANNs algorithms with a significance level of 0.05, and was borderline significant when against the ExcLa_ANNs approach. As shown in Table 4, the resulting pairwise p-values between the IAMC_ANNs and the En_MODTs, ExcLa_ANNs, and MANNs algorithms were 0.0007, 0.0550, and 0.0427, respectively (highlighted in bold in Table 4).

These results imply that the IAMC strategy can meaningfully improve the simultaneous prediction accuracy on multi-target problems when compared with the multitask neural network, the ensembles of multi-objective decision trees, and the approach that constructs independent classifiers for each target attribute.

4.3.2 Convergence Property

To examine the convergence properties of the inference process of the IAMC strategy, we display in Figure 4 the "exact match" accuracy obtained over the first 10 iterations for the 8 datasets: each depicted in a sub-figure. In this figure, $-\square-$ and $-\diamond-$ denote the IAMC approaches with Neural Networks and Decision Trees as single-target learning methods, respectively.

The graphs presented in Figure 4 show that the accuracy of the IAMC algorithm steadily increased and then stabilized in just a few iterations for all of the eight data sets. For example, for the Yeast data set, the IAMC method with ANNs applied needed only three iterations to achieve the best accuracy improvement and stability, as depicted in the top-left subfigure of Figure 4. With the ANNs as the single-target classifier, we notice impressive improvements over the first few iterations: from 11.33% to 17.74% and then to 18.53%. Similarly, against the same data set, the predictive accuracy of the IAMC approach with decision trees applied improved from 6.82% to 13.61% during the first two iterations and

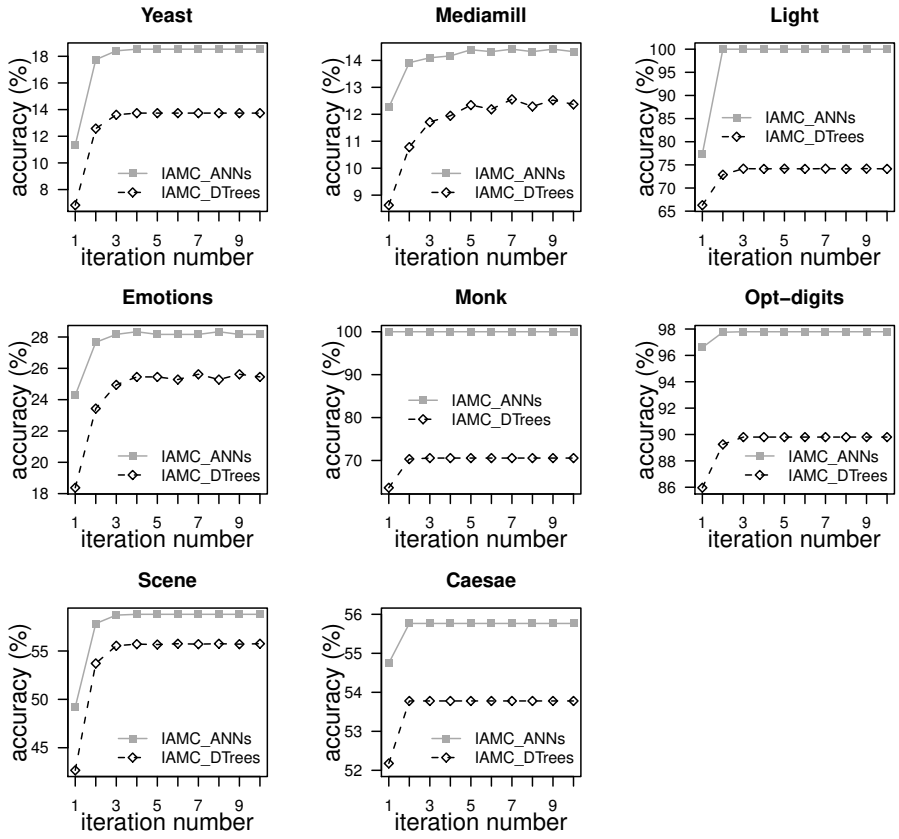

Fig. 4 Convergence of the IAMC methods against the eight data sets; −□− and −◇− denote IAMC approaches with Neural Networks and Decision Trees as single-target learning methods respectively

then stabilized. The graphs in Figure 4 also show that similar performances were achieved with the other seven data sets, regardless of the base learner applied. In particular, our further analysis indicates that, on all the eight data sets, the iteration process stopped in just a few iterations because the predicted values for all the target attributes converged. That is, the iteration stopped before reaching the maximum number of iterations pre-set for these experiments.

