Joint Mode Estimation in Multi-Label Classification by Chaining

Krzysztof Dembczyński[1,3], Willem Waegeman[2], and
Eyke Hüllermeier[1]

[1] Department of Mathematics and Computer Science, Marburg University,
Hans-Meerwein-Str., 35039 Marburg, Germany
{dembczynski, eyke}@informatik.uni-marburg.de
[2] Department of Mathematical Modelling, Statistics and Bioinformatics, Ghent University, Coupure links 653, B-9000 Ghent, willem.waegeman@ugent.be
[3] Institute of Computing Science, Poznań University of Technology, Piotrowo 2,
60-965 Poznań, Poland

Abstract. Many recently proposed algorithms in multi-label classification are believed to outperform their baseline competitors by exploiting structure and dependencies in the label space. However, most of these algorithms are presented in a purely application-driven manner, despite being intuitively appealing and showing strong performance in empirical studies. In this article we study one of these methods in detail, namely classifier chains, thereby helping to gain a better understanding of this approach. As a main result, we clarify that the original chaining method intends to predict the joint mode of the conditional distribution of label vectors in an approximate manner. Since exact inference is known to be intractable in general, this is of course a reasonable strategy. However, as a result of a theoretical regret analysis, we conclude that the existing greedy algorithm can perform quite poorly in terms of subset 0/1 loss. Therefore, we present an enhanced inference procedure for which the worst-case regret can be upper-bounded far more tightly. Finally, we discuss connections with related frameworks, such as conditional random fields and structured support vector machines, and we present experimental results confirming the validity of our theoretical findings.

1 Introduction

Multi-label classification (MLC) is a relatively new but rapidly expanding subfield of machine learning, which differs from conventional binary classification insofar as multiple binary labels have to be predicted simultaneously. This transition from predicting a single label to predicting multiple labels raises a number of computational and statistical challenges, such as the need for modeling statistical dependencies between labels and optimizing a wide range of loss functions in a potentially high-dimensional label space. However, it often happens that precise knowledge of structure and dependencies in the label space cannot be provided in MLC problems, in contrast to many other applications of structured output prediction, so MLC algorithms should detect this information automatically.

Most of the multi-label classification (MLC) methods proposed in recent years intend to exploit, in one way or the other, dependencies between the class labels. Comparing to simple binary relevance (BR) learning as a baseline, any gain in performance is normally explained by the fact that BR is ignoring such dependencies. Without questioning the correctness of such studies, one has to admit that a blanket explanation of that kind is hiding many subtle details, and indeed, the underlying mechanisms and true reasons for the improvements reported in experimental studies are rarely laid bare.

For example, the recently introduced classifier chains (CC) [1] has not been thoroughly analyzed in a theoretical way, despite being intuitively appealing and showing strong performance in empirical studies. In this paper, we will analyze this chaining method and its probabilistic variant more deeply, particularly in terms of loss minimization and the related issue of modelling label dependencies. From a probabilistic perspective, it is clear that different properties of the joint conditional distribution over labels are needed for optimizing the different loss functions that are currently used in MLC, so one cannot expect that a single multi-label classifier outperforms competing algorithms simultaneously for all possible loss functions [2]. From this viewpoint, we will show that the original CC algorithm is likely to outperform BR in terms of subset 0/1 loss, while BR is likely to outperform CC in terms of Hamming loss.

Conversely, different loss functions can be optimized with probabilistic classifier chains (PCC), the probabilistic variant of chaining introduced in [3]. The output of the classifier chain corresponds in the PCC method to an estimate of the joint probability distribution, for which an inference procedure is needed in order to obtain the right prediction for a given loss. The PCC method, however, has been only analyzed with an exhaustive inference algorithm that is intractable for problems with more than 12-15 labels. Therefore, an enhanced approximate inference algorithm is introduced, for which substantially tighter worst-case regret bounds are derived as a function of the running time of the algorithm (assuming that conditional probabilities can be estimated perfectly). In the experiments, we show that the exploitation of label dependencies by joint mode estimation leads to a clear improvement over BR in terms of subset 0/1 loss. For the Hamming loss, we show that it suffices to estimate the marginal modes.

2 Joint Mode versus Marginal Mode Prediction

Let $\mathcal{X}$ denote an instance space, and let $\mathcal{L} = \{\lambda_1, \lambda_2, \ldots, \lambda_m\}$ be a finite set of class labels. We assume that an instance $\boldsymbol{x} \in \mathcal{X}$ is (non-deterministically) associated with a subset of labels $L \in 2^{\mathcal{L}}$; this subset is often called the set of relevant labels, while the complement $\mathcal{L} \setminus L$ is considered as irrelevant for $\boldsymbol{x}$. We identify a set L of relevant labels with a binary vector $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$, in which $y_i = 1 \Leftrightarrow \lambda_i \in L$. By $\mathcal{Y} = \{0, 1\}^m$ we denote the set of possible labelings.

