On the combination of two decompositive multi-label classification methods

Grigorios Tsoumakas[1], Eneldo Loza Mencía[2], Ioannis Katakis[1], Sang-Hyeun Park[2], and Johannes Fürnkranz[2]

[1] Aristotle University of Thessaloniki, Greece
{greg,katak}@csd.auth.gr
[2] Technical University Darmstadt, Germany
{loza,park,juffi}@ke.tu-darmstadt.de

Abstract. In this paper, we compare and combine two approaches for multi-label classification that both decompose the initial problem into sets of smaller problems. The Calibrated Label Ranking approach is based on interpreting the multi-label problem as a preference learning problem and decomposes it into a quadratic number of binary classifiers. The HOMER approach reduces the original problem into a hierarchy of considerably simpler multi-label problems. Experimental results indicate that the use of HOMER is beneficial for the pairwise preference-based approach in terms of computational cost and quality of prediction.

1 Introduction

Traditional single-label classification is concerned with learning from a set of examples that are associated with a single label λ from a set of disjoint labels $\mathcal{L}, |\mathcal{L}| > 1$. In multi-label classification, the examples are associated with *a set* of labels $P \subseteq L$. Multi-label classification is of high practical relevance as it naturally surfaces in many problems with textual, multimedia, or biological data[3].

The methods that are proposed in the literature can be categorized into two different groups [1]: i) problem transformation methods, and ii) algorithm adaptation methods. The first group includes methods that are algorithm independent. They transform the multi-label classification task into one or more single-label classification, regression or ranking tasks. The second group includes methods that extend specific learning algorithms in order to handle multi-label data directly. Recently, a special group of transformation methods that *decompose* the original multi-label classification problem into a series of simpler problems has been proposed [2, 3]. These *decompositive* methods focus on dealing with problems with large number of labels.

In this paper, we compare and combine two recently proposed decompositive methods. The *HOMER* approach [2] decomposes the problem into a hierarchy of simpler problems, where each problem uses a reduced number of possible labels. The hierarchical structure of the labels is obtained by applying recursive

[3] A collection of datasets can be found at http://mlkd.csd.auth.gr/multilabel.html

clustering to the initial set of labels. The *Calibrated Label Ranking* approach [3] interprets a multi-label problem as a special case of a preference learning problem [4]. The sets of relevant labels that are associated with the training examples are interpreted as a bipartite preference relation between relevant and irrelevant labels. Each possible pairwise preference is independently modeled with a binary classifier. The predictions of these classifiers are then combined into an overall ranking of all labels, and an artificial calibration label indicates the position where the ranking should be split into relevant and irrelevant classes.

An obvious disadvantage of the preference-based approach is the need to train a quadratic number of classifiers. Recent studies have shown that training time is reasonable because: a) the individual problems are much smaller than in other approaches (e.g. the binary relevance method), and b) an efficient algorithm for prediction has recently been proposed [5]. However, the problem of having to store a quadratic number of classifiers still remains to be solved, despite some recent progress for particular families of base classifiers [6].

For this reason, we investigate the combination between HOMER and Calibrated Ranking, because the latter is expected to greatly benefit from the reduction in the number of labels that is provided by the former. In fact, our experimental results indicate that HOMER is able to improve the classification performance, training time, and classification time for the calibrated ranking approach as well as for the binary relevance approach.

The following section recapitulates the two methods that we study and combine. The extensive experimental study is presented in Section 3 while our conclusions are included in the last section of the paper.

2 Multi-label Classification Methods

In this section, the basic multi-label classification methods, HOMER and Calibrated Label Ranking (CLR) are described.

We represent an instance or object as a vector $\bar{\mathrm{x}} = (x_1, \ldots, x_a)$ in a feature space $\mathcal{X} \subseteq \mathbb{R}^a$. Each instance $\bar{\mathrm{x}}_i$ is assigned to a set of relevant labels P_i, a subset of the n possible classes $\mathcal{L} = \{\lambda_1, \ldots, \lambda_n\}$. For multi-label problems, the cardinality $|P_i|$ of the label sets is not restricted, whereas for binary problems it is exactly one. For the latter case we denote the set of classes with $\mathcal{L} = \{1, -1\}$ so that each object $\bar{\mathrm{x}}_i$ is assigned to a class $\lambda_i \in \{1, -1\}$, $P_i = \{\lambda_i\}$. Multi-label learning algorithms are trained on a training set $D = \{(\bar{\mathrm{x}}_i, P_i) \mid i = 1 \ldots |D|\}$.

2.1 HOMER

HOMER follows the divide-and-conquer paradigm of algorithm design. The main idea is the transformation of a multi-label classification task with a large set of labels $\mathcal{L}$ into a tree-shaped hierarchy of simpler multi-label classification tasks, each one dealing with a small number $k << n$ of labels.

This tree-shaped hierarchy has n leaves, one leaf for each class λ_j of $\mathcal{L}$. Each internal node ν contains the union of the label sets of its children, $\mathcal{L}_\nu = \bigcup \mathcal{L}_c, c \in \text{children}(\nu)$. The root contains all labels, $\mathcal{L}_{\text{root}} = \mathcal{L}$.

Each non-leaf node represents a multi-label prediction problem that assigns a set of meta-labels to an example. A *meta-label* μ_ν of a node ν is defined as the disjunction of the labels contained in that node, i.e., $\mu_\nu \equiv \bigvee \lambda_j, \lambda_j \in \mathcal{L}_\nu$. Meta-labels have the following semantics: a training example can be considered to be annotated with meta-label μ_ν if it is annotated with at least one of the labels in $\mathcal{L}_\nu$.

