JMLR: Workshop and Conference Proceedings 29:404–418, 2013 ACML 2013

Multilabel Classification through Random Graph Ensembles

Hongyu Su HONGYU.SU@AALTO.FI
Juho Rousu JUHO.ROUSU@AALTO.FI
Helsinki Institute of Information Technology (HIIT)
Department of Information and Computer Science
Aalto University, Konemiehentie 2, 02150 Espoo, Finland

Editor: Cheng Soon Ong and Tu Bao Ho

Abstract

We present new methods for multilabel classification, relying on ensemble learning on a collection of random output graphs imposed on the multilabel and a kernel-based structured output learner as the base classifier. For ensemble learning, differences among the output graphs provide the required base classifier diversity and lead to improved performance in the increasing size of the ensemble. We study different methods of forming the ensemble prediction, including majority voting and two methods that perform inferences over the graph structures before or after combining the base models into the ensemble. We compare the methods against the state-of-the-art machine learning approaches on a set of heterogeneous multilabel benchmark problems, including multilabel AdaBoost, convex multitask feature learning, as well as single target learning approaches represented by Bagging and SVM. In our experiments, the random graph ensembles are very competitive and robust, ranking first or second on most of the datasets. Overall, our results show that random graph ensembles are viable alternatives to flat multilabel and multitask learners.

Keywords: multilabel classification; structured output; ensemble methods; kernel methods; graphical models

1. Introduction

Multilabel and multitask classification rely on representations and learning methods that allow us to leverage the dependencies between the different labels. When such dependencies are given in form of a graph structure such as a sequence, a hierarchy or a network, structured output prediction (Taskar et al., 2003; Tsochantaridis et al., 2004; Rousu et al., 2006) becomes a viable option, and has achieved a remarkable success. For multilabel classification, limiting the applicability of the structured output prediction methods is the very fact they require the predefined output structure to be at hand, or alternatively auxiliary data where the structure can be learned from. When these are not available, flat multilabel learners or collections of single target classifiers are thus often resorted to.

In this paper, we study a different approach, namely using ensembles of graph labeling classifiers, trained on randomly generated output graph structures. The methods are based on the idea that variation in the graph structure shifts the inductive bias of the base learners and causes diversity in the predicted multilabels. Each base learner, on the other hand, is trained to predict as good as possible multilabels, which make them satisfy the weak learning assumption, necessary for successful ensemble learning.

Ensembles of multitask or multilabel classifiers have been proposed, but with important differences. The first group of methods, boosting type, rely on changing the weights of the training instances so that difficult to classify instances gradually receive more and more weights. The AdaBoost boosting framework has spawned multilabel variants (Schapire and Singer, 2000; Esuli et al., 2008). In these methods the multilabel is considered essentially as a flat vector. The second group of methods, Bagging, are based on bootstrap sampling the training set several times and building the base classifiers from the bootstrap samples. Thirdly, randomization has been used as the means of achieving diversity by Yan et al. (2007) who select different random subsets of input features and examples to induce the base classifiers, and by Su and Rousu (2011) who use majority voting over random graphs in drug bioactivity prediction context. Here we extend the last approach to two other types of ensembles and a wider set of applications, with gain in prediction performances.

The remainder of the article is structured as follows. In section 2 we present the structured output model used as the graph labeling base classifier. In Section 3 we present three multilabel ensemble learning methods based on the random graph labeling. In section 4 we present empirical evaluation of the methods. In section 5 we present concluding remarks.

2. Multilabel classification through graph labeling

We examine the following multilabel classification setting. We assume data from a domain $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is a set and $\mathcal{Y} = \mathcal{Y}_1 \times \cdots \times \mathcal{Y}_k$ is the set of multilabels, represented by a Cartesian product of the sets $\mathcal{Y}_j = \{1, \ldots, l_j\}, j = 1, \ldots, k$. A vector $\mathbf{y} = (y_1, \ldots, y_k) \in \mathcal{Y}$ is called the *multilabel* and the components y_j are called the *microlabels*. We assume that a training set $\{(x_i, \mathbf{y}_i)\}_{i=1}^m \subset \mathcal{X} \times \mathcal{Y}$ has been given. A pair $(x_i, \mathbf{y})$ where x_i is a training pattern and $\mathbf{y} \in \mathcal{Y}$ is arbitrary, is called a *pseudo-example*, to denote the fact that the pair may or may not be generated by the distribution generating the training examples. The goal is to learn a model $F : \mathcal{X} \mapsto \mathcal{Y}$ so that the expected loss over predictions on future instances is minimized, where the loss is chosen suitably for multilabel learning problems. By $\mathbf{1}_{\{\cdot\}}$ we denote the indicator function $\mathbf{1}_{\{A\}} = 1$, if A is true, $\mathbf{1}_{\{A\}} = 0$ otherwise.

Here, we consider solving multilabel classification with graph labeling classifiers that, in addition to the training set, assume a graph $G = (V, E)$ with nodes $V = \{1, \ldots, k\}$ corresponding to microlabels and edges $E \subset V \times V$ denoting potential dependencies between the microlabels. For an edge $e = (j, j') \in E$, by $\mathbf{y}_e = (y_j, y_{j'})$ we denote the edge label of e in multilabel $\mathbf{y}$, induced by concatenating the microlabels corresponding to end points of e, with corresponding domain of edge labelings $\mathcal{Y}_e = \mathcal{Y}_j \times \mathcal{Y}_{j'}$. By $\mathbf{y}_{ie}$ we denote the label of the edge e in the i'th training example. We also denote by u_j the possible label of node j, and by $\mathbf{u}_e$ the possible label of edge e. Naturally, $u_j \in \mathcal{Y}_j$ and $\mathbf{u}_e \in \mathcal{Y}_e$. See supplementary material for a complete list of notations.