We assume observations to be generated independently and identically according to a probability distribution $\mathbf{P}(\mathbf{X},\mathbf{Y})$ on $\mathcal{X} \times \mathcal{Y}$, i.e., an observation $\boldsymbol{y} = (y_1, \ldots, y_m)$ is the realization of a corresponding random vector $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_m)$. We denote by $\mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x})$ the conditional distribution of $\mathbf{Y} = \boldsymbol{y}$ given $\mathbf{X} = \boldsymbol{x}$, and by $\mathbf{P}(y_i = b|\boldsymbol{x})$ the corresponding marginal distribution of Y_i:

$$\mathbf{P}(y_i = b|\boldsymbol{x}) = \sum_{\boldsymbol{y} \in \mathcal{Y}: y_i = b} \mathbf{P}(\boldsymbol{y}|\boldsymbol{x})$$

Let us denote a multi-label classifier $\mathbf{h}$ as an $\mathcal{X} \rightarrow \mathcal{Y}$ mapping that returns a vector $\mathbf{h}(\boldsymbol{x}) = (h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots, h_m(\boldsymbol{x}))$ for a given instance $\boldsymbol{x} \in \mathcal{X}$. Given training data in the form of a finite set of observations $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{X} \times \mathcal{Y}$, drawn independently from $\mathbf{P}(\mathbf{X},\mathbf{Y})$, the goal in multi-label classification is to learn a classifier $\mathbf{h} : \mathcal{X} \rightarrow \mathcal{Y}$ that generalizes well beyond these observations in the sense of minimizing the risk with respect to a specific loss function. The risk of a classifier $\mathbf{h}$ is defined as the expected loss over the joint distribution $\mathbf{P}(\mathbf{X},\mathbf{Y})$:

$$R_L(\mathbf{h}) = \mathbb{E}_{\mathbf{XY}} L(\mathbf{Y}, \mathbf{h}(\mathbf{X})), \tag{1}$$

where $L(\cdot)$ is a loss function on multi-label predictions. The so-called risk-minimizing model $\mathbf{h}^*$ is determined in a pointwise way by the *risk minimizer*

$$\mathbf{h}^*(\boldsymbol{x}) = \arg\min_{\boldsymbol{h}} \mathbb{E}_{\mathbf{Y}|\mathbf{X}} L(\mathbf{Y}, \boldsymbol{h}(\boldsymbol{x})) = \arg\min_{\boldsymbol{h}} \sum_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) L(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})). \tag{2}$$

As we are dealing with a multivariate conditional probability distribution over the labels, two of its properties are always of interest: the joint and the marginal mode.

Proposition 1. *[3] Predicting the joint mode leads to a model of the following form:*

$$\mathbf{h}^*(\boldsymbol{x}) = \arg\max_{\boldsymbol{y} \in \mathcal{Y}} \mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) \ , \tag{3}$$

corresponding to the risk minimizer (2) of the so-called subset 0/1 loss*, which is formally defined as follows:*[4]

$$L_s(\boldsymbol{y}, \mathbf{h}(\boldsymbol{x})) = [\![\boldsymbol{y} \neq \mathbf{h}(\boldsymbol{x})]\!] \ . \tag{4}$$

Predicting the marginal (conditional) modes, in turn, leads to the model

$$h_i^*(\boldsymbol{x}) = \arg\max_{b \in \{0,1\}} \mathbf{P}(y_i = b \,|\, \boldsymbol{x}) \tag{5}$$

corresponding to the risk minimizer (2) for the Hamming loss, defined as the fraction of labels whose relevance is incorrectly predicted:

$$L_H(\boldsymbol{y}, \mathbf{h}(\boldsymbol{x})) = \frac{1}{m} \sum_{i=1}^{m} [\![y_i \neq h_i(\boldsymbol{x})]\!] \, . \tag{6}$$

[4] For a predicate P, the expression $[\![P]\!]$ evaluates to 1 if P is true and to 0 if P is false.

Predicting the joint (conditional) mode can be considered as a core operation in many structured output prediction methods such as conditional random fields [4]. Modeling the joint conditional distribution and its joint mode involves exploiting conditional dependence between labels, unlike modeling the marginal modes, where the gain by exploiting the conditional dependence, if any, is rather small. In order to improve the performance in estimating the marginal distributions, the methods should rather exploit the marginal dependence, as explained for example in [5].