These results suggest that the iteration process employed in the IAMC approach was very effective and enabled the learning method to converge quickly.

4.3.3 Why Does Iterative Inference Benefit the IAMC Algorithm?

In order to better understand the benefits of the iterative inference to the IAMC algorithm, we study why the IAMC approach improves over the collection of inde-

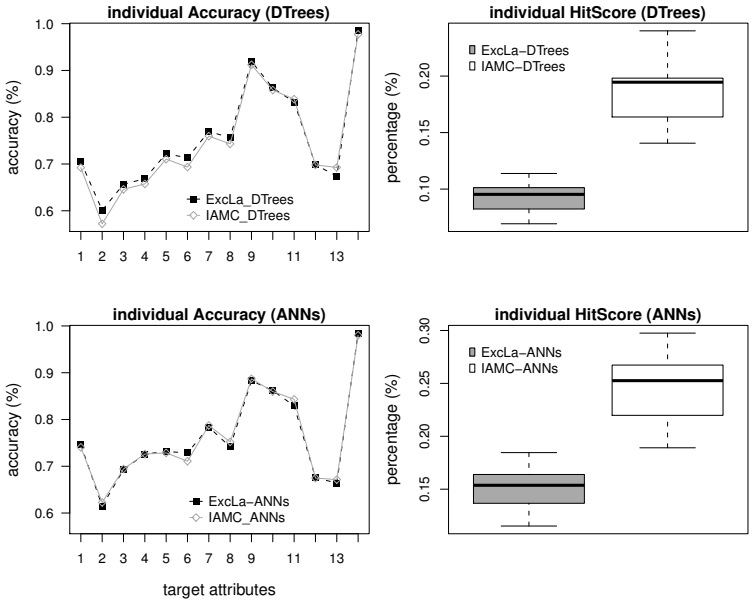

Fig. 5 Results for the Yeast data. Left subfigures: "per-target" accuracy for each of the 14 target attributes obtained by the ExcLa and IAMC methods with decision trees and neural networks as single-target learners respectively. Right subfigures (boxplot graphs): *mininum, median*, and *maximum* HitScore values obtained by the 14 single-target learners in the ExcLa and IAMC methods; a learner's HitScore is calculated as: among the instances that are correctly classified by the single-target learner, we measure what percentage of these instances is also correctly determined by all other single-target learners in the classifier collection.

pendent classifiers, namely the collection of *intrinsic* classifiers (equivalently, the collection of classifiers in the ExcLa algorithm). We examine the IAMC and ExcLa learning methods' predictions for each target attribute (we denote this accuracy as "per-target" accuracy since it measures the accuracy against individual target variable). We present the results against the Yeast and Opt-digits data sets in Figures 5 and 6, respectively. These two data sets were chosen because they contain more target attributes than other tested data.

In Figure 5, the left subfigures describe the per-target accuracy against each of the 14 target attributes of the Yeast data, obtained by the IAMC and ExcLa methods using decision trees and neural networks as single-target learners respectively. In the right subfigures among the instances that were correctly classified by a single-target learner, we measure what percentage of these instances was also correctly determined by all other single-target learners in the classifier collection (for description purpose, we denote this measurement as HitScore). In other words, the HitScore for an individual classifier k is calculated by the number of instances correctly classified by all classifiers divided by the total number of instances correctly classified by the learner k. The HitScore aims to indicate how useful a single-target learner's accuracy is in terms of simultaneous prediction of all target attributes. For the Opt-digits data, the accuracies and HitScores for the 10 individual target attributes were presented in Figure 6.

