Combining Two Aspects of Meta-Learning with Heterogeneous Meta Decision Trees

Ljupčo Todorovski, Sašo Džeroski
Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia
{Ljupco.Todorovski,Saso.Dzeroski}@ijs.si

Abstract *Two aspects of meta-learning are considered in the paper. The first is selecting the appropriate machine learning algorithm(s) for a given data set. The second is combining the predictions obtained with classifiers induced using different machine learning algorithms. A novel method for combining classifiers, based on meta decision trees (MDTs), is presented in the paper. Instead of giving a prediction, MDT leaves specify which classifier should be used to obtain a prediction. Decisions in MDTs depend on a set of meta-level attributes summarizing the properties of class probability distributions predicted by the base-level classifiers. The set of meta-level attributes is the same for different data sets: this property allows us to induce heterogeneous MDTs by using examples from more than one data set. In this case meta-level attributes summarizing the properties of the data set where the example originates from, can be added to the set of meta-level attributes for inducing MDTs. The performance of heterogeneous MDTs is empirically tested on twenty UCI data sets.*

1 Introduction

There are at least two aspects of meta-learning. The first aspect is choosing the most suitable machine learning algorithm(s) for a given data set [1]. The second one is combining the predictions obtained from multiple models induced using different machine learning algorithms [4].

The problem of choosing the most suitable machine learning algorithm(s) occurs when considering new data sets for analysis. The choice can be especially time consuming in the process of knowledge discovery in very large data sets, where it takes a lot of time to apply all the machine algorithms available in order to measure their performance. To avoid this problem, a machine learning algorithm can be used at the meta-level to learn to predict how well each of the (base-level) machine learning algorithms will perform on the data set from the properties of the data set (such as number of examples, class entropy, etc.). Using this predictor, users can discard algorithms that are not suitable for the data set at hand and save a lot of effort in trying out all the algorithms.

For data sets where several machine learning algorithms have been used to induce multiple (base-level) models, a problem of combining their predictions appears. The methods for combining the predictions obtained from the base-level models can be clustered in three combining paradigms: voting, stacking [15] and cascading [6]. In a voting scheme, each base-level model gives a vote for its prediction. The prediction receiving the most votes is the final prediction. In stacking, a learning algorithm is used to induce a meta-level model for combining the predictions of the base-level models. Cascading is an iterative process of combining multiple models: at each iteration, the learning data set is extended with the predictions obtained in the previous iteration.

In the paper, we propose a novel method for combining multiple models that brings together these two aspects of meta-level learning. It is based on meta decision trees (MDTs). The difference between meta and ordinary decision trees (ODTs) is that MDT leaves specify which base-level model should be used, instead of predicting the class value directly. The decisions in MDTs are based on a set of meta-level attributes that summarize th properties of the class probability distributions predicted by the base-level models for a given example [13]. Since the same set of meta-level attributes is used for different data sets, heterogeneous MDTs can be induced using examples from different data sets. In this case, the set of meta-level attributes can be extended with the properties of the data set where a given example originates from. These are the meta-level attributes used in studies related to the first aspect of meta-level learning.

The paper is organized as follows. In Section 2, the first aspect of meta-learning is presented. The second aspect of meta-learning and meta decision trees are presented in Section 3. Heterogeneous MDTs are described in Section 4. The results of the experimental evaluation of using heterogeneous MDTs for combining five classifiers on twenty data sets are presented in Section 5. Finally, Section 6 concludes the paper and gives directions for further work.

2 Selecting Appropriate Classifiers

One aspect of meta-level learning is choosing the most appropriate machine learning algorithm(s) for a given data set. In order to do this, we try to relate the performance of different machine learning algorithms on a given data set to the properties of that data set. This relation is induced using a learning algorithm at the meta-level. The meta-level data set consists of attributes representing different properties of data sets and a class value representing the performance of a machine learning algorithm on the data set.