3 Probabilistic Classifier Chains

The Probabilistic Classifier Chains (PCC) method has been introduced in [3] in an attempt to provide a probabilistic interpretation for the previously published Classifier Chains (CC) method [1]. The idea underlying PCC is to repeatedly apply the product rule of probability to the joint distribution of the labels $\boldsymbol{Y} = (Y_1, \ldots, Y_m)$:

$$\mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x}) = \prod_{k=1}^{m} \mathbf{P}(y_k | \boldsymbol{x}, y_1, \ldots, y_{k-1}) \tag{7}$$

In other words, PCC represents conditional label dependencies as a fully connected graph. From a theoretical point of view, the order of labels does not play any role, and (7) holds for any permutation of $\boldsymbol{Y} = (Y_1, \ldots, Y_m)$.

Learning a classifier chain can be considered as a very simple procedure. According to (7), we decompose the joint distribution to a sequence of marginal distributions that depend on a subset of the labels. These marginal distributions can be learned by m functions $f_i(\cdot)$ on an augmented input space $\mathcal{X} \times \{0,1\}^{i-1}$, taking $y_1, \ldots, y_{k-1}$ as additional input attributes:

$$\begin{aligned} f_k :\, & \mathcal{X} \times \{0,1\}^{k-1} \rightarrow [0,1] \\ & (\boldsymbol{x}, y_1, \ldots, y_{k-1}) \mapsto \mathbf{P}(y_k = 1 \,|\, \boldsymbol{x}, y_1, \ldots, y_{k-1}) \end{aligned}$$

We assume that the function $f_k(\cdot)$ can be interpreted as a *probabilistic* classifier whose prediction is the probability that $y_i = 1$, or at least a reasonable approximation thereof. Thus, for any $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$, its probability can be estimated by

$$\hat{\mathbf{P}}(\boldsymbol{y} | \boldsymbol{x}) = \prod_{k=1}^{m} f_k(\boldsymbol{x}, y_1, \ldots, y_{k-1})\,. \tag{8}$$

The problem is then to find the risk minimizer for a given loss function over the estimated joint conditional distribution. This process is often referred to as *inference*, and it will be thoroughly analyzed in the next section. To this end, it is convenient to represent the estimated joint conditional distribution as a probability tree. We define the probability tree as a structure $(V, E, \mathit{\Pi})$ with V the set of nodes, E the set of edges and $\mathit{\Pi} : E \rightarrow [0,1]$ a function that assigns positive weights to edges. Moreover, let us denote a node at depth k as

$v_{\boldsymbol{a}} = (a_1, ..., a_k) \in \{0, 1\}^k$, then the weight of the edge between such a node and its ancestor $\boldsymbol{pa}(v) = (a_1, ..., a_{k-1})$ at depth $k-1$ is given by

$$\Pi(v_{\boldsymbol{a}}) = \mathbf{P}(Y_k = a_k \mid \boldsymbol{x}, y_1 = a_1, ..., y_{k-1} = a_{k-1}) \, .$$

As such, depth k of the probability tree represents the decision that is taken in the k-th classifier of the chain. The root of the tree $v_R = \emptyset$ corresponds to depth $k = 0$ with $\Pi(v_R) = 1$.

4 Inference in Probabilistic Classifiers Chains

Originally, two approaches have been proposed for inferring a prediction from an estimated chain: an approach based on greedy search, being the integral part of the original CC method [1], and an approach based on exhaustive search, as considered in the PCC method [3].

4.1 Inference by Exhaustive Search

In inference by *exhaustive search* one assumes that an optimal prediction can be computed explicitly via (2), given an estimate of $\mathbf{P}(\boldsymbol{y} \,|\, \boldsymbol{x})$ for all $\boldsymbol{y}$ and a loss function $L(\cdot)$. Nonetheless, such an approach is extremely costly, as it results in taking the sum over an exponential (2^m) number of label combinations. Moreover, the brute-force search for the optimal solution would also require to check all possible combinations of labels. For some loss functions, like subset 0/1 and Hamming loss, one iteration through the label combinations suffices to compute the optimal solutions, however, this still limits the applicability of the method to datasets with a small to moderate number of labels.

From this point of view, PCC can be treated as a general method for multi-label classification. However, approximate algorithms might be needed for loss functions for which exact inference becomes intractable. Due to lack of space, we will not offer such a discussion in this paper. In contrast, we will focuss on the subset 0/1 loss, for which an enhanced approximate algorithm is developed.

4.2 Inference by Greedy Search

Inference by *greedy search*, for which the pseudo code is given in Algorithm 1, has been introduced as an integral part of the CC method. Briefly summarized, this inference algorithm just follows a single path from the root to one specific leaf. For a new instance $\boldsymbol{x}$ to be classified, the model f_1 predicts $\hat{y}_1$, i.e., the relevance of λ_1 for $\boldsymbol{x}$, as usual. Then, f_2 predicts the relevance of λ_2, taking $\boldsymbol{x}$ *plus the predicted value* $\hat{y}_1 \in \{0, 1\}$ as an input. Proceeding in this way, f_i predicts $\hat{y}_i$ using $\hat{y}_1, \ldots, \hat{y}_{i-1}$ as additional input information. The main advantages of this approach are (a) its low cost and (b) the possibility to use non-probabilistic classifiers, as one only needs to know whether a given label is relevant or not to take a greedy decision in following a path from the root to a leaf. However, we will show for two loss functions that the regret of such an approach can be large.