2.1. Graph labeling classifier

As the graph labeling classifier in this work we use max-margin structured prediction, which aims to learn a compatibility score

$$\psi(x, \mathbf{y}) = \langle w, \varphi(x, \mathbf{y}) \rangle = \sum_{e \in E} \langle w_e, \varphi_e(x, \mathbf{y}_e) \rangle = \sum_{e \in E} \psi_e(x, \mathbf{y}_e) \tag{1}$$

between an input x and a multilabel $\mathbf{y}$, where by $\langle \cdot, \cdot \rangle$ we denote the inner product and $\psi_e(x, \mathbf{y}_e)$ is a shorthand for the compatibility score, or potential, between an edge label $\mathbf{y}_e$ and the object x. The joint feature map

$$\varphi(x, \mathbf{y}) = \phi(x) \otimes \Upsilon(\mathbf{y}) = \phi(x) \otimes (\Upsilon_e(\mathbf{y}_e))_{e \in E} = (\varphi_e(x, \mathbf{y}_e))_{e \in E}$$

is given by a tensor product of an input feature $\phi(x)$ and the feature space embedding of the multilabel $\Upsilon(\mathbf{y}) = (\Upsilon_e(\mathbf{y}_e))_{e \in E}$, consisting of edge labeling indicators $\Upsilon_e(\mathbf{y}_e) = \left(\mathbf{1}_{\{\mathbf{y}_e = \mathbf{u}_e\}}\right)_{\mathbf{u}_e \in \mathcal{Y}_e}$. The benefit of the tensor product representation is that context (edge labeling) sensitive weights can be learned for input features and no prior alignment of input and output features needs to be assumed.

The parameters of the model are learned through max-margin optimization, where the primal optimization problem takes the form (e.g. Taskar et al., 2003; Tsochantaridis et al., 2004; Rousu et al., 2006)

$$\begin{aligned} &\min_{w} \ \frac{1}{2}||w||^2 + C \sum_{i=1}^{m} \xi_i \\ &\text{s.t. } \langle w, \varphi(x_i, \mathbf{y}_i) \rangle \geq \operatorname*{\mathbf{argmax}}_{\mathbf{y} \in \mathcal{Y}} \left(\langle w, \varphi(x_i, \mathbf{y}) \rangle + \ell(\mathbf{y}_i, \mathbf{y}) \right) - \xi_i, \\ &\qquad \text{for } i = 1, \ldots, m \end{aligned} \tag{2}$$

where w contains the weights to be learned, ξ_i denotes the slack allotted to each example, $\ell(\mathbf{y}_i, \mathbf{y})$ is the loss between pseudo-labeling and correct labeling and C is the slack parameter that controls the amount of regularization in the model. The primal form can be interpreted as maximizing the loss-scaled margin between the correct training example and incorrect pseudo-examples. The Lagrangian dual form of (2) is given as

$$\begin{aligned} &\max_{\alpha \geq 0} \ \alpha^T \ell - \frac{1}{2} \alpha^T K \alpha \\ &\qquad \text{s.t.} \sum_{\mathbf{y}} \alpha(i, \mathbf{y}) \leq C, \forall i = 1, \ldots, m \text{ and } \mathbf{y} \in \mathcal{Y}, \end{aligned} \tag{3}$$

where $\alpha = (\alpha(i, \mathbf{y}))_{i, \mathbf{y}}$ denotes the dual variables and $\ell = (\ell(\mathbf{y}_i, \mathbf{y}))_{i, \mathbf{y}}$ the loss for each pseudo-example $(x_i, \mathbf{y})$. The joint kernel

$$\begin{aligned} K(x_i, \mathbf{y}; x_j, \mathbf{y}') &= \langle \varphi(x_i, \mathbf{y}_i) - \varphi(x_i, \mathbf{y}), \varphi(x_j, \mathbf{y}_j) - \varphi(x_j, \mathbf{y}') \rangle \\ &= \langle \phi(x_i), \phi(x_j) \rangle_\phi \cdot \langle (\Upsilon(\mathbf{y}_i) - \Upsilon(\mathbf{y}), \Upsilon(\mathbf{y}_j) - \Upsilon(\mathbf{y}') \rangle_\Upsilon \\ &= K_\phi(x_i, x_j) \cdot \left(K_\Upsilon(\mathbf{y}_i, \mathbf{y}_j) - K_\Upsilon(\mathbf{y}_i, \mathbf{y}') - K_\Upsilon(\mathbf{y}, \mathbf{y}_j) + K_\Upsilon(\mathbf{y}, \mathbf{y}') \right) \end{aligned}$$

is composed by product of input $K_\phi(x_i, x_j) = \langle x_i, x_j \rangle_\phi$ and output $K_\Upsilon(\mathbf{y}, \mathbf{y}') = \langle \mathbf{y}', \mathbf{y} \rangle_\Upsilon = \sum_e K_{\Upsilon, e}(\mathbf{y}_e, \mathbf{y}'_e)$ kernels, with $K_{\Upsilon, e}(u, u') = \langle \Upsilon_e(u), \Upsilon_e(u') \rangle_\Upsilon$.

2.2. Factorized dual form

The model (3) is transformed to the factorized dual form, where the edge-marginals of dual variables are used in place of the original dual variables

$$\mu(i, e, \mathbf{u}_e) = \sum_{\mathbf{y} \in \mathcal{Y}} \mathbf{1}_{\{\Upsilon_e(\mathbf{y}) = \mathbf{u}_e\}} \alpha(i, \mathbf{y}), \tag{4}$$

where $e = (j, j') \in E$ is an edge in the output network and $\mathbf{u}_e \in \mathcal{Y}_j \times \mathcal{Y}_{j'}$ is a possible labeling for the edge (j, j'). Using the factorized dual representation, we can state the dual problem (3) in equivalent form (c.f. Taskar et al., 2003; Rousu et al., 2007) as

$$\max_{\mu \in \mathcal{M}} \mu^T \ell - \frac{1}{2} \mu^T K_{\mathcal{M}} \mu, \tag{5}$$

where $\ell = \left(\mathbf{1}_{\{\mathbf{y}_{ie} \neq \mathbf{u}_e\}}\right)_{i,e,\mathbf{u}_e}$ is the vector of losses between the edge-labelings, and $\mu = (\mu(i, e, \mathbf{u}_e))_{i,e,\mathbf{u}_e} \in \mathcal{M}$ is the vector of marginal dual variables lying in the marginal polytope (c.f. Wainwright et al., 2005)