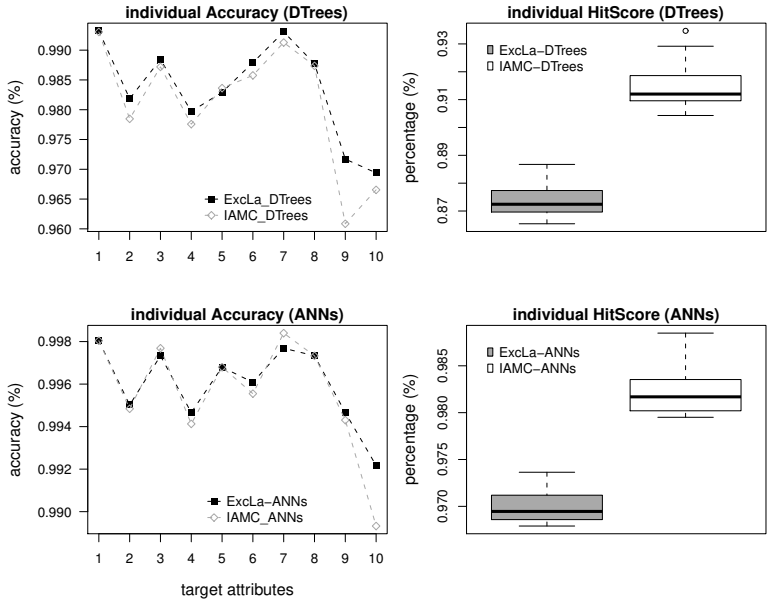

Fig. 6 "Per-target" accuracies and HitScores for the 10 individual target attributes of the Opt-digits data

The left subfigures in 5 and 6 show that all individual targets are predicted equally well for the two tested methods, namely obtaining similar "per-target" accuracy, but the right subfigures clearly show that all individual learners in our proposed method make more correct classifications on the same set of instances, resulting in higher "exact match" accuracy.

For the Yeast data, when considering using decision trees as single-target learners, the results as presented in the top-left subfigure of Figure 5 suggest that the IAMC and ExcLa methods obtained very similar accuracy for each target attribute. The two curves in this subfigure were almost identical. However, results as presented in the top-right boxplot subfigure show that the HitScores of the individual learners in the IAMC collection were higher than that in the ExcLa collection. The results indicate that even the lowest HitScore in the IAMC collection was higher than the highest HitScore in the ExcLa collection. These results imply that the iterative classification approach employed in the IAMC method allowed all individual learners to focus on correctly predicting the same set of instances, resulting in higher individual HitScore values. Interestingly, this goal may be achieved through degrading the predictions against some individual target attributes. For example, as shown in the top left subfigure, against target attribute 2, the prediction of the IAMC approach was lower than that of the ExcLa method. Similar results can also be observed in the bottom two subfigures of Figure 5, where neural networks were used as the single-target learners in the IAMC and ExcLa methods when against the Yeast data.

Against the Opt-digits data, results as presented in Figure 6 confirmed the above observations. That is, although the individual predictive performance of the learners in the IAMC collection was very similar with that in the ExcLa collection,

Table 5 Average execution time required (in second) for each fold for the IAMC_DTrees, ExcLa_DTrees, and En-MODTs methods

Data Set Used	IAMC_DTrees		ExcLa_DTrees	Ensembles of MODTs
	training	inference		
Yeast	15.4	0.12	10.3	7.4
Emotions	0.8	0.05	0.4	0.35
Scene	18	0.11	9.8	7.3
Mediamill	1003	1.7	617	236
Monks	0.08	0.04	0.04	0.01
Caesar	0.06	0.03	0.04	0.04
Light	3.7	0.15	1.9	0.8
Opt-digits	6.2	0.16	3.1	0.56

Table 6 Average execution time required (in second) for each fold for the IAMC_ANNs, ExcLa_ANNs, and MANNs methods

Data Set Used	IAMC_ANNs		ExcLa_ANNs	MANNs
	training	inference		
Yeast	8979	11.6	3955	73.89
Emotions	163	0.2	77.1	10.32
Scene	10647	13.7	5529	218.5
Mediamill	29398	63	11618	4105.2
Monks	4.11	0.1	1.8	1.79
Caesar	2.805	0.11	1.3	4.15
Light	506	1.2	134	100.48
Opt-digits	2276	3.5	1004	94.46

the individual learners in the IAMC collection made more useful correct classifications, in terms of simultaneous predictions of all target attributes correctly, than that of the learners in the ExcLa collection.