Algorithm 1 Inference by Greedy Search

```
$v \leftarrow$ the root of the probability tree
while $v$ is not a leaf do
    $\boldsymbol{lc}(v), \boldsymbol{rc}(v) \leftarrow$ left and right child of $v$
    if $\Pi(\boldsymbol{lc}(v) \geq \Pi(\boldsymbol{rc}(v))$ then
        $v \leftarrow \boldsymbol{lc}(v)$
    else
        $v \leftarrow \boldsymbol{rc}(v)$
    end if
end while
return $v = (a_1, ..., a_m)$ as the mode
```

The regret of a classifier $\boldsymbol{h}$ with respect to a loss function L_z is defined as follows:

$$r_{L_z}(\boldsymbol{h}) = R_{L_z}(\boldsymbol{h}) - R_{L_z}(\boldsymbol{h}^*_z), \tag{9}$$

where R is the risk given by (1), and $\boldsymbol{h}^*_z$ is the Bayes-optimal classifier with respect to the loss function L_z. In the following, we consider the regret with respect to the Hamming loss, given by

$$r_H(\boldsymbol{h}) = \mathbb{E}_{\mathbf{XY}} L_H(\mathbf{Y}, \boldsymbol{h}(\mathbf{X})) - \mathbb{E}_{\mathbf{XY}} L_H(\mathbf{Y}, \boldsymbol{h}^*_H(\mathbf{X})),$$

and the subset 0/1 loss, given by

$$r_s(\boldsymbol{h}) = \mathbb{E}_{\mathbf{XY}} L_s(\mathbf{Y}, \boldsymbol{h}(\mathbf{X})) - \mathbb{E}_{\mathbf{XY}} L_s(\mathbf{Y}, \boldsymbol{h}^*_s(\mathbf{X})).$$

Since both loss functions are decomposable with respect to individual instances, we analyze the expectation over $\mathbf{Y}$ for a given $\boldsymbol{x}$. The following proposition identifies the highest value of the regret for the greedy approach in terms of the subset 0/1 loss and the Hamming loss (proof omitted due to lack of space).

Theorem 1. *Under the assumption that a probabilistic classifier chain obtains a perfect estimate of the conditional probability $\mathbf{P}(\boldsymbol{y}|\boldsymbol{x})$, the following tight upper bounds hold for the regret of the prediction $\boldsymbol{h}_G(\boldsymbol{x})$ of the greedy approach:*

$$\sup_{\mathbf{P}} \left(\mathbb{E}_{\mathbf{Y}} L_s(\mathbf{Y}, \boldsymbol{h}_G(\boldsymbol{x})) - \mathbb{E}_{\mathbf{Y}} L_s(\mathbf{Y}, \boldsymbol{h}^*_s(\boldsymbol{x}))\right) = 2^{-1} - 2^{-m},$$

$$\sup_{\mathbf{P}} \left(\mathbb{E}_{\mathbf{Y}} L_H(\mathbf{Y}, \boldsymbol{h}_G(\boldsymbol{x})) - \mathbb{E}_{\mathbf{Y}} L_H(\mathbf{Y}, \boldsymbol{h}^*_H(\boldsymbol{x}))\right) = 1 - \frac{2}{m}\sum_{i=1}^{m} 2^{-i},$$

where the supremum is taken over all probability distributions on $\mathcal{Y}$.

As we can see, the regret is quite high in both cases, suggesting that inference by greedy search can yield a poor performance for both loss functions. Nevertheless, we argue that this approach is still more appropriate for the subset 0/1 loss. When the number of labels increases, the regret converges to 0.5 for the subset 0/1 loss, while it even converges to the maximum possible value of 1 for the Hamming loss. Hence, it is tempting to conclude that the greedy search procedure is indeed more suitable for estimating the joint than the marginal mode, all the more since the subset 0/1 loss, in terms of its absolute value, is even higher than the Hamming loss (which, for example, can already be reduced to 1/2 by random guessing). Furthermore, one may wonder whether one can find an optimal order of labels in the chain, for which the regret would decrease to zero. Unfortunately, this is provably impossible (proof omitted due to space restrictions). An interesting issue being a subject of our future work is to check whether the maximal value of the regret becomes smaller if the order of the labels would be changed or even optimized.