$$\mathcal{M} = \{\mu | \exists \alpha \text{ s.t. } \mu(i, e, \mathbf{u}_e) = \sum_{\mathbf{y} \in \mathcal{Y}} \mathbf{1}_{\{\mathbf{y}_{ie} = \mathbf{u}_e\}} \alpha(i, \mathbf{y}), \forall i, \mathbf{u}_e, e\}$$

of the dual variables, the set of all combinations of marginal dual variables (4) of a training examples that correspond to some α in the original dual feasible set in (3). The factorized joint kernel is given by $K_{\mathcal{M}} = diag(K_{\varphi,e})_{e \in E}$, where

$$\begin{aligned} &K_{\varphi,e}(x_i, \mathbf{y}_e; x_j, \mathbf{y}'_e) \\ &\qquad = K_\phi(x_i, x_j) \cdot \left(K_{\Upsilon,e}(\mathbf{y}_{ie}, \mathbf{y}_{je}) - K_{\Upsilon,e}(\mathbf{y}_{ie}, \mathbf{y}_e) - K_{\Upsilon,e}(\mathbf{y}_e, \mathbf{y}_{je}) + K_{\Upsilon,e}(\mathbf{y}_e, \mathbf{y}'_e)\right) \end{aligned}$$

containing the joint kernel values pertaining to the edge e.

The factorized dual problem (5) is a quadratic program with a number of variables *linear* in both the size of the output network and the number of training examples. There is an exponential reduction in the number of dual variables from the original dual (3), however, with the penalty of more complex feasible polytope. For solving (5) we use MMCRF (Rousu et al., 2007) that relies on a conditional gradient method. Update directions are found in linear time via probabilistic inference, making use of the the exact correspondence of maximum margin violating multilabel in the primal (2) and steepest feasible gradient of the dual objective (3).

2.3. Inference

With the factorized dual, the compatibility score of labeling an edge e as $\mathbf{y}_e$ given input x can be expressed in terms of kernels and marginal dual variables as shown by the following lemma.

Lemma 1 *Let w be the solution to (2), $\varphi(x, \mathbf{y})$ be the joint feature map, and let $G = (E, V)$ be the graph defining the output graph structure, and let us denote*

$$H_e(i, \mathbf{u}_e; x, \mathbf{y}_e) = K_\phi(x, x_i) \cdot \left(K_{\Upsilon,e}(y_{ie}, \mathbf{y}_e) - K_{\Upsilon,e}(\mathbf{u}_e, \mathbf{y}_e)\right).$$

Then, we have

$$\psi_e(x, \mathbf{y}_e) = \langle w_e, \varphi_e(x, \mathbf{y}_e) \rangle = \sum_{i, \mathbf{u}_e} \mu(i, e, \mathbf{u}_e) \cdot H_e(i, \mathbf{u}_e; x, \mathbf{y}_e),$$

where μ is the marginal dual variable learned by solving optimization problem (5).

Proof See supplementary material. ■

Consequently, the inference problem can be solved in the factorized dual by

$$\begin{aligned}\hat{\mathbf{y}}(x) =& \mathbf{argmax}_{\mathbf{y}\in\mathcal{Y}} \sum_e \psi_e(x, \mathbf{y}_e) = \mathbf{argmax}_{\mathbf{y}\in\mathcal{Y}} \sum_e \langle \mathbf{w}_e, \varphi_e(x, \mathbf{y}_e)\rangle \\ =& \mathbf{argmax}_{\mathbf{y}\in\mathcal{Y}} \sum_{e,i,\mathbf{u}_e} \mu(i, e, \mathbf{u}_e) H_e(i, \mathbf{u}_e; x, \mathbf{y}_e).\end{aligned} \tag{6}$$

The inference problem (6) is used not only in prediction phase to output multilabel $\hat{\mathbf{y}}$ that is compatible with input x, but also in model training to find the pseudo-example $\mathbf{y}$ that violates margin maximally. To solve (6), any commonly used inference technique can be used. In this paper we use MMCRF that relies on the message-passing method, also referred as loopy belief propagation (LBP). We use early stopping in inference of LBP, so that the number of iterations is limited by the diameter of the output graph G.

3. Learning graph labeling ensembles

In this section we consider generating ensembles of multilabel classifiers, where each base model is a graph labeling classifier. Algorithm 1 depicts the general form of the learning approach. We assume a function to output a random graph $G^{(t)}$ for each stage of the ensemble, a base learner to learn the graph labeling model $F^{(t)}(\cdot)$, and an aggregation function $A(\cdot)$ to compose the ensemble model. The prediction of the model is then obtained by aggregating the base model predictions

$$F(x) = A(F^{(1)}(x), \ldots, F^{(T)}(x)).$$

Given a set of base models trained on different graph structures we expect the predicted labels of the ensemble have diversity which is known to be necessary for ensemble learning. At the same time, since the graph labeling classifiers aim to learn accurate multilabels, we expect the individual base classifiers to be reasonably accurate, irrespective of the slight changes in the underlying graphs. Indeed, in this work we use randomly generated graphs to emphasize this point. We consider the following three aggregation methods:

- In *majority voting ensemble*, each base learner gives a prediction of the multilabel. The ensemble prediction is obtained by taking the most frequent value for each microlabel. Majority voting aggregation is admissible for any multilabel classifier.

Algorithm 1 Graph Labeling Ensemble Learning

Input: Training sample $S = \{(x_i, \mathbf{y}_i)\}_{i=1}^m$, ensemble size T, graph generating oracle function $outputGraph : t \in \{1, \ldots, T\} \mapsto \mathcal{G}_k$, aggregation function $A(\cdot) : \mathcal{F} \times \cdots \times \mathcal{F} \mapsto \mathcal{Y}$

Output: Multilabel classification ensemble $F(\cdot) : \mathcal{X} \mapsto \mathcal{Y}$

1: **for** $t \in \{1, \ldots, T\}$ **do**
2: $\quad G^{(t)} = \text{outputGraph}(t)$
3: $\quad F^t(\cdot) = \text{learnGraphLabelingClassifier}((x_i)_{i=1}^m, (\mathbf{y}_i)_{i=1}^m, G^{(t)})$
4: **end for**
5: $F(\cdot) = A(F^{(1)}(\cdot), \ldots, F^{(T)}(\cdot))$

Second, we consider two aggregation strategies that assume the base classifier has a conditional random field structure:

- In *average-of-maximum-marginals aggregation*, each base learner infers local maximum marginal scores for each microlabel. The ensemble prediction is taken as the value with highest average local score.
- In *maximum-of-average-marginals aggregation*, the local edge potentials of each base model are first averaged over the ensemble and maximum global marginal scores are inferred from the averages.