These experimental results suggest that for the Yeast and Opt-digits data sets, although the per-target accuracy, i.e., accuracy against individual target variable, is high for the ExcLa algorithms, the "exact match" accuracy is low. This is because individual learners make correct classifications on different instances of the test set. On the other hand, the iterative inference strategy may not improve the individual learners' predictive accuracy, i.e., the per-target accuracy, but it was able to encourage all individual classifiers to make correct classifications on common instances, thus improving the simultaneous predictions of the multiple target attributes.

4.3.4 Execution Time Required

To evaluate the performance of the four tested strategies in terms of run time, we provide the average one-fold running time needed (in seconds) for each of the eight data sets in Tables 5 and 6. Table 5 depicts the comparison of the tree-based methods, and Table 6 focuses on the ANNs-based approaches.

Results from Table 5 show that the IAMC method with decision trees as single-target learners was two or three times slower than the En-MODTs approach, and about two times slower than the ExcLa_DTrees method. As expected, the IAMC strategy was slower because the IAMC algorithm consists of two collections of single-target learners. When considering the comparison of ANNs-based methods, results as shown in Table 6 imply similar conclusions: the MANNs was the fastest and the IAMC_ANNs was about two times slower than the ExcLa_ANNs method. In addition, as expected, the results shown in Tables 5 and 6 indicate that the

IAMC method with neural networks as single-target learners required longer execution time when compared to using decision trees as single-target learners.

However, the time required for inference in the IAMC method is much smaller than its training time. For example, the inference time required for the IAMC method with decision trees as single-target learners was less than 0.16 seconds, except for the Mediamill data set, where the IAMC method required 1.7 seconds for the inference process. Efficient inferences were also obtained when neural networks were used as base learners. These observations suggest that parallel learning can be applied to the training of the IAMC strategy (because all individual intrinsic and relational models can be constructed simultaneously) to potentially significantly speed up the overall running of the IAMC method.

5 Conclusions and Future Work

Many real-world applications record data with multiple related target attributes and require simultaneously determining all the target attributes correctly. This paper proposed a novel solution, which employs an iterative classification strategy to exploit the relationships among multiple targets, to achieve higher accuracy.

We evaluated the proposed method using eight data sets. Experiments show that the strategy outperforms 1) an approach that constructs independent classifiers for each target, 2) a multitask neural network method, and 3) ensembles of multi-objective decision trees in terms of simultaneously predicting all target attributes correctly. A key advantage of the proposed method is that it allows many state-of-the-art single-target learning techniques to be directly "plugged" into the proposed scheme to resolve multi-target classification problems.

Future directions of this work include investigating alternative ordering techniques, studying the error bounds of the IAMC strategy, further researching the convergence properties of the IAMC method, and investigating the applicability of aggregated features (Guo and Viktor, 2006) for targets attributes when these targets have the same label set.