Nevertheless, let us remark that the risk minimizers of the Hamming loss and the subset 0/1 loss coincide in many specific situations, like conditional independence of labels or if the probability of the joint mode is greater than or equal to 0.5 [2]. One can easily observe that the worst-case regret of the greedy search algorithm is zero for both losses in these two situations. At the same time, these facts may also explain why algorithms, despite not tailored for specific losses, have been reported to obtain good results in many empirical studies.

4.3 An ϵ-Approximate Algorithm

Since the regret of the greedy search procedure can be high, we propose in this section a specific algorithm for which a much smaller upper bound on the regret can be derived. From a graph-theoretic perspective, the algorithm computes the shortest path between the root of the probability tree and a fictitious dummy node that is connected to the leaves of the probability tree. Given the probability tree structure that was introduced in the previous section, let us define the path distance $\overline{\Pi}(v_{\boldsymbol{a}})$ between the root node $v_R = \emptyset$ and any node $v_{\boldsymbol{a}}$ recursively, as a product of edge weights:

$$\overline{\Pi}(v_{\boldsymbol{a}}) = \Pi(v_{\boldsymbol{a}}) \times \overline{\Pi}(\boldsymbol{pa}(v_{\boldsymbol{a}}))\,, \tag{10}$$

where $\boldsymbol{pa}(v)$ denotes the parent of a given node v.

Using this notation, the pseudo code of our algorithm is summarized in Algorithm 2. In a nutshell, the algorithm starts from the root of the probability tree, which is the single element of an ordered list Q. In every iteration, the top element of the list is popped and the children of the corresponding node are visited. The path distance $\overline{\Pi}(v)$ to the root can be recursively computed for these children, and they are added to the list if the path distance is bigger than the threshold $\epsilon = 2^{-k}$ with $1 \leq k \leq m$. Basically, they are inserted in the list at the appropriate position, so that the order imposed by $\overline{\Pi}(v)$ is respected.

The while loop of the algorithm stops in two situations: (1) when the element popped from the list Q corresponds to a leaf of the probability tree or (2) when the list Q is empty. The label combination corresponding to the leaf is then returned in the former case, while inference by greedy search, as described above, is applied to define a path from all non-survived nodes from the list K (i.e., nodes for which none of their children has been added to Q) to a leaf with corresponding

Algorithm 2 ϵ-Approximate Inference

```
ordered list $Q \leftarrow \{v_R\}$ (contains root node initially)
ordered list $K \leftarrow \{\}$ (non-survived parents)
define $\overline{\Pi}(v_R) = 1$
$\epsilon \leftarrow 2^{-k}$ with $k \leq m$
while $Q \neq \emptyset$ do
    $v \leftarrow$ pop first element in $Q$
    if $v$ is a leaf then
        delete all elements in $K$ and break the while loop
    end if
    $\boldsymbol{lc}(v), \boldsymbol{rc}(v) \leftarrow$ left and right child of $v$
    compute $\overline{\Pi}(\boldsymbol{lc}(v))$ and $\overline{\Pi}(\boldsymbol{rc}(v))$ recursively from $\overline{\Pi}(v)$ using Eq. (10)
    if $\overline{\Pi}(\boldsymbol{lc}(v)) \geq \epsilon$ then
        insert $\boldsymbol{lc}(v)$ in list $Q$ sorted according to $\overline{\Pi}(\boldsymbol{lc}(v))$
    end if
    if $\overline{\Pi}(\boldsymbol{rc}(v)) \geq \epsilon$ then
        insert $\boldsymbol{rc}(v)$ in list $Q$ sorted according to $\overline{\Pi}(\boldsymbol{rc}(v))$
    end if
    if $\boldsymbol{lc}(v)$ and $\boldsymbol{rc}(v)$ are not inserted to the list then
        insert $v$ in list $K$ sorted according to $\overline{\Pi}(v)$
    end if
end while
$\epsilon \leftarrow 0$
while $K \neq \emptyset$ do
    $v' \leftarrow$ pop first element in $K$ and apply Algorithm 1 downward on it
    if $\overline{\Pi}(v') \geq \epsilon$ then
        $v \leftarrow v'$ and $\epsilon \leftarrow \overline{\Pi}(v')$
    end if
end while
return $v = (a_1, ..., a_m)$ as the mode
```

prediction in the latter case. The following theorem states that in both cases the regret of the prediction can be bounded as a function of the number of iterations of the algorithm (proof omitted).