In the following, we detail the above aggregation strategies.

3.1. Majority voting ensemble (MVE)

The first ensemble model we consider is the majority voting ensemble (MVE), which was introduced in drug prediction context by Su and Rousu (2011). In MVE, the ensemble prediction or each microlabel is the most frequently appearing prediction among the base classifiers

$$F_j^{\mathrm{MVE}}(x) = \mathbf{argmax}_{y_j \in \mathcal{Y}_j} \left(\frac{1}{T} \sum_{i=1}^{T} \mathbf{1}_{\{F_j^{(t)}(x) = y_j\}} \right),$$

where $F^{(t)}(x) = (F_j^{(t)}(x))_{j=1}^k$ is the predicted multilabel in t'th base classifier. When using (5) as the base classifier, predictions $F^{(t)}(x)$ are obtained via solving the inference problem (6). We note, however, in principle, any multilabel learner will fit into the MVE framework as long as it adapts to a collection of output graphs $\mathcal{G} = \{G^{(1)}, \cdots, G^{(T)}\}$ and generates multilabel predictions accordingly from each graph.

3.2. Average of Max-Marginal Aggregation (AMM)

Next, we consider an ensemble model where we perform inference over the graph to extract information on the learned compatibility scores in each base models. Thus, we assume that we have access to the compatibility scores between the inputs and edge labelings

$$\Psi_E^{(t)}(x) = (\psi_e^{(t)}(x, \mathbf{u}_e))_{e \in E^{(t)}, \mathbf{u}_e \in \mathcal{Y}_e}.$$

In the Average of Max-Marginals (AMM) model, our goal is to infer for each microlabel u of each node j its *max-marginal* (Wainwright et al., 2005), that is, the maximum score of a multilabel that is consistent with $y_j = u_j$

$$\tilde{\psi}_j(x, u_j) = \max_{\{\mathbf{y} \in \mathcal{Y} : y_j = u_j\}} \sum_e \psi_e(x, \mathbf{y}_e). \tag{7}$$

One readily sees (7) as a variant of the inference problem (6), with similar solution techniques. The maximization operation fixes the labeling of the node $y_j = u_j$ and queries the optimal configuration for the remaining part of output graph. In message-passing algorithms, only slight modification is needed to make sure that only the messages consistent

with the microlabel restriction are considered. To obtain the vector $\tilde{\Psi}(x) = (\tilde{\psi}_j(x, u_j))_{j,u_j}$ the same inference is repeated for each target-microlabel pair (j, u_j), hence it has quadratic time complexity in the number of edges in the output graph.

Given the max-marginals of the base models, the Average of Max-Marginals (AMM) ensemble is constructed as follows. Let $\mathcal{G} = \{G^{(1)}, \cdots, G^{(T)}\}$ be a set of output graphs, and let $\{\tilde{\Psi}^{(1)}(x), \cdots, \tilde{\Psi}^{(T)}(x)\}$ be the max-marginal vectors of the base classifiers trained on the output graphs. The ensemble prediction for each target is obtained by averaging the max-marginals of the base models and choosing the maximizing microlabel for the node:

$$F_j^{\text{AMM}}(x) = \operatorname*{\mathbf{argmax}}_{u_j \in \mathcal{Y}_j} \frac{1}{|T|} \sum_{t=1}^{T} \tilde{\psi}_{j,u_j}^{(t)}(x),$$

and the predicted multilabel is composed from the predicted microlabels

$$F^{\text{AMM}}(x) = \left(F_j^{\text{AMM}}(x)\right)_{j \in V}.$$

In principle, AMM ensemble can give different predictions compared to MVE, since the most frequent label may not be the ensemble prediction if it has lower average max-marginal score.

3.3. Maximum Average Marginals aggregation (MAM)

The next model, the Maximum of Average Marginals (MAM) ensemble, first collects the local compatibility scores $\Psi_E^{(t)}(x)$ from individual base learners, averages them and finally performs inference on the global consensus graph with averaged edge potentials. The model is defined as

$$F^{\text{MAM}}(x) = \operatorname*{\mathbf{argmax}}_{\mathbf{y} \in \mathcal{Y}} \sum_{e \in E_t} \frac{1}{T} \sum_{t=1}^{T} \psi_e^{(t)}(x, \mathbf{y}_e) = \operatorname*{\mathbf{argmax}}_{\mathbf{y} \in \mathcal{Y}} \frac{1}{T} \sum_{t=1}^{T} \sum_{e} \langle \mathbf{w}_e^{(t)}, \varphi_e(x, \mathbf{y}_e) \rangle.$$

With the factorized dual representation, this ensemble scheme can be implemented simply and efficiently in terms of marginal dual variables and the associated kernels. Using the Lemma (1) the above can be equivalently expressed as

$$\begin{aligned} F^{\text{MAM}}(x) &= \operatorname*{\mathbf{argmax}}_{\mathbf{y} \in \mathcal{Y}} \frac{1}{T} \sum_{t=1}^{T} \sum_{i,e,\mathbf{u}_e} \mu^{(t)}(i, e, \mathbf{u}_e) \cdot H_e(i, \mathbf{u}_e; x, \mathbf{y}_e) \\ &= \operatorname*{\mathbf{argmax}}_{\mathbf{y} \in \mathcal{Y}} \sum_{i,e,\mathbf{u}_e} \bar{\mu}(i, e, \mathbf{u}_e) H_e(i, \mathbf{u}_e; x, \mathbf{y}_e), \end{aligned}$$

where we denote by $\bar{\mu}(i, e, \mathbf{u}_e) = \frac{1}{T} \sum_{t=1}^{T} \mu^{(t)}(i, e, \mathbf{u}_e)$ the marginal dual variable averaged over the ensemble. We note that $\mu^{(t)}$ is originally defined on edge set $E^{(t)}$, $\mu^{(t)}$ from different random graph are not mutually consistent. In practice, we first construct a consensus graph $\tilde{G} = (\tilde{E}, V)$ by pooling edge sets $E^{(t)}$, then complete $\mu^{(t)}$ on $\tilde{E}$ where missing components are computed via local consistency constraints. Thus, the ensemble prediction can be computed in marginal dual form without explicit access to input features, and the only input needed from the different base models are the values of the marginal dual variables.