HOMER associates a multi-label classifier h_ν with each internal node ν of the hierarchy. The task of h_ν is the prediction of one or more of the meta-labels of its children. Therefore, the set of labels for h_ν is $M_\nu = \{\mu_c \mid c \in \text{children}(\nu)\}$. Figure 1 shows a sample hierarchy produced for a multi-label classification task with 8 labels $\{\lambda_1, \ldots, \lambda_8\}$.

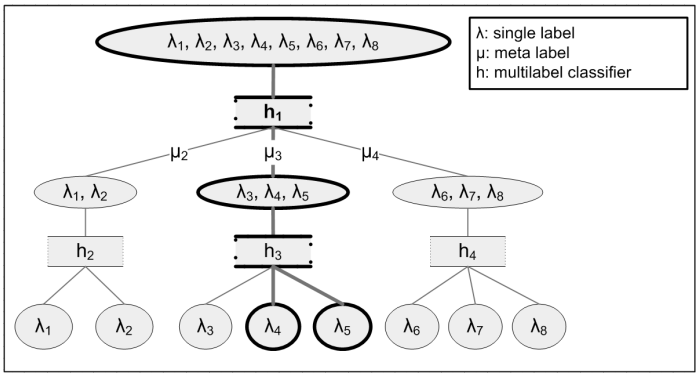


Fig. 1. Sample hierarchy for a multi-label classification task with 8 labels

For the multi-label classification of a new instance $\bar{\text{x}}$, HOMER starts with h_{root} and follows a recursive process forwarding $\bar{\text{x}}$ to the multi-label classifier h_c of a child node c only if μ_c is among the predictions of $h_{\text{parent(c)}}$. Eventually, this process may lead to the prediction of one or more single-labels by the multi-label classifier(s) just above the corresponding leaf(ves). The union of these predicted single-labels is the output of the proposed approach in this case, while the empty set is returned otherwise.

In the training phase, HOMER creates the tree recursively in a top-down depth-first fashion starting with the root. At each node ν, k child nodes are first created using a clustering algorithm (see below). In case $|L_\nu| < k$, the number of children is set to $|L_\nu|$. Each such child ν filters the data of its parent, keeping only the examples that are annotated with at least one of its own labels: $D_\nu = \{(\bar{\text{x}}_i, P_i) \mid (\bar{\text{x}}_i, P_i) \in D_{\text{parent}(\nu)}, P_i \cap L_\nu \neq \emptyset\}$. The root uses the whole training set, $D_{\text{root}} = D$. The examples in D_ν are then transformed into meta-examples $(\bar{\text{x}}_i, Z_i)$, where $Z_i = \{\mu_c \mid c \in \mathit{children}(\nu), P_i \cap L_c \neq \emptyset\}$, which are subsequently used for training h_ν.

The main issue in the former process is how to distribute the labels of $\mathcal{L}_\nu$ to the k children. We argue that labels should be *evenly* distributed to k subsets in a way such that labels belonging to the same subset are as *similar* as possible. Such a task can be thought of as clustering with the additional constrain of equal cluster size. It has been considered in the past in the literature, under the name *balanced clustering* [7]. In [2], a new balanced clustering algorithm named balanced k-means has been proposed for HOMER, which guarantees that the clusters will be of exactly the same size.

The justification for preferring similarity-based distribution is that if similar labels of a node ν are placed in the same subset, then only a few (ideally just one) meta-labels of h_ν will be predicted and thus the rest sub-trees will not be activated. This will lead to reduced cost during the operation and testing of HOMER. Another expected benefit is that each child node will probably contain less training examples. The justification for preferring an even distribution is that the multi-label classifiers at each node will deal with a more balanced distribution of positive examples for each meta-label. This is expected to lead to improved predictive performance.

We could consider HOMER as the combination of two components: a) an algorithm that constructs a hierarchy on top of the labels of a multi-label dataset, and b) a generalization of the well-known Pachinko-machine hierarchical classification algorithm [8] to the multi-label case.

In [2], HOMER, when using the well-known binary relevance classifier (BR) as the base multi-label classifier in each internal node, has proven to outperform BR in terms of quality of prediction and, especially, classification time. In this paper, we also study the use of the Calibrated Label Ranking method as a multi-label classifiers at the nodes of HOMER.

2.2 Calibrated Label Ranking and QWeighted

QCLR [5] is a recently proposed efficient approach for multi-label classification. This algorithm combines three components: a) the pairwise decomposition of multi-label problems [9], b) the calibrated label ranking [3] for determining a bipartition (multi-label result) and c) an adaption of the QWeighted algorithm [10] for efficient voting aggregation that is used for prediction.

In the pairwise binarization method for multiclass classification, one classifier is trained for each pair of classes, i.e., a problem with n different classes is decomposed into $\frac{n(n-1)}{2}$ smaller subproblems. For each pair of classes (λ_u, λ_v), only examples belonging to either λ_u or λ_v are used to train the corresponding classifier $o_{u,v}$. In the multi-label case, an example is added to the training set for classifier $o_{u,v}$ if λ_u is a relevant class and λ_v is an irrelevant class or vice versa, i.e., $(\lambda_u, \lambda_v) \in P \times N \cup N \times P$ with $N = \mathcal{L} \backslash P$ as negative label set. The pairwise binarization method is often regarded as superior to binary relevance because it profits from simpler decision boundaries in the subproblems [11, 12, 9]. The predictions of the base classifiers $o_{u,v}$ may then be interpreted as *preference statements* that predict for a given example which of the two labels λ_u or λ_v is preferred. In order to convert these binary preferences into a class ranking, we

use a simple voting strategy known as *max-wins*, which interprets each binary preference as a vote for the preferred class. Classes are then ranked according to the number of received votes. Ties are broken randomly in our case.