Theorem 2. *Let $k \leq m$. Under the assumption that a probabilistic classifier chain obtains a perfect estimate of the conditional probability $\mathbf{P}(\boldsymbol{y}|\boldsymbol{x})$, Algorithm 2 needs less than $\mathcal{O}(m2^k)$ iterations to find a prediction $h_\epsilon(\boldsymbol{x})$ with low worst-case regret for subset 0/1 loss, i.e.*

$$\sup_{\mathbf{P}} \left(\mathbb{E}_{\mathbf{Y}} L_s(\mathbf{Y}, \boldsymbol{h}_\epsilon(\boldsymbol{x})) - \mathbb{E}_{\mathbf{Y}} L_s(\mathbf{Y}, \boldsymbol{h}^*_s(\boldsymbol{x})) \right) \leq 2^{-k} - 2^{-m} .$$

Thus, the algorithm finds an upper bound on the regret as a function of the running time of the algorithm. Consequently, the algorithm will always find the mode of the distribution, if the probability mass of the mode is higher than the upper bound on the regret. This is summarized in the following corollary.

Corollary 1. *Let $k \leq m$ and let $\mathbf{P}$ be a probability distribution for which the joint mode has a probability mass bigger than 2^{-k}, then Algorithm 2 needs less than $m2^k$ iterations to find a prediction $h_\epsilon(\boldsymbol{x})$ that corresponds to this mode.*

5 Related Methods

If logistic regression models are used as base classifiers in the chain, a strong correspondence with conditional random fields can be established. This type of methods defines a probabilistic model for a set of output variables conditioned on a set of input variables in the following way:

$$\mathbf{P}_{\boldsymbol{w}}(\boldsymbol{y} \mid \boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x}, \boldsymbol{w})} e^{-\boldsymbol{w}^T \Psi(\boldsymbol{x}, \boldsymbol{y})} \,,$$

with $\boldsymbol{w}$ a vector of parameters, $\Psi(\boldsymbol{x}, \boldsymbol{y})$ a joint feature mapping on input and output variables, and $Z(\boldsymbol{x}, \boldsymbol{w}) = \sum_{\boldsymbol{y} \in \{0,1\}^m} e^{-\boldsymbol{w}^T \Psi(\boldsymbol{x}, \boldsymbol{y})}$ a normalization constant.

Probabilistic classifier chains with logistic regression models as base classifiers lead to the same representation with a specific choice for $\Psi(\boldsymbol{x}, \boldsymbol{y})$.[5] Let us denote by $y'_i = (2y_i - 1) \in \{-1, 1\}$. By applying the logistic model to every base classifier in the chain we obtain:

$$\begin{aligned}
\mathbf{P}_{\boldsymbol{w}}(\boldsymbol{y} \mid \boldsymbol{x}) &= \prod_{i=1}^{m} \frac{e^{-y'_i \boldsymbol{w}^T \phi(x, y'_1, \ldots, y'_{i-1})}}{e^{-y'_i \boldsymbol{w}^T \phi(x, y'_1, \ldots, y'_{i-1})} + e^{y'_i \boldsymbol{w}^T \phi(x, y'_1, \ldots, y'_{i-1})}} \\
&= \frac{1}{Z'(\boldsymbol{x}, \boldsymbol{w})} e^{-\sum_{i=1}^{m} y'_i \boldsymbol{w}^T \phi(x, y'_1, \ldots, y'_{i-1})}
\end{aligned}$$

with $\boldsymbol{w}_i$ the weight vector for the i-th classifier, ϕ a feature mapping and Z' different from Z. In the case of linear models, we end up with the following model:

$$\mathbf{P}_{\boldsymbol{w}}(\boldsymbol{y} \mid \boldsymbol{x}) = \frac{1}{Z'(\boldsymbol{x}, \boldsymbol{w})} e^{-\sum_{i=1}^{m} -y'_i \left(\sum_j w_j x_j + \sum_{j=1}^{i-1} w_j y'_j\right)}$$

So, the feature mapping $\Psi(\boldsymbol{x}, \boldsymbol{y}) = (y'_1 \boldsymbol{x}, \ldots, y'_m \boldsymbol{x}, y'_1 y'_2 \boldsymbol{x}, y'_1 y'_3 \boldsymbol{x}, \ldots, y'_{m-1} y'_m \boldsymbol{x})$ models all pairwise dependencies between labels. Hence, one may expect very similar results for the two approaches, but the fitted models will not necessarily be identical. As a main benefit, our approach allows to solve m learning problems independently during the training phase, without imposing any restrictions on modeling label dependencies. Let us also remark that we need complex inference algorithms for conditional random fields to solve MLC problems [4].

Similarly, a strong relationship might be claimed with structured support vector machines, which only differ from conditional random fields in the loss that is minimized, namely structured hinge loss instead of log-loss [6]. However, similar needs for approximate inference of general loss functions arise in such a

[5] We assume that the last element of $\boldsymbol{x}$ is always one (the bias term).

Table 1. Basic statistics for the datasets, including training and test set sizes, number of features and labels, and minimal, average, and maximal number of relevant labels.