3.4. The MAM Ensemble Analysis

Here, we present theoretical analysis of the improvement of the MAM ensemble over the mean of the base classifiers. The analysis follows the spirit of the single-label ensemble analysis by Brown and Kuncheva (2010), generalizing it to multilabel MAM ensemble.

Assume there is a collection of T individual base learners, indexed by $t \in \{1, \cdots, T\}$, that output compatibility scores $\psi_e^{(t)}(x, \mathbf{u}_e)$ for all $t \in \{1, \ldots, T\}$, $e \in E^{(t)}$, and $\mathbf{u}_e \in \mathcal{Y}_e$. For the purposes of this analysis, we express the compatibility scores in terms of the nodes (microlabels) instead of the edges and their labelings. We denote by

$$\psi_j(x, y_j) = \sum_{\substack{e=(j,j'),\\ e \in N(j)}} \mathbf{1}_{\{y_j = u_j\}} \frac{1}{2} \psi_e(x, \mathbf{u}_e)$$

the sum of compatibility scores of the set of edges $N(j)$ incident to node j with consistent labeling $\mathbf{y}_e = (y_j, y_{j'}), y_j = u_j$. Then, the compatibility score for the input and the multilabel in (1) can be alternatively expressed as

$$\psi(x, \mathbf{y}) = \sum_{e \in E} \psi_e(x, \mathbf{y}_e) = \sum_{j \in V} \psi_j(x, y_j).$$

The compatibility score from MAM ensemble can be similarly represented in terms of the nodes by

$$\psi^{\text{MAM}}(x, \mathbf{y}) = \frac{1}{T} \sum_t \psi^{(t)}(x, \mathbf{y}) = \sum_{e \in E} \bar{\psi}_e(x, \mathbf{y}_e) = \sum_{j \in V} \bar{\psi}_j(x, y_j),$$

where we have denoted $\bar{\psi}_j(x, y_j) = \frac{1}{T} \sum_t \psi_j^{(t)}(x, y_j)$ and $\bar{\psi}_e(x, \mathbf{y}_e) = \frac{1}{T} \sum_t \psi_e^{(t)}(x, \mathbf{y}_e)$.

Assume now the ground truth, the optimal compatibility score of an example and multilabel pair $(x, \mathbf{y})$, is given by $\psi^*(x, \mathbf{y}) = \sum_{j \in V} \psi_j^*(x, y_j)$. We study the reconstruction error of the compatibility score distribution, given by the squared distance of the estimated score distributions from the ensemble and the ground truth. The reconstruction error of the MAM ensemble can be expressed as

$$\Delta^R_{\text{MAM}}(x, \mathbf{y}) = \left(\psi^*(x, \mathbf{y}) - \psi^{\text{MAM}}(x, \mathbf{y})\right)^2,$$

and the average reconstruction error of the base learners can be expressed as

$$\Delta^R_I(x, \mathbf{y}) = \frac{1}{T} \sum_t \left(\psi^*(x, \mathbf{y}) - \psi^{(t)}(x, \mathbf{y})\right)^2.$$

We denote by $\Psi_j(x, y_j)$ a random variable of the compatibility scores obtained by the base learners and $\{\psi_j^{(1)}(x, y_j), \cdots, \psi_j^{(T)}(x, y_j)\}$ as a sample from its distribution. We have the following result:

Theorem 1 *The reconstruction error of compatibility score distribution given by MAM ensemble $\Delta^R_{MAM}(x, \mathbf{y})$ is guaranteed to be no greater than the average reconstruction error given by individual base learners $\Delta^R_I(x, \mathbf{y})$.*

In addition, the gap can be estimated as

$$\Delta_I^R(x,\mathbf{y}) - \Delta_{MAM}^R(x,\mathbf{y}) = \boldsymbol{Var}(\sum_{j\in V}\Psi_j(x,y_j)) \geq 0.$$

The variance can be further expanded as

$$\boldsymbol{Var}(\sum_{j\in V}\Psi_j(x,y_j)) = \underbrace{\sum_{j\in V}\boldsymbol{Var}(\Psi_j(x,y_j))}_{diversity} + \underbrace{\sum_{\substack{p,q\in V,\\ p\neq q}}\boldsymbol{Cov}(\Psi_p(x,y_p),\Psi_q(x,y_q))}_{coherence}.$$

Proof By expanding and simplifying the squares we get

$$\begin{aligned}
\Delta_I^R(x,\mathbf{y}) - \Delta_{\mathrm{MAM}}^R(x,\mathbf{y}) &= \frac{1}{T}\sum_t \left(\psi^*(x,\mathbf{y}) - \psi^{(t)}(x,\mathbf{y})\right)^2 - \left(\psi^*(x,\mathbf{y}) - \psi^{\mathrm{MAM}}(x,\mathbf{y})\right)^2 \\
&= \frac{1}{T}\sum_t \left(\sum_{j\in V}\psi_j^*(x,y_j) - \sum_{j\in V}\psi_j^{(t)}(x,y_j)\right)^2 - \left(\sum_{j\in V}\psi_j^*(x,y_j) - \sum_{j\in V}\frac{1}{T}\sum_t\psi_j^{(t)}(x,y_j)\right)^2 \\
&= \frac{1}{T}\sum_t \left(\sum_{j\in V}\psi_j^{(t)}(x,y_j)\right)^2 - \left(\frac{1}{T}\sum_t\sum_{j\in V}\psi_j^{(t)}(x,y_j)\right)^2 \\
&= \mathbf{Var}(\sum_{j\in V}\Psi_j(x,y_j)) \\
&\geq 0.
\end{aligned}$$