References

Alimoglu F, Alpaydin E (1997) Combining multiple representations and classifiers for handwritten digit recognition. In: Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, pp 637–640

Ando RK, Zhang T (2005) Structures from multiple tasks and unlabeled data. J Mach Learn Res 6:1817–1853

Bakker B, Heskes T (2003) Task clustering and gating for bayesian multitask learning. J Mach Learn Res 4:83–99

Baxter J (2000) A model of inductive bias learning. Journal of Artificial Intelligence Research 12:149–198

Ben-David S, Schuller R (2003) Exploiting task relatedness for mulitple task learning. In: COLT'03, pp 567–580

Bishop CM (1996) Neural Networks for Pattern Recognition. Oxford University Press, UK

Blockeel H, Raedt LD, Ramong J (1998) Top-down induction of clustering trees. In: ICML'98, Morgan Kaufmann, pp 55–63

Boutell MR, Luo J, Shen X, Brown CM (2004) Learning multi-label scene classification. Pattern Recognition 37(9):1757 – 1771

Caruana R (1997) Multitask learning. Machine Learning 28(1):41–75

Chakrabarti S, Dom B, Indyk P (1998) Enhanced hypertext categorization using hyperlinks. In: SIGMOD '98, ACM, New York, NY, USA, pp 307–318

Clark P, Niblett T (1989) The `CN2` induction algorithm. Mach Learn 3(4):261–283

Crammer K, Singer Y (2003) A family of additive online algorithms for category ranking. J Mach Learn Res 3:1025–1058

Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30

Elisseeff A, Weston J (2001) A kernel method for multi-labelled classification. In: Advances in Neural Info. Proc. Systems 14, pp 681–687

Evgeniou T, Pontil M (2004) Regularized multi–task learning. In: KDD'04, pp 109–117

Fang J, Ji S, Xue Y, Carin L (2008) Multitask classification by learning the task relevance. Signal Processing Letters, IEEE 15:593–596

Friedman M (1937) The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. Journal of the American Statistical Association 32(200):675–701

Friedman M (1940) A Comparison of Alternative Tests of Significance for the Problem of m Rankings. The Annals of Mathematical Statistics 11(1):86–92

Fürnkranz J, Hüllermeier E, Loza Mencía E, Brinker K (2008) Multilabel classification via calibrated label ranking. Mach Learn 73(2):133–153

van der Gaag LC, de Waal PR (2006) Multi-dimensional bayesian network classifiers. In: Probabilistic Graphical Models, pp 107–114

Gaag LCVD, Renooij S, Witteman C, Aleman BMP, Taal BG (2001) Probabilities for a probabilistic network: A case-study in oesophageal carcinoma. In: AI in Medicine, pp 123–148

Getoor L, Taskar B (2007) Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning). The MIT Press

Ghamrawi N, McCallum A (2005) Collective multi-label classification. In: CIKM '05, ACM, pp 195–200

Godbole S, Sarawagi S (2004) Discriminative methods for multi-labeled classification. In: In Proceedings of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, pp 22–30

Guo H, Viktor HL (2006) Mining relational data through correlation-based multiple view validation. In: KDD '06, New York, NY, USA, pp 567–573

Guo H, Viktor HL, Paquet E (2011) Privacy disclosure and preservation in learning with multi-relational databases. JCSE 5(3):183–196

Hollander M, Wolfe DA (1999) Nonparametric Statistical Methods, 2nd Edition. New York: John Wiley & Sons

Hüllermeier E, Fürnkranz J, Cheng W, Brinker K (2008) Label ranking by learning pairwise preferences. Artif Intell 172(16-17):1897–1916

Ji S, Tang L, Yu S, Ye J (2008) Extracting shared subspace for multi-label classification. In: KDD '08, ACM, New York, NY, USA, pp 381–389

Kocev D, Vens C, Struyf J, Džeroski S (2007) Ensembles of multi-objective decision trees. In: ECML'07, pp 624–631

Last M (2004) Multi-objective classification with info-fuzzy networks. In: ECML, pp 239–249

Lipkin M, Engle RL, Jr, Flehinger BJ, Gerstman LJ, Atamer MA (1969) Computer-aided differential diagnosis of hematologic diseases. Annals of the New York Aca of Sci 161:670–679

Lu Q, Getoor L (2003) Link-based classification. In: ICML'03

Macskassy SA, Provost F (2003) A simple relational classifier. In: Proceedings of the Second Workshop on Multi-Relational Data Mining (MRDM-2003) at KDD-2003, pp 64–76