Several different approaches to meta-level learning have been proposed. In [1] a set of trivial, statistical and information theory measures of data sets is related to the performance of different classification algorithms. This set includes trivial measures like the number of classes, attributes, examples, etc.; statistical measures like average skewness and kurtosis of attribute values distributions; and information theory based measures like entropy of the class distribution and average mutual information between attributes and class. For each data set, classification algorithms are, according to their performance, clus-

tered in two categories: applicable and inapplicable. Thus, the meta-level model predicts whether the machine learning algorithm is applicable or inapplicable to a given data set.

Another approach, named landmarking [10], makes use of the performances of a set of simple learning algorithms (named landmarkers) on the given data set as the meta-level properties of that data set. Also, instead of predicting the applicability of machine learning algorihtm for a given task, ranking of different machine learning algorithms according to their performance on a given data set can be predicted, like in [5].

In this preliminary study, we used a reduced set of seven meta-level attributes to characterize the data set at hand. These are the following: number of classes, examples, discrete and continuous attributes, number of all attributes, maximal class probability and entropy of the class probability distribution. In contrast with the framework presented above where applicability of different classifiers are predicted, the meta-level models for selecting classifier induced in this paper predict which base-level classification algorithm would perform best on a given data set.

Table 1: Meta-level model induced with C4.5 as a meta-level learner

```
nda > 13 :  C45
nda <= 13 :
|       nc <= 4 :  LB
|       nc > 4 :  KNN
```

An example of a meta-level decision tree induced by C4.5 from the seven meta-level attributes is presented in Table 1. It specifies that C4.5 performs best in the data sets with more than thirteen discrete attributes. For other data sets (with at most thirteen discrete attributes) the choice is made on the basis of the number of classes. If there are more than four classes in the data set, the best classifier is k-NN; otherwise linear Bayes is chosen.

3 Combining Multiple Classifiers

3.1 Overview of State-of-the-Art Methods

Another aspect of meta-level learning is combining the predictions of a diverse set of base-level models induced with different machine learning algorithms. Given the predictions of the base-level models, the combining algorithm has to make the final prediction for a given example. Meta-level combining techniques make use of a machine learning algorithm to induce a meta-level model for combining the predictions of the base-level models.

One of the meta-level methods for combining multiple models is stacking [15], where neural network is typically used as a meta-level learner. Decision trees along with a naive Bayesian classifier have also been used as meta-level learners in [2]. In the same study, hierarchical multi-level combining method, based on tree structures, is introduced. At higher levels, meta-level classifiers from the previous levels are combined. Cascading [6] is an iterative method for combining classifiers. At each iteration, the training data set is extended with the class probability distributions predicted by the classifier induced in the previous iteration.

Meta decision trees (MDTs), a novel method for combining multiple models, have been proposed in [13]. MDTs can not be used to learn base-level models: they are specialized for use at the meta-level.

3.2 Meta Decision Trees

The structure of a meta decision tree is identical to the structure of an ordinary decision tree (ODT). A decision (inner) node specifies a test to be carried out on a single attribute value and each outcome of the test has its own branch leading to the appropriate subtree. In a leaf node, a MDT predicts which model is to be used for classification of an example, instead of predicting the class value of the example directly like an ODT.

In the process of inducing meta decision trees two types of attributes are used. *Ordinary* attributes are used in the decision (inner) nodes of the MDT. The role of these attributes is identical to the role of the attributes used for inducing ordinary decision trees. *Class* attributes are used in the leaf nodes only. Each base-level model has its class attribute: the values of the class attribute are equal to the predictions obtained by the base-level model. Thus, the class attribute assigned to the leaf node of the MDT decides which base-level model should be used for prediction.

Ordinary attributes in MDTs are properties of the class probability distributions predicted for a given example by the base-level models. Namely, the most general form of a prediction returned by a classification model for a given example is a probability distribution over the possible classes. Let the base-level classifier $C_{\mathcal{L}}$ generated with learning algorithm $\mathcal{L}$ return the probability distribution $\mathbf{p}_{\mathcal{L}}(e)$, when applied to example e:

$$\mathbf{p}_{\mathcal{L}}(e) = (p_{\mathcal{L}}^{(1)}(e), p_{\mathcal{L}}^{(2)}(e), \ldots, p_{\mathcal{L}}^{(m)}(e)),$$

where m is the number of classes. The k-th element in this vector denotes the probability that the example e belongs to class c_k as estimated by the model $C_{\mathcal{L}}$. The class c_* with the highest class probability $p_{\mathcal{L}}^{(*)}$ is predicted by base-level classifier $C_{\mathcal{L}}$.