To convert the resulting ranking of labels into a multi-label prediction, we use the *calibrated label ranking* (CLR) approach [3]. This technique avoids the need for learning a threshold function for separating relevant from irrelevant labels, which is often performed as a post-processing phase after computing a ranking of all possible classes. The key idea is to introduce an artificial *calibration label* λ_0, which represents the split-point between relevant and irrelevant labels. Thus, it is assumed to be preferred over all irrelevant labels, but all relevant labels are preferred over λ_0. As it turns out, the resulting n additional binary classifiers $\{ o_{i,0} \mid i = 1 \dots n \}$ are identical to the classifiers that are trained by the binary relevance approach. Thus, each classifier $o_{i,0}$ is trained in a one-against-all fashion by using the whole dataset with $\{ \bar{\mathrm{x}} \mid \lambda_i \in P_{\bar{\mathrm{x}}} \} \subseteq \mathcal{X}$ as positive examples and $\{ \bar{\mathrm{x}} \mid \lambda_i \in N_{\bar{\mathrm{x}}} \} \subseteq \mathcal{X}$ as negative examples. At prediction time, we will thus get a ranking over $n + 1$ labels (the n original labels plus the calibration label).

For the multiclass case, the voting strategy can be performed efficiently with the Quick Weighted Voting algorithm (QWeighted), which computes the class with the highest accumulated voting mass without evaluating *all* pairwise classifiers. It exploits the fact that during a voting procedure some classes can be excluded from the set of possible top rank classes early on, because even if they reach the maximal voting mass in the remaining evaluations they can no longer exceed the current maximum. For example, if class λ_a has received more than $n - j$ votes and class λ_b has lost j binary votings, it is impossible for λ_b to achieve a higher total voting mass than λ_a. Thus further evaluations with λ_b can be safely ignored for the comparison of these two classes.

QWeighted can be adapted to multi-label classification by repeating the process. We can compute the top class λ_{top} using QWeighted, remove this class from $\mathcal{L}$ and repeat this step, until the returned class has fewer votes than the artificial label λ_0, which means that all remaining classes will be considered to be irrelevant. More precisely, during this procedure, we can go already to the next iteration, if the current top ranked class λ_t has accumulated more votes than the artificial label λ_0. For the multi-label classification the information that a particular class is ranked *above* the calibrated label is sufficient, and we do not need to know *by which amount*. The class λ_t is then not removed from the set of labels (opposing to the introductory sentence), because its incident classifiers $o_{t,j}$ may be still be needed for computing the votes for other classes. However, it can henceforth no longer be selected as a new top rank candidate. Self-evidently, the information about which pairwise classifiers have been evaluated and their results are carried through the iterations so that no pairwise classifier is evaluated more than once.

It is easy to see that in the testing phase, the number of base classifier evaluations for this approach for multi-label classification is bounded from above by $n + d \cdot C_{\mathrm{QW}}$, since we always evaluate the n classifiers involving the calibrated class, and have to do one iteration of QWeighted for each of the (on average)

Table 1. Name, number of examples used for training and testing, number of features and labels, label cardinality and density, and number of distinct labelsets for each dataset used in the experiments

	examples						distinct
name	train	test	features	labels	cardinality	density	labelsets
HiFind	16452	16519	98	632	37.304	0.059	32734
eccv2002	42379	4686	36	374	3.525	0.009	3175
jmlr2003	48859	16503	46	153	3.071	0.020	3115
mediamill	30993	12914	120	101	4.376	0.043	6555

d relevant labels. Assuming that QWeighted on average needs $C_{QW} = n \log n$ base classifier evaluations as suggested in [10], we can expect an average number of $n + dn \log n$ classifier evaluations. Thus, the classification effectiveness of this approach crucially depends on the average number d of relevant labels. We can expect a high reduction of pairwise comparisons if d is small compared to n, which holds for most real-world multi-label datasets.

3 Evaluation

In this section, after the presentation of the experimental setup, we will discuss the effect of the several parameters of HOMER and then compare it in terms of training time, classification time and predictive performance against its base multi-label classifiers.

3.1 Setup

We conducted experiments on four large multi-label datasets with at least 100 labels and 10000 training examples. The first one, *HiFind*, contains 32769 music titles annotated on average with 37 from 632 different labels [13]. The second dataset, *eccv2002* [14], is a popular benchmark for image classification and annotation methods. It is based on 5000 Corel images, 4500 of which are used for training and the rest 500 for testing. The third one, *jmlr2003*, is produced from the first (001) subset of the data accompanying [15]. It is based on 6932 images, 5188 of which are used to create the training set and the rest 1744 to create the test set. The last one is based on the *Mediamill* Challenge dataset [16]. It contains pre-computed low-level multimedia features from the 85 hours of international broadcast news video of the TRECVID 2005/2006 benchmark. Table 3.1 shows the number of examples used for training and testing, the number of features, the number of labels, the label cardinality and density, and the number of distinct labelsets for each dataset.