Mitchell MT (1996) Machine Learning. McGraw Hill, New York, USA

Neville J, Jensen D (2000) Iterative classification in relational data. In: AAAI Workshop on Learning Statistical Models from Relational Data,13-20

Oh HJ, Myaeng SH, Lee MH (2000) A practical hypertext catergorization method using links and incrementally available class information. In: SIGIR '00, ACM, New York, NY, USA, pp 264–271

Quinlan JR (1993) C4.5: programs for machine learning. Morgan Kaufmann Publishers Inc., USA

Read J (2008) A pruned problem transformation method for multi-label classification. In: NZCSRS 2008, pp 143–150

Read J, Pfahringer B, Holmes G, Frank E (2009) Classifier chains for multi-label classification. In: ECML/PKDD (2)'09, pp 254–269

Siegel S, Castellan NJ (1988) Nonparametric Statistics for The Behavioral Sciences. McGraw-Hill Humanities

Snoek CGM, Worring M, van Gemert JC, Geusebroek JM, Smeulders AWM (2006) The challenge problem for automated detection of 101 semantic concepts in multimedia. In: MULTIMEDIA '06, ACM, pp 421–430

Suzuki E, Gotoh M, Choki Y (2001) Bloomy decision tree for multi-objective classification. In: PKDD '01, Springer-Verlag, London, UK, pp 436–447

Taskar B, Segal E, Koller D (2001) Probabilistic classification and clustering in relational data. In: IJCAI'01, pp 870–878

Thabtah FA, Cowling P, Peng Y (2004) Mmac: A new multi-class, multi-label associative classification approach. In: ICDM '04, pp 217–224

Thrun S, Bala J, Bloedorn E, Bratko I, Cestnik B, Cheng J, Jong KD, Dzeroski S, Fahlman S, Fisher D, Hamann R, Kaufman K, Keller S, Kononenko I, Kreuziger J, Michalski R, Mitchell T, Pachowicz P, Reich Y, Vafaie H, Welde WVD, Wenzel W, Wnek J, Zhang J (1991) The monk's problems a performance comparison of different learning algorithms. Tech. rep.

Tsoumakas G, Katakis I (2007) Multi label classification: An overview. International Journal of Data Warehouse and Mining 3(3):1–13

Tsoumakas G, Vlahavas I (2007) Random k-labelsets: An ensemble method for multilabel classification. In: ECML 2007, Warsaw, Poland, pp 406–417

Tsoumakas G, Katakis I, Vlahavas I (2009) Mining multi-label data. Unpublished book chapter

Ueda N, Saito K (2003) Parametric mixture models for multi-labeled text. URL citeseer.ist.psu.edu/ueda03parametric.html

Wieczorkowska A, Synak P, Raś Z (2006) Multi-label classification of emotions in music. Intelligent Info Processing and Web Mining pp 307–315

Witten IH, Frank E (2000) Data mining: practical machine learning tools and techniques with Java implementations. Morgan Kaufmann, CA, USA

Xi P, Lee WS, Shu C (2007) Analysis of segmented human body scans. In: GI '07, pp 19–26

Xue Y, Liao X, Carin L, Krishnapuram B (2007) Multi-task learning for classification with dirichlet process priors. J Mach Learn Res 8:35–63

Yang Y (2001) A study on thresholding strategies for text categorization. In: SIGIR, ACM Press, pp 137–145

Yu K, Tresp V, Schwaighofer A (2005) Learning gaussian processes from multiple tasks. In: ICML '05, ACM, New York, NY, USA, pp 1012–1019

Zenko B, Dzeroski S (2008) Learning classification rules for multiple target attributes. In: PAKDD'08, pp 454–465

Zhang ML, Zhou ZH (2006) Multilabel neural networks with applications to functional genomics and text categorization. IEEE Trans on Knowl and Data Eng 18(10):1338–1351

Zhang ML, Zhou ZH (2007a) Ml-knn: A lazy learning approach to multi-label learning. Pattern Recognition 40:2007

Zhang ML, Zhou ZH (2007b) Multi-label learning by instance differentiation. In: AAAI'07, AAAI Press, pp 669–674