DATA SET	# TRAIN INST.	# TEST INST.	# ATTR.	# LAB.	MIN	AVE.	MAX
SCENE	1211	1196	294	6	1	1.062	3
YEAST	1500	917	103	14	1	4.228	11
TMC2007-500	21519	7077	500	22	1	2.226	10
MEDICAL	333	645	1449	45	1	1.255	3
ENRON	1123	579	1001	53	1	3.386	11
REUTERS (SUBSET 1)	3000	3000	500	103	1	3.176	11
MEDIAMILL	30993	12914	120	101	0	4.363	18
EMOTIONS	391	202	72	6	1	1.813	3
SYNTH1	471	5045	6000	6	1	2.045	6
SYNTH2	1000	10000	40	10	1	1	1

context, as discussed thoroughly by [7] from a multi-label classification perspective. In general, efficient inference in the presence of many labels is required in many MLC methods, including the label powerset classifier [8, 9] and Bayesian networks [10]. All these methods suffer from the large complexity of the output domain, and approximate inference algorithms are required for dealing with real-world data.

6 Experimental Study

The experiments that we describe here intend to confirm our theoretical claims. To this end, we follow a similar experimental setup as in [7], in which four benchmark and two synthetic datasets with known training and test parts have been used. We extend this setup with four other datasets to emphasize the interesting computational complexity properties of our approach for high-dimensional label spaces. All eight real-world datasets were downloaded from the MULAN[6] and LibSVM[7] multi-label dataset repositories and the two synthetic datasets were generated using the description in [7].[8] All the datasets are described in Table 2.

In the experiment we show that inference by greedy search is more appropriate for estimating the joint mode, while substantial performance gains can be obtained by applying our ϵ-approximate inference algorithm. Moreover, using this strategy, we reach a computational cost that is more than fair for real-world applications. As a result, we perform a comparison of the three variants of PCC: 1) inference by greedy search for PCC, which resembles the ϵ-approximate inference algorithm to PCC with $\epsilon = 0.5$ (denoted PCC $\epsilon = 0.5$), 2) the ϵ-approximate inference algorithm with $\epsilon = 0.25$ (PCC $\epsilon = 0.25$), 3) the exact

[6] http://mulan.sourceforge.net/datasets.html

[7] http://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/datasets/multi-label.html

[8] The original training and test sets have not been published for the two synthetic datasets. We do not describe these datasets here due to space limitations, and we refer the reader to the original paper. To obtain more stable results, we report the results as an average over 5 replications of these synthetic datasets.

inference, meaning $\epsilon = 0$ (PCC $\epsilon = 0.0$). We also compare with a binary relevance (BR) learner that serves as a baseline by training a classifier for each label separately. It should perform well for the Hamming loss, while all the variants of PCC should perform well for the subset 0/1 loss. As a base learner we use a regularized logistic regression model. We apply an internal three-fold cross-validation[9] on training data for tuning the regularization parameter with possible values $\{1000, 100, 1, 0.1, 0.01, 0.001\}$. This tuning is performed for each base classifier by choosing the model with lowest empirical logistic loss in order to obtain probability estimates that are as accurate as possible.

The results are given in Table 2. We can observe that our ϵ-approximate inference works as expected: with decreasing ϵ, the subset 0/1 loss usually decreases. If this is not the case, then all the inference algorithms perform almost equally. Interestingly, the exact algorithm PCC $\epsilon = 0.0$ performs fast, being in the worse case only 2 times slower than the greedy approach. We can also observe that the greedy approach is appropriate for the subset 0/1 loss. It always obtained better results than BR for this loss, while BR is almost always better for the Hamming loss. In general, BR performs the best in estimating the marginal modes. Interestingly, for datasets with many labels and for all the algorithms, almost no difference in performance was observed on the Hamming loss, in contrast to the subset 0/1 loss.

7 Discussion

Summarizing the above theoretical and empirical results, let us conclude that our ϵ-approximate inference algorithm provides accurate and efficient estimates of the joint mode. The greedy inference algorithm, which is an integral part of the original CC algorithm, seems to be mainly tailored for subset 0/1 loss. This was not clear from the original paper.

Due to lack of space, other important issues playing a key role in chaining could not be discussed in detail. Probabilistic classifier chains can be easily extended for marginal mode estimation, leading to a general class of models that exhibit many interesting properties, such as mechanisms for parallelization, possibilities for applying different base learners, strong connections with conditional random fields and a predictive performance that is competitive with structured SVMs. We also intend to investigate in future work the effect of ensembling multiple classifiers, as considered for CC and PCC in the original papers, and the necessity for taking conditional dependence into account in marginal mode estimation, which is often put forward as the main shortcoming of binary relevance approaches.

References

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

[9] for large datasets (with number of training instances ≥ 10000) we used 66% split.

Table 2. Results on benchmark data sets, including training and test times, Hamming and subset 0/1 losses on test sets with standard errors. In bold: the best results for a given dataset and loss function. SAA = same as above.