The following three properties of class probability distributions are used as ordinary attributes in MDTs. First, $\mathcal{L}$_maxprob is the highest class probability (i.e. the probability of the predicted class):

$$\mathcal{L}\text{\_maxprob} = \max_{k=1}^{m} p_{\mathcal{L}}^{(k)}(e).$$

Next, $\mathcal{L}$_entropy is the entropy of the class probability distribution:

$$\mathcal{L}\text{\_entropy} = -\sum_{k=1}^{m} p_{\mathcal{L}}^{(k)}(e) \cdot \log_2 p_{\mathcal{L}}^{(k)}(e).$$

Finally, the $\mathcal{L}_{\text{weight}}$ is the fraction of the training examples used to estimate the class distribution for example e. For decision trees, it is the weight of the examples in the leaf node used to classify the example. For rules, it is the weight of the examples covered by the rule(s) which was used to classify the example. This property does not apply to the nearest neighbor and naive Bayes classifiers.

Table 2: A meta decision tree induced from the balance data set.

```
ltree_entropy <= 0.37699:
|       knn_maxprob <= 0.75079:  LTREE (*)
|       knn_maxprob > 0.75079:  KNN
ltree_entropy > 0.37699:  (**)
|       knn_entropy > 1.49841:  KNN
|       knn_entropy <= 1.49841:
|       |       c45_weight <= 0.11388:  LTREE
|       |       c45_weight > 0.11388:
|       |       |       c45_maxprob <= 0.95:  LTREE
|       |       |       c45_maxprob > 0.95:  C45
```

An example MDT is given in Figure 2. The leaf denoted by (*) specifies that the LTree model is to be used to classify an example if the entropy of the probability distribution returned by it is smaller than 0.38 and the maximum probability in the probability distribution returned by the k-NN model is smaller than 0.75. In sum, if the LTree model is confident in its prediction and the k-NN model is not so confident in its prediction, the leaf recommends using the LTree prediction, which is consistent with common sense in the domain of model combination. The other branch of the tree (**) is based on a much smaller number of examples and is thus much less reliable. It also doesn't make much sense.

Despite the fact that the MDTs are presented as a combining method, they can be also viewed as a method for choosing appropriate classifier for each individual example in the data set. This is different from other methods for selecting appropriate classifiers, presented in Section 2, where the choise is made for the whole data set and not for the individual examples. Nevertheless, they allow us to bring together methods for selecting and combining classifiers, as described in the following subsection.

4 Heterogeneous Meta Decision Trees

Heterogeneous Meta Decision Trees (HMDTs) are MDTs induced from more than one base-level data set. Inducing MDTs on more than one data set is possible because the same set of meta-level attributes, summarizing the properties of class probability distributions predicted by the base-level classifiers, are used for different data sets. However, if we induce HMDTs using the same set of meta-level attributes as for MDTs, the selection of appropriate classifier for an example would be independent of the data set where the example originates from. In order to take the data set where an example originates from into account, HMDTs can use meta-level attributes that summarize data set properties. Seven such properties are used in our experiments: number of classes (nc), number of examples (ne), number of discrete (nda) and continuous (nca) attributes, number of all attributes (na), maximal class probability (class_maxprob) and entropy of the class probability distribution (class_entropy). These are the seven data set properties used in Section 2 as a set of meta-level attributes for selecting the appropriate classifier.