The expression of variance can be further expanded as

$$\begin{aligned}
\mathbf{Var}(\sum_{j\in V}\Psi_j(x,y_j)) &= \sum_{p,q\in V}\mathbf{Cov}(\Psi_p(x,y_p),\Psi_q(x,y_q)) \\
&= \sum_{j\in V}\mathbf{Var}(\Psi_j(x,y_j)) + \sum_{\substack{p,q\in V,\\ p\neq q}}\mathbf{Cov}(\Psi_p(x,y_p),\Psi_q(x,y_q)).
\end{aligned}$$

■

The Theorem 1 states that the reconstruction error from MAM ensemble is guaranteed to be less than or equal to the average reconstruction error from the individuals. In particular, the improvement can be further addressed by two terms, namely *diversity* and *coherence*. The classifier diversity measures the variance of predictions from base learners independently on each single labels. It has been previously studied in single-label classifier ensemble context by Krogh and Vedelsby (1995). The diversity term prefers the variability of individuals that learn from different perspectives. It is a well known factor to improve the ensemble performance. The coherence term, that is specific to the multilabel classifiers, indicates that the more the microlabel predictions vary together, the greater advantage multilabel ensemble gets over the base learners. This supports our intuitive understanding that microlabel correlations are keys to successful multilabel learning.

Table 1: Statistics of multilabel datasets used in our experiments. For *NCI60* and *Fingerprint* dataset where there is no explicit feature representation, the rows of kernel matrix is assumed as feature vector.

Dataset	Statistics				
	Instances	**Labels**	**Features**	**Cardinality**	**Density**
Emotions	593	6	72	1.87	0.31
Yeast	2417	14	103	4.24	0.30
Scene	2407	6	294	1.07	0.18
Enron	1702	53	1001	3.36	0.06
Cal500	502	174	68	26.04	0.15
Fingerprint	490	286	490	49.10	0.17
NCI60	4547	60	4547	11.05	0.18
Medical	978	45	1449	1.14	0.03
Circle10	1000	10	3	8.54	0.85
CIrcle50	1000	50	3	35.63	0.71

4. Experiments

4.1. Datasets

We experiment on a collection of ten multilabel datasets from different domains, including chemical, biological, and text classification problems. The *NCI60* dataset contains 4547 drug candidates with their cancer inhibition potentials in 60 cell line targets. The *Fingerprint* dataset links 490 molecular mass spectra together to 286 molecular substructures used as prediction targets. Four text classification datasets[1] are also used in our experiment. In addition, two artificial *Circle* dataset are generated according to (Bian et al., 2012) with different amount of labels. An overview of the datasets is shown in Table 1, where *cardinality* is the average number of positive microlabels in the examples, defined as

$$cardinality = \frac{1}{m}\sum_{i=1}^{m}|\{j|\mathbf{y}_{ij} = 1\}|,$$

and *density* is the average number of labels of examples divided by the size of label space as

$$density = cardinality/k.$$

We calculate linear kernel on datasets where examples are described by feature vectors. For text classification datasets, we first compute TF-IDF weighted features. For *Fingerprint* datasets we compute quadratic kernel over the 'bag' of mass/charge peak intensities in the MS/MS spectra. On this dataset, as feature vectors for non-kernelized methods the rows of the training kernel matrix are used, due to the intractability of using the explicit features.

1. Available at http://mulan.sourceforge.net/datasets.html

4.2. Compared Classification Methods

For comparison, we choose the following established classification methods form different perspectives towards multilabel classification, accounting for single-label and multilabel, as well as ensemble and standalone methods:

- Support Vector Machine (SVM) is used as the single-label non-ensemble baseline classification model. In practice, we train a collection of SVMs, one for each microlabel.
- Bagging (Breiman, 1996) is used as the benchmark single-label ensemble method. In practice, we randomly select 40% of the data as input to SVM to get a weak hypothesis, and repeat the process until we collect an ensemble of 60 weak hypotheses.
- MMCRF (Rousu et al., 2007) is used both as a standalone multilabel classifier and the base classifier in the ensembles. Individual MMCRF models are trained with random tree as output graph structures.
- Multi-task feature learning (MTL), proposed in (Argyriou et al., 2007), is used as another multilabel benchmark.
- AdaBoostMH is a multilabel variant of AdaBoost developed in (Schapire and Singer, 2000). In our study, we use real-valued decision tree with at most 100 leaves as base learner of AdaBoostMH. We successively generate an ensemble of 100 weak hypothesises.

4.3. Obtaining Random Output Graphs

Output graphs for the graph labeling classifiers are generated by first drawing a random $k \times k$ matrix with non-negative edge weights and then extracting a maximum weight spanning tree out of the matrix. The spanning tree connects all targets so that the complex microlabel dependencies can be learned. Also, the tree structure facilitates efficient inference.

4.4. Parameter Selection and Evaluation Measures

We first sample 10% data uniform at random from each experimental dataset for parameter selection. Both SVM and MMCRF base models have margin softness parameter C, which potentially need to be tuned. We tested parameter C from a set $\{0.01, 0.1, 0.5, 1, 5, 10\}$ based on tuning data for both SVM and base learner MMCRF, then keep the best ones for the following validation step. We also perform extensive selection on γ parameters in MTL model in the same range as margin softness parameters.

Because most of the multilabel datasets are highly biased with regards to multilabel density, we use the following *stratified* 5-*fold cross validation* scheme in the experiments reported, such that we group examples in equivalent classes based on the number of positive labels they have. Each equivalent class is then randomly split into five local folds, after that the local folds are merged to create five global folds. The proposed procedure ensures that also the smaller classes have representations in all folds.