The experiments were conducted using the Mulan library of algorithms for multi-label learning [17]. As base classifier we used the decision tree learner J48 with standard settings, which is an implementation of C4.5 included in the WEKA framework [18].

The effectiveness of all algorithms is evaluated with label-based micro-averaged [19] recall, precision and F1 (the harmonic mean of recall and precision). Let $\hat{P}_i$ be the predicted label set for example $\bar{x}_i$, in contrast to the true label set P_i. The micro-averaged recall and precision are calculated as follows:

$$Recall = \frac{\sum_{i=1}^{|D|} |\hat{P}_i \cap P_i|}{\sum_{i=1}^{|D|} |\hat{P}_i|} \qquad Precision = \frac{\sum_{i=1}^{|D|} |\hat{P}_i \cap P_i|}{\sum_{i=1}^{|D|} |P_i|}$$

We also evaluate the efficiency of all algorithms based on their run time (for training and classification).

3.2 Results of HOMER with QCLR

This section presents and discusses the results of using HOMER together with QCLR as the multi-label algorithm for building models at each internal node of the hierarchy. We experimented with 8 different numbers of partitions (i.e., k ranges from 3 to 10) and 3 different methods for partitioning the set of labels at each internal node: a) random and even distribution (R) of the labels to the children nodes, b) clustering (C) using the expectation minimization (EM) algorithm (as implemented in Weka [18]), and c) balanced clustering (B) using the algorithm introduced in [2]. In addition to HOMER with QCLR[4] as multi-label classifier we ran the experiments using HOMER with BR[5] and also using the plain algorithms BR and QCLR without HOMER.

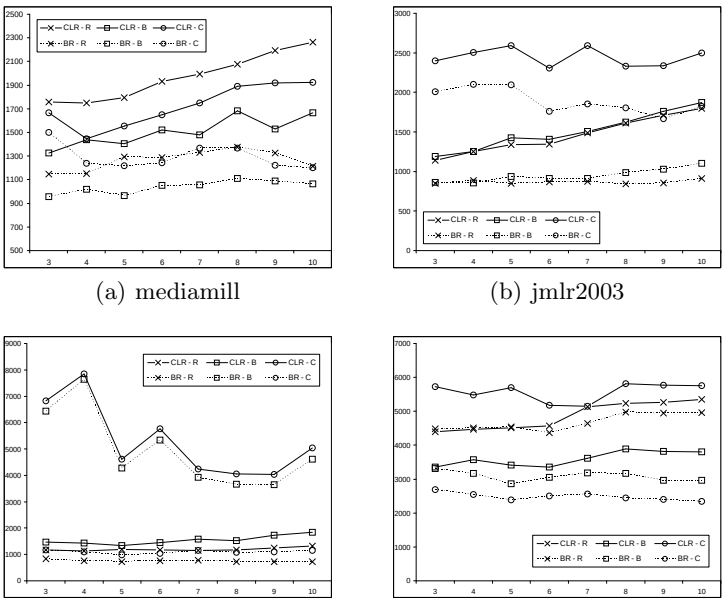
Training Time Figure 2 shows the training time of the HOMER variants in seconds. We would expect that the training time of the random partitioning variant should be less than that of the balanced clustering variant, since they both deal with the same number of labels and create and train the same number of multi-label classifiers, but balanced clustering needs some additional time to distribute the labels according to similarity as well[6]. However, this is clearly noticed only in *eccv2002*. In *jmlr2003* there is no clear winner for all numbers of partitions, while in *mediamill* and *HiFind* we notice that the balanced clustering approach requires less time, independently of the multi-label learning algorithm that is used (BR or CLR) and the number of partitions.

These results can be explained by the following observation. As clustering is based on the values of the labels, the children produced with balanced clustering, will contain labels that typically appear or do not appear together. This in turn means that more examples of the parent node will be filtered, leading to a reduced number of training examples. This was also observed in [2]. Here, we

[4] In the following graphs this combination is denoted as CLR-R, CLR-B, CLR-C respectively for all three different partitioning approaches

[5] In the following graphs this combination is denoted as BR-R, BR-B, BR-C respectively for all three different partitioning approaches

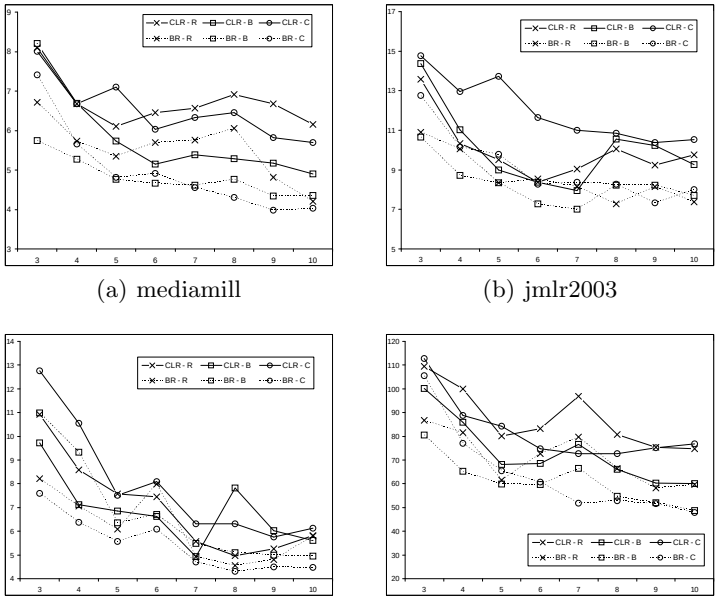
[6] Proper evaluation should separately measure the time to build the models and the time for balanced clustering. We will consider this in our future work.