Table 3: A heterogeneous meta decision tree induced from twenty data sets using the extended set of meta-level attributes.

```
cn2_maxprob <= 0.70032:  LTREE
cn2_maxprob > 0.70032:
|       knn_maxprob <= 0.84143:
|       |       ne <= 768:  LB
|       |       ne > 768:  CN2 (*)
|       knn_maxprob > 0.84143:
|       |       ltree_maxprob <= 0.978:
|       |       |       nca <= 0:  CN2
|       |       |       nca > 0:  KNN (**)
|       |       ltree_maxprob > 0.978:
|       |       |       class_entropy <= 1.20574:
|       |       |       |       ne <= 768:  CN2
|       |       |       |       ne > 768:
|       |       |       |       |       ne <= 2310:  CN2
|       |       |       |       |       ne > 2310:  C45
|       |       |       class_entropy > 1.20574:
|       |       |       |       ltree_maxprob <= 0.998:  KNN
|       |       |       |       ltree_maxprob > 0.998:  LTREE
```

redundant? is it because of MLC45 or it can also happens in C4.5

HMDTs bring together two aspects of meta-level learning. They are used to combine classifiers (second aspect of meta-learning) by selecting the appropriate classifier for each individual example from different data sets. The set of meta-level attributes used in HMDTs is the union of the meta-level attributes used in the methods related to each aspect of meta-level learning.

An example heterogeneous MDT induced from twenty data sets using the extended set of meta-level attributes is given in Figure 3. The leaf denoted by (*) specifies that the CN2 model should be used to classify an example if (1) the maximum probability in the class probability distribution predicted by CN2 is larger than 0.7; (2) the maximum probability in the class probability distribution returned by k-NN is smaller than 0.84; (3) the number of examples in the data set where the example originates from is 768. Note that this branch is partially consistent with meta-learning common sense: the CN2 model is chosen if it is confident with its prediction (maximum probability larger than 0.7) and the k-NN model is not so confident at the same time. The decision in the node marked (**) also makes sense: the k-NN algorithm performs better than CN2 in data sets with continuous attributes.

5 Experiments

5.1 Data Sets and Base-Level Experiments

In order to evaluate the performance of meta decision trees, we performed experiments on a collection of twenty data sets from the UCI Repository of Machine Learning Databases and Domain Theories [9]. These data sets have been widely used in other comparative

studies. The data set properties (used also as meta-level attributes, see Section 4) are given in Table 4.

Table 4: Data set properties

Data set	nc	ne	nda	nca	na	class_maxprob	class_entropy
australian	2	690	8	6	14	0.56	0.99
balance	3	625	0	4	4	0.46	1.32
breast-w	2	699	9	0	9	0.66	0.92
bridges-td	2	102	4	3	7	0.85	0.61
car	4	1728	6	0	6	0.70	1.21
chess	2	3196	36	0	36	0.52	0.99
diabetes	2	768	0	8	8	0.65	0.93
echocardiogram	2	131	1	5	6	0.67	0.91
german	2	1000	13	7	20	0.70	0.88
glass	6	214	0	9	9	0.36	2.18
heart	2	270	6	7	13	0.56	0.99
hepatitis	2	155	13	6	19	0.79	0.74
hypothyroid	2	3163	18	7	25	0.95	0.29
image	7	2310	0	19	19	0.14	2.78
ionosphere	2	351	0	34	34	0.64	0.94
iris	3	150	0	4	4	0.33	1.58
soya	19	683	35	0	35	0.13	3.79
tic-tac-toe	2	958	9	0	9	0.65	0.93
vote	2	435	16	0	16	0.61	0.96
wine	3	178	0	13	13	0.40	1.56

Five learning algorithms were used in the base-level experiments: two tree-learning algorithms C4.5 [11] and LTree [7], the rule-learning algorithm CN2 [3], the k-nearest neighbor (k-NN) algorithm [14] and a modification of the naive Bayes algorithm [8]. All algorithms were used with their default settings.

5.2 Meta-Level Experiments

In the meta-level experiments, we compared the performances of four different meta-level approaches. Three of them are combining methods based on MDTs and the last one is a method for selecting the most appropriate classifier.

The classification errors of different meta-level approaches are presented in Table 5. Classification errors were measured using 10-fold stratified cross validation. The data set was first partitioned into ten folds with equal sizes and similar class distributions. Nine folds of the data set was used for inducing the model, which is then tested on the remaining fold. In the experiments with heterogeneous MDTs, the nine folds from all twenty data sets are merged to induce an HMDT, which is then tested on the remaining fold of each data set. In all experiments, cross validation is repeated 10 times using a different random reordering of the examples in the data set. The same set of re-orderings were used for all experiments. The average and standard deviation (over the ten cross validations) of the classification error on unseen examples are reported.