	TRAIN TIME [S]	HAMMING LOSS	SUBSET 0/1 LOSS	TEST TIME [S]	TRAIN TIME [S]	HAMMING LOSS	SUBSET 0/1 LOSS	TEST TIME [S]
	SCENE				YEAST			
PCC ϵ=.5	420.641	0.115±.004	0.417±.014	0.375	232.249	0.213±.005	0.787±.014	0.172
PCC ϵ=.25	SAA	0.107±.004	**0.385**±.014	0.375	SAA	0.211±.006	0.764±.014	0.281
PCC ϵ=.0	SAA	0.107±.004	**0.385**±.014	0.375	SAA	0.210±.006	**0.761**±.014	0.344
BR	417.985	**0.102**±.003	0.509±.014	0.328	204.405	**0.199**±.005	0.842±.012	0.141
	MEDIAMILL				REUTERS			
PCC ϵ=.5	37202.797	0.032±.000	**0.885**±.003	41.234	15227.574	0.018±.001	0.615±.009	19.438
PCC ϵ=.25	SAA	0.032±.000	0.886±.003	53.454	SAA	**0.017**±.001	0.601±.009	21.938
PCC ϵ=.0	SAA	0.034±.000	**0.885**±.003	86.547	SAA	**0.017**±.001	**0.598**±.009	23.250
BR	16903.109	**0.030**±.000	0.902±.003	26.062	13476.883	**0.017**±.001	0.689±.008	15.359
	SYNTH1				SYNTH2			
PCC ϵ=.5	7591.826	**0.067**±.002	**0.238**±.006	15.828	26.968	**0.000**±.000	**0.000**±.000	0.735
PCC ϵ=.25	SAA	**0.067**±.002	0.239±.006	15.578	SAA	**0.000**±.000	**0.000**±.000	0.734
PCC ϵ=.0	SAA	**0.067**±.002	0.239±.006	15.735	SAA	**0.000**±.000	**0.000**±.000	0.766
BR	6955.159	**0.067**±.002	0.240±.006	12.687	16.453	0.084±.001	0.832±.004	0.609
	TMC2007-500				ENRON			
PCC ϵ=.5	21703.017	0.056±.001	0.676±.006	9.360	13387.680	0.047±.001	0.869±.014	3.547
PCC ϵ=.25	SAA	0.056±.001	0.670±.006	13.969	SAA	**0.046**±.001	0.848±.015	5.031
PCC ϵ=.0	SAA	0.056±.001	**0.668**±.006	14.359	SAA	0.047±.001	**0.845**±.015	8.907
BR	22942.510	**0.055**±.001	0.685±.006	8.312	11894.534	0.047±.001	0.886±.013	3.046
	EMOTIONS				MEDICAL			
PCC ϵ=.5	14.078	0.224±.013	0.752±.030	0.015	2613.459	0.016±.001	0.546±.020	4.407
PCC ϵ=.25	SAA	**0.219**±.013	**0.718**±.032	0.016	SAA	**0.015**±.001	**0.541**±.020	4.109
PCC ϵ=.0	SAA	0.222±.014	**0.718**±.032	0.015	SAA	**0.015**±.001	**0.541**±.020	4.172
BR	12.328	0.226±.011	0.812±.027	0.016	2337.824	0.016±.001	0.550±.020	3.110

2. Dembczyński, K., Waegeman, W., Cheng, W., Hüllermeier, E.: Regret analysis for performance metrics in multi-label classication: The case of Hamming and subset zero-one loss. In: ECML/PKDD 2010, Springer (2010)
3. Dembczyński, K., Cheng, W., Hüllermeier, E.: Bayes optimal multilabel classification via probabilistic classifier chains. In: ICML 2010, Omnipress (2010)
4. Ghamrawi, N., McCallum, A.: Collective multi-label classification. In: CIKM 2005, ACM (2005) 195–200
5. Breiman, L., Friedman, J.: Predicting multivariate responses in multiple linear regression. J R Stat. Soc. Ser. B **69** (1997) 3–54
6. Pletscher, P., Ong, C.S., Buhmann, J.M.: Entropy and margin maximization for structured output learning. In: ECML/PKDD 2010, Springer (2010)
7. Finley, T., Joachims, T.: Training structural SVMs when exact inference is intractable. In: ICML 2008, Omnipress (2008)
8. Boutell, M., Luo, J., Shen, X., Brown, C.: Learning multi-label scene classification. Pattern Recognition **37**(9) (2004) 1757–1771
9. Tsoumakas, G., Katakis, I.: Multi label classification: An overview. Int J Data Warehousing and Mining **3**(3) (2007) 1–13
10. Zhang, M.L., Zhang, K.: Multi-label learning by exploiting label dependency. In: ACM SIGKDD 2010, ACM (2010) 999–1008