(a) mediamill (b) jmlr2003

(c) eccv2002 (d) HiFind

Fig. 2. Training time over number of partitions for the six HOMER variants



(a) mediamill (b) jmlr2003

(c) eccv2002 (d) HiFind

Fig. 3. Testing time over number of partitions for the six HOMER variants

notice that the gains in training time are higher for CLR compared to BR. This is an expected result based on the previous observation, because CLR trains its binary classifiers only on those examples where the values of the corresponding labels differ.

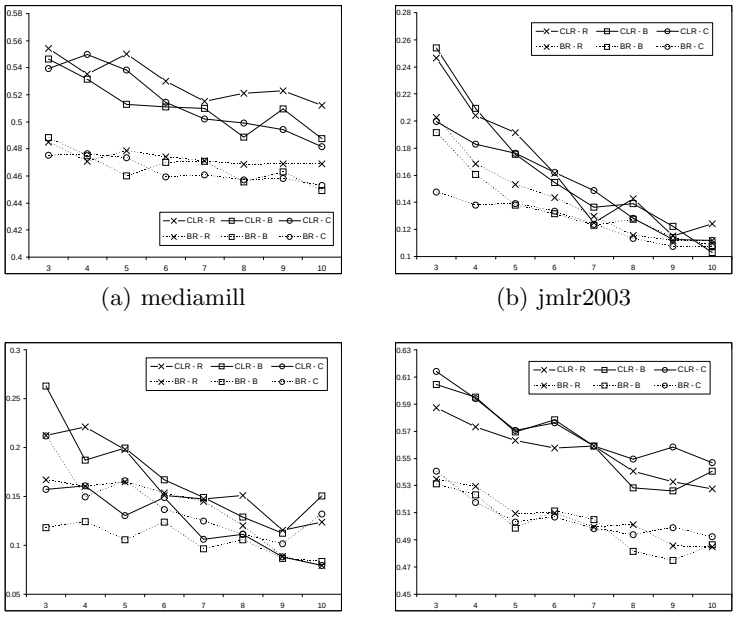
One issue that still needs to be explained, is how this behavior is affected by the different datasets. In this direction, we notice that the gains in training time seem to be correlated with the density of the dataset. The reason, again based on the previous observation, is that the lower the number of label appearances with respect to the number of labels (density), the lower the gains that can be achieved by clustering co-occurring labels together.

Concerning the plain clustering partitioning method, we notice that it is clearly the worst one in terms of training apart from the *mediamill* dataset. The plain clustering method requires more time to perform the clustering as it is based on the expectation maximization algorithm. *Mediamill* is also the smallest dataset, where it seems that the time required for clustering does not surpass the gains from the clustering process. This is why plain clustering appears to be better than random partitioning, especially for CLR. The loss in performance is more evident in *eccv2002* and *jmlr2003* due to the lower density.

Testing Time Figure 3 shows the testing times of the HOMER variants in seconds. Here the results are not as clear as in the case of the training time. Apart from the *jmlr2003* dataset, it seems that balanced clustering leads to less testing time compared to random partitioning irrespectively of the multi-label learning algorithm. Also plain clustering seems to be worse than the rest of the partitioning methods in *eccv2002* and *jmlr2003* for most of the partition numbers. Finally, we could comment that the classification time seems to decrease with respect to the number of partitions probably due the smaller height of the tree ($log_k(n)$).

Prediction Quality Figures 4 and 5 show the recall and precision results for the HOMER variants on all four datasets. We can see that recall decreases, while precision increases with the number of partitions, independently of the multi-label learner and partitioning method used. One potential reason for this behavior could be that smaller number of partitions lead to more general meta-labels that are more difficult to distinguish. Apparently this leads to a more relaxed prediction, so that at each inner node the multi-label classifier does predict more meta-labels and as a consequence more of the original labels, but with lower precision.

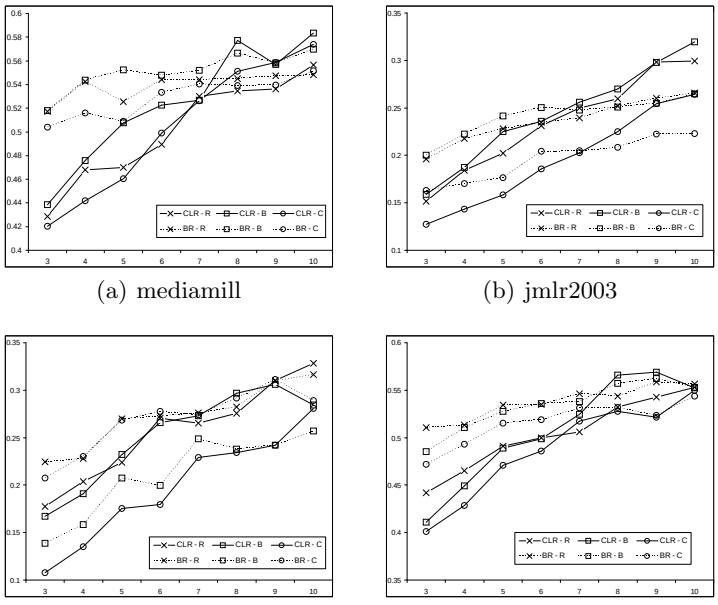
The recall of the CLR based HOMER variants seems to be larger than that of the BR based HOMER variants, irrespectively of the number of partitions. This is totally clear in *mediamill* and *HiFind*, but less clear in *jmlr2003* and *eccv2002*, though it stills holds if we compare the two learners under the same partitioning method. As far as the partitioning method is concerned, there is no clear trend with respect to recall, while the plain clustering method seems to have the worst precision for both BR and CLR based HOMER.