The performances of the following four meta-learning methods are compared:

Table 5: Classification errors (in %) of different meta-level classifiers

Data set	MDT		HMDT		HMDT-W		ML	
australian	14.62	±0.89	14.17	±0.66	14.88	±0.69	14.13	±0.23
balance	7.22	±0.60	7.75	±0.87	9.45	±0.65	13.31	±0.31
breast-w	3.06	±0.37	3.35	±0.37	4.08	±0.36	6.43	±0.39
bridges-td	16.03	±1.72	14.91	±1.29	14.60	±1.70	12.10	±0.64
car	4.04	±0.51	4.17	±0.47	4.29	±0.66	5.90	±0.72
chess	0.54	±0.08	0.58	±0.07	0.56	±0.07	0.52	±0.03
diabetes	23.95	±0.98	23.78	±0.62	24.66	±0.72	25.89	±0.49
echo	31.54	±1.11	31.97	±3.47	31.85	±1.73	29.16	±1.37
german	25.51	±0.66	25.63	±0.89	27.25	±0.97	26.59	±0.56
glass	30.08	±1.80	29.55	±2.01	31.05	±1.88	36.94	±0.73
heart	17.54	±1.21	17.20	±1.43	18.40	±1.89	15.11	±0.52
hepatitis	17.53	±1.16	16.84	±1.71	17.41	±2.33	15.10	±0.68
hypothyroid	0.78	±0.07	0.94	±0.09	1.10	±0.09	0.72	±0.04
image	2.52	±0.22	2.67	±0.35	2.94	±0.24	14.05	±0.44
ionosphere	9.49	±0.77	9.37	±1.09	9.86	±0.82	13.50	±0.54
iris	2.67	±0.52	3.03	±0.70	3.95	±1.34	2.00	±0.00
soya	6.02	±0.55	5.90	±0.43	6.06	±0.35	17.69	±0.44
tic-tac-toe	0.52	±0.25	0.77	±0.17	0.61	±0.24	30.42	±0.24
vote	4.26	±0.38	4.81	±0.39	4.66	±0.42	9.77	±0.24
wine	2.26	±0.88	2.56	±0.76	2.55	±0.82	2.81	±0.70
Average	**11.01**	±0.74	**11.00**	±0.89	**11.51**	±0.90	**14.61**	±0.47

MDT is a stacking algorithm with meta decision trees induced on a single data set. Details about the stacking algorithm are given at the end of this section.

HMDT is a stacking algorithm with heterogeneous meta decision trees induced on all twenty (base-level) data sets.

HMDT-W is the same as **HMDT**, except for giving different weight to the examples in the training set. When uniform weights of the examples are used, as in **HMDT**, data sets with a large number of examples have bigger impact on the induced heterogeneous MDT. In order to enlarge the impact of the examples from the data set where the induced MDT will be used, half of the total weight is given to that data set. The other half of the total weight is equally distributed among other data sets. The fraction of the weight for each data set is uniformly distributed among the examples that belong to that data set.

ML is a meta-level method for selecting the most appropriate classifier using C4.5 at the meta-level. We used the leave-one-out method to carry out experiments with **ML** instead of 10-fold cross validation: the meta-level model for each data set was induced using meta-level data about the other nineteen data sets. Classification performance of the base-level classifiers was measured using 10-fold cross validation. The data set properties described in Section 3 were used as meta-level attributes.

The stacking algorithm we use is similar to the standard stacking technique described in [15]. For time complexity reasons, we use stratified 10-fold cross validation for base-level

classification instead of the leave-one-out method. By performing 10-fold cross validation on the training set, the class attributes in the meta-level training set are obtained by classifying testing examples that were not used for building the base-level classifiers.

Another difference from the standard stacking technique is the use of class probability distributions obtained by the base-level classifiers. Standard stacking methods only use class predictions. The output of the base-level classifier for each example in a data set consist of at least two components: the predicted class and the class probability distribution. All the base-level algorithms used in this study calculate the class probability distribution for classified examples, but two of them (k-NN and naive Bayes) do not calculate the weight of the examples used for classification (see Section 3). The code of the other three of them (C4.5, CN2 and LTree) was adapted to output the class probability distribution as well as the weight of the examples used for classification.