To quantitatively evaluate the performance of different classifiers, we adopt several performance measures. We report *multilabel accuracy* which counts the proportion of multilabel

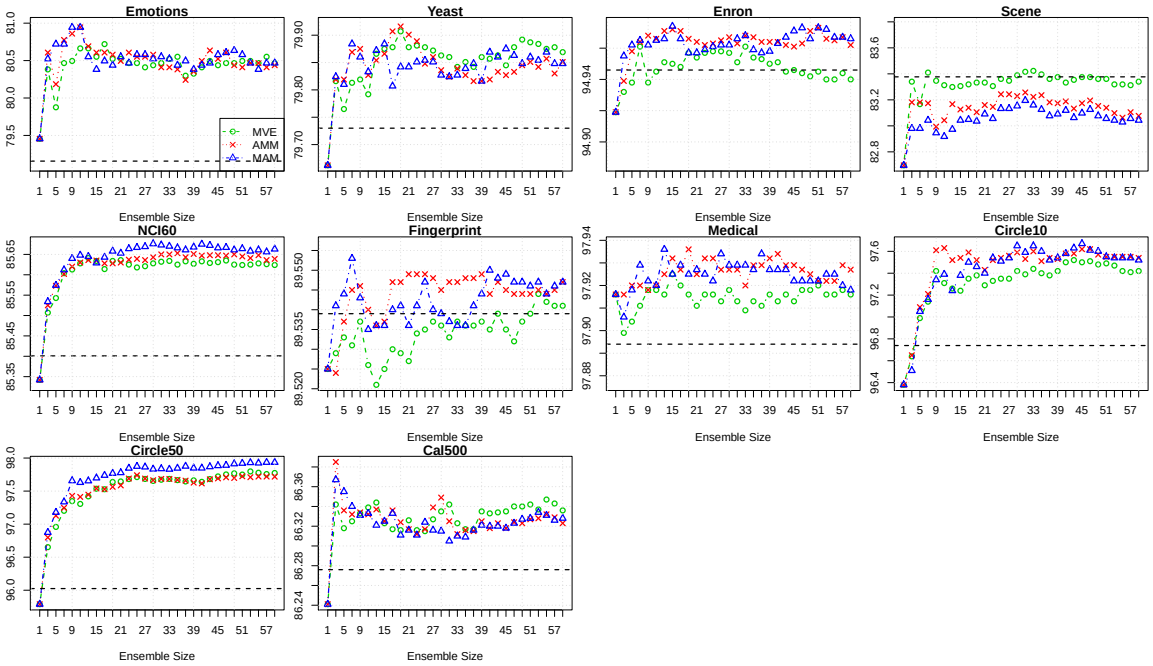

Figure 1: Ensemble learning curve (microlabel accuracy) plotted as the size of ensemble. Average performance of base learner with random tree as output graph structure is denoted as horizontal dash line.

predictions that have all of the microlabels being correct, *microlabel accuracy* as the proportion of microlabel being correct, and *microlabel* F_1 *score* that is the harmonic mean of microlabel precision and recall $F_1 = 2 \cdot \frac{Pre \times Rec}{Pre + Rce}$.

4.5. Comparison of Different Ensemble Approaches

Figure 1 depicts the ensemble learning curves in varying datasets with respect to microlabel accuracy. In general, there is a clear trend of improving microlabel accuracy for random tree based ensemble approaches as more individual base models are combined. We observe the similar trends in multilabel accuracy and microlabel F_1 space (see supplementary material for plots). We also notice that most of the learning curves converge even with a small ensemble size.

All three proposed ensemble learners (MVE, AMM, MAM) outperform their base learner MMCRF (horizontal dash lines) with significant margins in almost all datasets, the *Scene* being the only exception. AMM and MAM outperform MVE in all datasets except for *Scene* and *Cal500*. Furthermore, MAM approach surpasses AMM in nine out of ten datasets. Consequently, we choose MAM for the further studies described in the following section.

Table 2: Prediction performance of each algorithm in terms of microlabel accuracy, multilabel accuracy, and microlabel F_1 score ('−' denotes no positive predictions). @*Top2* counts how many times the algorithm achieves at least the second best.

Dataset	**Microlabel Accuracy**					
	Svm	Bagging	AdaBoost	Mtl	Mmcrf	Mam
Emotions	77.3±1.9	74.1±1.8	76.8±1.6	*79.8±1.8*	79.2±0.9	**80.5±1.4**
Yeast	**80.0±0.6**	78.4±0.7	74.8±0.3	79.3±0.2	79.7±0.3	*79.9±0.4*
Scene	**90.2±0.3**	87.8±0.8	84.3±0.4	*88.4±0.6*	83.4±0.2	83.0±0.2
Enron	93.6±0.2	93.7±0.1	86.2±0.2	93.5±0.1	*94.9±0.1*	**95.0±0.2**
Cal500	**86.3±0.3**	86.0±0.2	74.9±0.4	86.2±0.2	**86.3±0.2**	**86.3±0.3**
Fingerprint	**89.7±0.2**	85.0±0.7	84.1±0.5	82.7±0.3	*89.5±0.3*	*89.5±0.8*
NCI60	84.7±0.7	79.5±0.8	79.3±1.0	84.0±1.1	*85.4±0.9*	**85.7±0.7**
Medical	97.4±0.1	97.4±0.1	91.4±0.3	97.4±0.1	**97.9±0.1**	**97.9±0.1**
Circle10	94.8±0.9	92.9±0.9	**98.0±0.4**	93.7±1.4	96.7±0.7	*97.5±0.3*
Circle50	94.1±0.3	91.7±0.3	*96.6±0.2*	93.8±0.7	96.0±0.1	**97.9±0.2**
@*Top*2	4	0	2	2	*5*	**9**
Dataset	**Multilabel Accuracy**					
	Svm	Bagging	AdaBoost	Mtl	Mmcrf	Mam
Emotions	21.2±3.4	20.9±2.6	23.8±2.3	25.5±3.5	*26.5±3.1*	**30.4±4.2**
Yeast	*14.0±1.8*	13.1±1.2	7.5±1.3	11.3±2.8	13.8±1.5	**14.0±0.6**
Scene	**52.8±1.0**	*46.5±2.5*	34.7±1.8	44.8±3.0	12.6±0.7	5.4±0.5
Enron	0.4±0.1	0.1±0.2	0.0±0.0	0.4±0.3	*11.7±1.2*	**12.1±1.0**
Cal500	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
Fingerprint	**1.0±1.0**	0.0±0.0	0.0±0.0	0.0±0.0	*0.4±0.9*	*0.4±0.5*
NCI60	*43.1±1.3*	21.1±1.3	2.5±0.6	**47.0±1.4**	36.9±0.8	40.0±1.0
Medical	8.2±2.3	8.2±1.6	5.1±1.0	8.2±1.2	*35.9±2.1*	**36.9±4.6**
Circle10	69.1±4.0	64.8±3.2	**86.0±2.0**	66.8±3.4	75.2±5.6	*82.3±2.2*
Circle50	29.7±2.5	21.7±2.6	28.9±3.6	27.7±3.4	*30.8±1.9*	**53.8±2.2**
@*Top*2	5	2	2	2	*6*	**8**
Dataset	**Microlabel F_1 Score**					
	Svm	Bagging	AdaBoost	Mtl	Mmcrf	Mam
Emotions	57.1±4.4	61.5±3.1	*66.2±2.9*	64.6±3.0	64.6±1.2	**66.3±2.3**
Yeast	62.6±1.2	**65.5±1.3**	*63.5±0.6*	60.2±0.5	62.4±0.7	62.4±0.6
Scene	*68.3±0.9*	**69.9±1.9**	64.8±0.8	61.5±2.4	23.7±1.2	11.6±0.9
Enron	29.4±1.0	38.8±1.5	42.3±1.1	-	**53.8±1.3**	*53.7±0.7*
Cal500	31.4±0.8	*40.1±0.3*	**44.3±0.5**	28.6±0.6	32.7±0.9	32.3±0.9
Fingerprint	**66.3±0.8**	64.4±1.9	62.8±1.6	0.4±0.4	*65.0±1.4*	*65.0±2.1*
NCI60	45.9±1.9	**53.9±1.3**	32.9±2.0	32.9±0.9	46.7±2.8	*47.1±2.9*
Medical	-	-	33.7±1.1	-	*49.5±3.5*	**50.3±3.5**
Circle10	97.0±0.5	96.0±0.5	**98.8±0.2**	96.4±0.9	98.1±0.4	*98.6±0.2*
Circle50	96.0±0.3	94.5±0.2	*97.6±0.1*	95.7±0.5	97.2±0.1	**98.6±0.1**
@*Top*2	2	4	*5*	0	3	**7**