(a) mediamill (b) jmlr2003

(c) eccv2002 (d) HiFind

Fig. 4. Micro recall over number of partitions for the six HOMER variants



(a) mediamill (b) jmlr2003

(c) eccv2002 (d) HiFind

Fig. 5. Micro precision over number of partitions for the six HOMER variants

The decrease in recall is stronger for CLR than for BR in the low density datasets *eccv2002* and *jmlr2003*. This means that in low density datasets, a small number of partitions favors the recall of HOMER with CLR. On the other hand the increase in precision is stronger for CLR than for BR in the high density datasets *mediamill* and *HiFind*. This means that in high density datasets a large number of partitions favors the precision of HOMER. A potential reason is the fact that CLR underestimates the size of the predicted labelsets [3]. It seems that this underestimation increases with the number of labels, as seen in the results of CLR that are discussed later on in this paper.

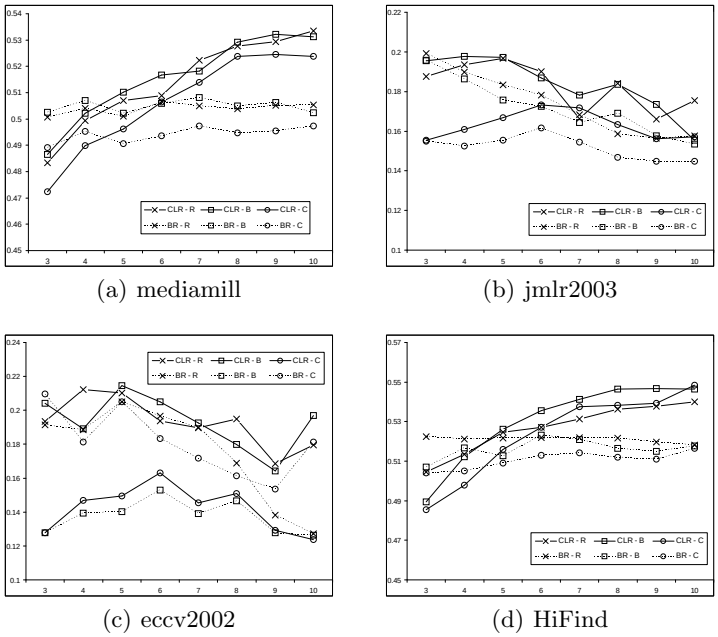
Figure 6 shows the micro-averaged F_1 measure of HOMER for the datasets. As far as BR based HOMER is concerned no clear trend can be detected with respect to the number of partitions. With respect to the partitioning method, the plain clustering approach seems inferior to the rest, while no clear winner between balanced clustering and random partitioning can be announced. As far as CLR is concerned, as already outlined in the previous paragraphs, in low density datasets we notice a decrease of F_1 with respect to the number of partitions, while in high density datasets we notice an increase of F_1. We could therefore consider this as a guideline for selecting the number of partitions for HOMER with CLR based on the density of the dataset. Overall, the CLR based HOMER seems to be achieving better results for a larger percentage of different partition numbers, compared to the BR based HOMER. In terms of the partitioning method, the plain clustering approach seems inferior to the rest for both CLR and BR.

3.3 Comparison of HOMER against its base classifiers

For the direct comparison of HOMER against the flat approaches in Table 2 we chose the configuration with balanced clustering and 10 partitions. Note that no results could be retrieved for CLR on the *HiFind* dataset due to the high memory requirements. To circumvent this problem for problems with a large number of classes was a main objective of combining HOMER with CLR as base classifier.

Prediction Quality It is especially interesting to observe the opposite behavior in terms of recall and precision of the different approaches. CLR shows the best precision performance with a large margin over the other algorithms on all datasets. On the other hand, its recall values are particularly low. This confirms previous results that CLR does underestimate the size of the predicted labelsets [3]. Our results indicate that this is particularly true for datasets with a high number of classes such as *eccv2002*, where CLR returns only 3.84% of the correct labels, while 58.11% of the returned labels are actually correct, compared to the 36.58% by BR and around 28% by both HOMERs. On the other hand, on the *mediamill* dataset, CLR's gain in precision seems to make up its low recall, thereby producing the highest average F1 value.

BR has a similar behavior of predicting relatively few labels with increasing number of labels. This is probably due to the greater imbalance of positive to



(a) mediamill (b) jmlr2003

(c) eccv2002 (d) HiFind

Fig. 6. Micro F_1 over number of partitions for the six HOMER variants

negative examples for large problems, which leads to less frequent predictions of positive examples than the class distribution would suggest. HOMER shifts the trade-off between recall and precision to a more balanced level, increasing recall but losing precision. The reason is probably the smaller problems in terms of number of classes that CLR has to solve in the HOMER setting. This was already shown in the correlative behavior between number of partitions and precision. The effect can also be seen when using BR as multi-label base classifier technique for HOMER, but it is less pronounced since the plain BR itself produces more balanced results.