To summarize, the following set of meta-level attributes is used for inducing meta decision trees: C45_maxprob, C45_entropy, C45_weight, CN2_maxprob, CN2_entropy, CN2_weight, kNN_maxprob, kNN_entropy, LTree_maxprob, LTree_entropy, LTree_weight, LB_maxprob, LB_entropy, C45_prediction, CN2_prediction, kNN_prediction, LTree_prediction and LB_prediction. The last five meta-level attributes (predictions of the base-level attributes) are used as class attribtues in the leaves of the MDTs. In the case of inducing hterogeneous MDTs seven meta-level attributes describing the data set properties are added.

For the purpose of inducing MDTs and HMDTs, we developed MLC4.5 [13], a modification of C4.5. The entropy based "impurity of the node" measure used in C4.5 is replaced with the accuracy based one. The current version of MLC4.5 does not include any techniques for post-pruning MDTs.

5.3 Experimental Results

First, we calculated the average classification errors achieved with the described meta-level methods (see Table 5). The lowest average error is achieved using the **HMDT** method. A comparative analysis of the performance of **HMDT** versus the other four meta-level classifiers is given in Table 6.

The figures in Table 6 represent the relative error reduction achieved by using the **HMDT** algorithm as compared to each of the other algorithms, calculated as $1 - hmdt\_error/other\_method\_error$. Positive/negative figures denote better/worse performance of **HMDT**. The statistical significance of the differences is tested using paired t-tests with significance level of 95%: $+/-$ to the right of a figure in the table means that **HMDT** is significantly better/worse. The average here is calculated as $1 - \texttt{GeometricMean}(hmdt\_error/other\_method\_error)$.

When compared to the **HMDT-W** method, where examples from different data sets have different weights, **HMDT** is significantly better in seven and significantly worse in only one data set, giving the overall improvement of 5%. We can conclude that enlarging the impact of the data set at hand when inducing heterogeneous MDTs significantly deteriorates their predictive performance. However, there are other re-weighting schemes (like

Table 6: Classification error reduction (in %) achieved with stacking using meta decision trees as compared to other meta-level classifiers. The + and - signs indicate the significance of the difference at 95% level.

Data set	MDT		HMDT-W		ML	
australian	3.08		4.77	+	-0.28	
balance	-7.34		17.99	+	41.77	+
breast-w	-9.48		17.89	+	47.90	+
bridges-td	6.99		-2.12		-23.22	-
car	-3.22		2.80		29.32	+
chess	-7.41		-3.57		-11.54	-
diabetes	0.71		3.57	+	8.15	+
echo	-1.36		-0.38		-9.64	-
german	-0.47		5.94	+	3.61	+
glass	1.76		4.83	+	20.01	+
heart	1.94		6.52		-13.83	-
hepatitis	3.94		3.27		-11.52	-
hypothyroid	-20.51	-	14.55	+	-30.56	-
image	-5.95		9.18		81.00	+
ionosphere	1.26		4.97		30.59	+
iris	-13.48		23.29		-51.50	-
soya	1.99		2.64		66.65	+
tic-tac-toe	-48.08	-	-26.23	-	97.47	+
vote	-12.91	-	-3.22		50.77	+
wine	-13.27		-0.39		8.90	
Average	**-5.50**		**4.82**		**33.48**	

giving all the data sets equal weights) that should be also considered.

Despite the fact that the average classification error of HMDT when compared to MDT is smaller, Table 6 shows that HMDT is slightly worse than MDT. The performance of HMDT is even significantly worse in three data sets. Therefore, we can conclude that inducing (H)MDTs on more than one data set does not improve their performance. This can be due to the fact that we used a reduced set of only seven trivial data set properties. The reduced set of meta-level attributes can also be the reason for the poor performance of the ML method. It is 33% worse than HMDT on average, being significantly worse in eleven out of twenty domains.