4.6. Multilabel Prediction Performance

We examine whether our proposed ensemble model (MAM) can boost the prediction performance in multilabel classification problems. Therefore, we compare our model with other advanced methods including both single-label and multilabel classifiers, both standalone and ensemble frameworks. Table 2 shows the performance of difference methods in terms of microlabel accuracy, multilabel accuracy and microlabel F_1 score, where the best performance in each dataset is emphasised in **boldface** and the second best is in *italics*. We also count how many times each algorithm achieves at least the second best performance. The total count is shown as *'@Top2'*.

We observe from Table 2 that MAM outperforms both standalone and ensemble competitors in all three measurements. In particular, it is ranked nine times as top 2 methods in microlabel accuracy, eight times in multilabel accuracy, and seven times in microlabel F_1 score. The only datasets where MAM is consistently outside the top 2 is the *Scene* dataset. The dataset is practically a single-label multiclass dataset, with very few examples with more than one positive microlabel. The graph-based approaches MMCRF and MAM do not seem to be able to cope with the extreme label sparsity. However, on this dataset the single target classifiers SVM and Bagging outperform all compared multilabel classifiers.

In these experiments, MMCRF also performs robustly, being in top 2 on half of the datasets with respect to microlabel and multilabel accuracy, however, quite consistently trailing to MAM, often with a noticeable margin.

We also notice that the standalone single target classifier SVM is competitive against most multilabel methods, placing in top 2 more often than Bagging, AdaBoost and MTL with respect to microlabel and microlabel accuracy.

Overall, the results indicate that ensemble by MAM is a robust and competitive alternatives for multilabel classification.

5. Conclusions

In this paper we have put forward new methods for multilabel classification, relying on ensemble learning on random output graphs. In our experiments, models thus created have favourable predictive performances on a heterogeneous collection of multilabel datasets, compared to several established methods. The theoretical analysis of the MAM ensemble highlights the covariance of the compatibility scores between the inputs and microlabels learned by the base learners as the quantity explaining the advantage of the ensemble prediction over the base learners. Our results indicate that structured output prediction methods can be successfully applied to problems where no prior known output structure exists, and thus widen the applicability of the structured output prediction.

We leave it as an open problem to analyze the generalization error of this type of classifiers. We also plan to link diversity term to model performance through empirical evaluations.

Acknowledgments

The work was financially supported by Helsinki Doctoral Programme in Computer Science (Hecse), Academy of Finland grant 118653 (ALGODAN), IST Programme of the European

Community under the PASCAL2 Network of Excellence, ICT-2007-216886. This publication only reflects the authors' views.

References

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In *Advances in Neural Information Processing Systems 19*. MIT Press, 2007.

Wei Bian, Bo Xie, and Dacheng Tao. Corrlog: Correlated logistic models for joint prediction of multiple labels. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics (AISTATS-12)*, volume 22, pages 109–117, 2012.

Leo Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

Gavin Brown and Ludmila I Kuncheva. good and bad diversity in majority vote ensembles. In *Multiple Classifier Systems*, pages 124–133. Springer, 2010.

A. Esuli, T. Fagni, and F. Sebastiani. Boosting multi-label hierarchical text categorization. *Information Retrieval*, 11(4):287–313, 2008.

Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation, and active learning. In *Advances in Neural Information Processing Systems*, pages 231–238. MIT Press, 1995.

J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor. Kernel-Based Learning of Hierarchical Multilabel Classification Models. *The Journal of Machine Learning Research*, 7: 1601–1626, 2006.

J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor. Efficient algorithms for max-margin structured classification. *Predicting Structured Data*, pages 105–129, 2007.

Robert E. Schapire and Yoram Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135 – 168, 2000.

H. Su and J. Rousu. Multi-task drug bioactivity classification with graph labeling ensembles. *Pattern Recognition in Bioinformatics*, pages 157–167, 2011.

B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *Neural Information Processing Systems*, 2003.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *ICML'04*, pages 823–830, 2004.

M.J. Wainwright, T.S. Jaakkola, and A.S. Willsky. MAP estimation via agreement on trees: message-passing and linear programming. *IEEE Transactions on Information Theory*, 51 (11):3697–3717, 2005.

R. Yan, J. Tesic, and J.R. Smith. Model-shared subspace boosting for multi-label classification. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 834–843. ACM, 2007.