Due to the great differences in recall and precision between the algorithms, we decided to omit the Hamming losses in Table 2 though this measure is usually used for evaluating multi-label algorithms, since Hamming loss generally favors algorithms with high precision and low recall,[7] which in this case means to favor CLR. The F1 measure, which returns the harmonic mean between recall and precision, allows a more commensurate analysis in this particular case since it penalizes greater differences to a higher degree. Except on the *mediamill* dataset, for which the approx. 100 classes do not show a great impact on CLR's recall, HOMER achieves the highest micro-averaged F1 value. In particular it outperforms BR on every dataset, which is especially interesting since HOMER is the

[7] Returning zero labels to returning 50 of which 25 are correct would result in the same loss.

Table 2. Performance measures and computational costs on the different datasets. Results for Binary Relevance (BR), QWeighted calibrated label ranking (QCLR), HOMER with balanced clustering and 10 partitions and BR (H+BR), HOMER with balanced clustering and 10 partitions and QCLR (H+QCLR) are shown.

Method	mediamill	jmlr2003	eccv2002	HiFind
`micro Precision`				
BR	58.00 %	32.27 %	36.58 %	59.43 %
QCLR	73.89 %	56.18 %	58.11 %	–
H+BR	56.98 %	26.48 %	28.91 %	55.31 %
H+QCLR	58.35 %	31.93 %	28.43 %	55.26 %
`micro Recall`				
BR	44.79 %	9.85 %	7.42 %	45.73 %
QCLR	43.86 %	4.57 %	3.84 %	–
H+BR	44.91 %	10.81 %	13.21 %	48.64 %
H+QCLR	48.77 %	10.28 %	15.07 %	54.06 %
`micro F1`				
BR	50.55 %	15.09 %	12.34 %	51.65 %
QCLR	55.04 %	8.45 %	7.21 %	–
H+BR	50.23 %	15.36 %	18.14 %	51.76 %
H+QCLR	53.13 %	15.55 %	19.70 %	54.65 %
`Training Time`				
BR	2413.40	2801.17	2701.32	4179.66
QCLR	7423.19	6542.51	7460.14	–
H+BR	1065.21	1101.61	1144.47	2345.39
H+QCLR	1667.29	1871.00	1836.34	3801.53
`Testing Time`				
BR	3.84	6.67	5.47	50.47
QCLR	103.59	119.28	154.65	–
H+BR	4.35	7.70	4.48	48.77
H+QCLR	4.90	9.26	5.62	60.02

direct competitor of BR in terms of computational costs. Similarly, HOMER+BR beats the plain BR except for *mediamill*, for which both algorithm are almost equal. HOMER+BR in general achieves less accurate predictions than using the pairwise approach as base classifier: in terms of F1 HOMER+BR is beaten on all datasets, in terms of recall and precision both algorithm are either almost equal (HOMER+BR slightly ahead) or HOMER+QCLR is clearly on top.

Computational Time As shown in Table 2, HOMER is able to reduce the training time in comparison to plain BR approx. between 60% and 44% for using BR as base and between 33% and 10% for using QCLR. The first comparison is especially interesting since HOMER+BR has to train more base classifiers than BR: one classifier for each class at the leafs such as BR in addition to the classifiers in the inner nodes. However, this is done obviously with less training examples due to the filtering of examples at the inner nodes. Comparing the two HOMER variants, we can observe that the overhead of training the pairwise classifiers is always less than training the one-against-all classifiers. Note that QCLR has to train the same classifiers as BR for the comparison to the calibrating artificial class plus the pairwise classifiers between real classes. This may seem very surprising since training the pairwise classifiers requires $|P|/|D|$ times more training examples than training the BR classifiers[8], i.e. the amount is multiplied by the cardinality of the multi-label problem (cf. [3]). But when the base classifier has a super-linear complexity in terms of training examples, the reduced size of the binary subproblems by the pairwise approach may lead to a reduced complexity [11], which is the case for J48. In addition, another factor could be that by clustering the cardinality of the reduced multi-label problems is often strongly reduced to the extreme case 1, where pairwise classifiers utilize in total fewer trainings examples than BR. However, we defer the investigation of this previously unseen observation for future work.

For testing, HOMER+BR is slightly slower than BR for the smaller *mediamill* and *jmlr2003*, but for the greater datasets *eccv2002* and *HiFind* it requires less time. Although HOMER+BR has trained more base classifiers than the plain BR approach, it may invoke less base classifiers since great part of the classifier tree is pruned each time a meta-label is predicted as negative. This effect was already observed in previous work on a dataset with almost 1000 classes [2]. HOMER+QCLR spends between 3% and 40% more time than BR, however, testing costs are so small for J48 compared to training time that this increase is almost not noticeable. Again, the overhead for evaluating the additional pairwise classifiers in HOMER+QCLR only require a small fraction of the time needed for the BR classifiers. Nevertheless it is not possible to simply compute the overhead as difference between the time for HOMER+BR and +QCLR since prediction accuracy, especially precision, also strongly influences the classification time. As expected, CLR requires the most computations for learning and

[8] This estimation of training examples is too rough for the special case $|P|/|D| = 1$, which refers to a reduction to a multiclass problem. In this regard, the pairwise classifiers use in total fewer examples than the One-Against-All classifiers [11].

predicting. However, the factor in training costs is proportional to the average label set size per example, which makes the costs acceptable for most of the multi-label problems since the label sets tend to be small. For prediction, the usage of QWeighted is able to considerably reduce the costs in comparison to the evaluation of all pairwise base classifier while maintaining the advantage of the pairwise approach in terms of predictive performance.