6 Discussion

We have presented a new technique for combining classifiers based on meta decision trees (MDTs). Heterogeneous meta decision trees bring together two aspects of meta-level learning. They combine base-level classifiers (second aspect) by selecting the appropriate classifier (first aspect) for individual examples in the data set. The set of meta-level attributes used to induce HMDTs contains attributes used in methods related to each of the two

aspects of meta-learning.

The performance of HMDTs is slightly worse than the performance of MDTs induced on a single data set. This can be due to the fact that we used only seven meta-level attributes summarizing the data set properties. Thus, the first obvious direction for further work is to use a complete set of data set properties used in other studies related to the first aspect of meta-level learning [1]. Data set properties based on landmarking, used in a recent study on meta-learning [10], can be also added to the set of meta-level attributes used to induce HMDTs.

In the meta-level experiments with HMDTs, we only used one re-weighting scheme for examples originating from different data sets. Other schemes, like giving all the data sets equal weights, should be also empirically evaluated.

The consistency of meta decision trees with common sense model combination knowledge, as briefly discussed in Section 3, opens another direction for further research. The process of inducing meta-level classifiers should be biased to produce only meta-level classifiers consistent with existing knowledge. This can be achieved using strong language bias within MLC4.5 or, probably more easily, within a framework of meta decision rules, where rule templates could be used. Strong declarative bias can be also defined if we use Inductive Logic Programming (ILP) learning algorithm at the meta-level. In order to do this, the ILP meta-learning framework presented in [12] should be extended in the direction of inducing HMDTs.

References

[1] Brazdil, P. B. and Henery, R. J. (1994) Analysis of Results. In Michie, D., Spiegelhalter, D. J., and Taylor, C. C., editors: *Machine learning, neural and statistical classification.* Ellis Horwood.

[2] Chan, P. K. and Stolfo, S. J. (1997) On the Accuracy of Meta-learning for Scalable Data Mining. *Journal of Intelligent Information Systems* 8(1): 5–28.

[3] Clark, P. and Boswell, R. (1991) Rule induction with CN2: Some recent improvements. In *Proceedings of the Fifth European Working Session on Learning*: 151–163. Springer-Verlag.

[4] Dietterich, T. G. (1997) Machine-Learning Research: Four Current Directions. *AI Magazine* 18(4): 97–136.

[5] Gama, J. and Brazdil, P. (1995) Characterization of Classification Algorithms. In Pintro-Ferreira, C. et al., editors: *Progress in Artificial Intelligence.* Springer-Verlag.

[6] Gama, J. (1998) Combining Classifiers by Constructive Induction. In *Proceedings of the Ninth European Conference on Machine Learning.* Springer-Verlag.

[7] Gama, J. (1999) Discriminant trees. In *Proceedings of the Sixteenth International Conference on Machine Learning*: 134-142. Morgan Kaufmann.

[8] Gama, J. (2000) A Linear-Bayes Classifier. Technical Report. Artificial Intelligence and Computer Science Laboratory, University of Porto.

[9] Murphy, P. M. and Aha, D. W. (1994) *UCI repository of machine learning databases* [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

[10] Pfahringer, B., Bensusan, H. and Giraud-Carrier, C. (2000) Meta-Learning by Landmarking Various Learning Algorithms. To appear in *Proceedings of the Seventeenth International Conference on Machine Learning.*

[11] Quinlan, J. R. (1993) *C4.5: Programs for Machine Learning.* Morgan Kaufmann.

[12] Todorovski, L. and Džeroski, S. (1999) Experiments in Meta-Level Learning with ILP. In *Proceedings of the Third European Conference on Principles of Data Mining and Knowledge Discovery*: 98–106. Springer-Verlag.

[13] Todorovski, L. and Džeroski, S. (2000) Meta Decision Trees. Submitted to *ECML2000 Workshop on Meta-learning: Building Automatic Advice Strategies for Model Selection.*

[14] Wettschereck, D. (1994) *A study of distance-based machine learning algorithms.* PhD Thesis, Department of Computer Science, Oregon State University, Corvallis, OR.

[15] Wolpert, D. (1992) Stacked Generalization. *Neural Networks* 5(2): 241–260.