4 Conclusions

This paper performed an empirical study of the performance of HOMER. Compared to previous work [2], the experimental part examines an additional multi-label learner for training each node of the hierarchy (QWeighted calibrated label ranking) on four large multi-label datasets with a variety of characteristics. Interestingly, the results showed that the instantiation of the multi-label learner of HOMER to QCLR can lead to better results compared to instantiating it to BR at a small expense in training and classification time. HOMER improves the training time of BR and this is even more important for QCLR. In terms of classification time HOMER substantially improves QCLR, while for BR the benefits appear for the two largest datasets in terms of number of labels. Except for the *mediamill* dataset (where the differences are rather small), HOMER managed to improve the performance of the base multi-label learner (both BR and QCLR).

HOMER also deals with the scalability problem of QCLR in terms of memory with respect to the number of labels, since it substantially reduces the amount of needed classifiers. In the same manner it provides a significant reduction in training and test time for the pairwise CLR methods. It is also shown that HOMER is able to equilibrate recall and precision, especially for QCLR which seems to underestimate the number of labels per instance for problems with a high number of labels.

Acknowledgements We gratefully acknowledge support from the partnership programs IKYDA of the State Scholarship Foundation of Greece and PPP of the German Academic Exchange Service (DAAD). Sang-Hyeun Park is supported by the German Science Foundation (DFG).

References

1. Tsoumakas, G., Katakis, I.: Multi-label classification: An overview. International Journal of Data Warehousing and Mining **3**(3) (2007) 1–13
2. Tsoumakas, G., Katakis, I., Vlahavas, I.: Effective and efficient multilabel classification in domains with large number of labels. In: Proceedings of the ECML/PKDD-08 Workshop on Mining Multidimensional Data (MMD-08), Antwerp, Belgium (2008) 30–44
3. Fürnkranz, J., Hüllermeier, E., Loza Mencía, E., Brinker, K.: Multilabel classification via calibrated label ranking. Machine Learning **73**(2) (2008) 133–153

4. Hüllermeier, E., Fürnkranz, J., Cheng, W., Brinker, K.: Label ranking by learning pairwise preferences. Artificial Intelligence **172** (2008) 1897–1916
5. Loza Mencía, E., Park, S.H., Fürnkranz, J.: Efficient voting prediction for pairwise multilabel classification. In: Proceedings of the 11th European Symposium on Artificial Neural Networks (ESANN-09), Springer-Verlag (2009)
6. Loza Mencía, E., Fürnkranz, J.: Efficient pairwise multilabel classification for large-scale problems in the legal domain. In: Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Disocvery in Databases (ECML-PKDD-2008), Part II, Antwerp, Belgium (2008) 50–65
7. Banerjee, A., Ghosh, J.: Scalable clustering algorithms with balancing constraints. Data Mining and Knowledge Discovery **13**(3) (2006) 365–395
8. Koller, D., Sahami, M.: Hierarchically classifying documents using very few words. In: Proceedings of the 14th International Conference on Machine Learning (ICML-97), Nashville (1997) 170–178
9. Loza Mencía, E., Fürnkranz, J.: Pairwise learning of multilabel classifications with perceptrons. In: Proceedings of the 2008 IEEE International Joint Conference on Neural Networks (IJCNN 08), Hong Kong (2008) 2900–2907
10. Park, S.H., Fürnkranz, J.: Efficient pairwise classification. In: Proceedings of 18th European Conference on Machine Learning (ECML-07), Warsaw, Poland (2007) 658–665
11. Fürnkranz, J.: Round Robin Classification. Journal of Machine Learning Research **2** (2002) 721–747
12. Hsu, C.W., Lin, C.J.: A Comparison of Methods for Multi-class Support Vector Machines. IEEE Transactions on Neural Networks **13**(2) (2002) 415–425
13. Pachet, F., Roy, P.: Improving multilabel analysis of music titles: A large-scale validation of the correction approach. IEEE Transactions on Audio, Speech, and Language Processing **17**(2) (2009) 335–343
14. Duygulu, P., Barnard, K., de Freitas, N., Forsyth, D.: Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In: 7th European Conference on Computer Vision. (2002) (IV):97–112
15. Barnard, K., Duygulu, P., de Freitas, N., Forsyth, D., Blei, D., Jordan, M.I.: Matching words and pictures. Journal of Machine Learning Research **3** (2003) 1107–1135
16. Snoek, C.G.M., Worring, M., van Gemert, J.C., Geusebroek, J.M., Smeulders, A.W.M.: The challenge problem for automated detection of 101 semantic concepts in multimedia. In: MULTIMEDIA '06: Proceedings of the 14th annual ACM international conference on Multimedia, New York, NY, USA, ACM (2006) 421–430
17. Tsoumakas, G., Vlahavas, I.: Random k-labelsets: An ensemble method for multilabel classification. In: Proceedings of the 18th European Conference on Machine Learning (ECML 2007), Warsaw, Poland (September 17-21 2007) 406–417
18. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. 2 edn. Morgan Kaufmann, San Francisco (2005)
19. Yang, Y.: An evaluation of statistical approaches to text categorization. Information Retrieval **1** (1999) 67–88