Machine Learning
https://doi.org/10.1007/s10994-021-06107-2

A flexible class of dependence-aware multi-label loss functions

Eyke Hüllermeier[1] · **Marcel Wever**[2] · **Eneldo Loza Mencia**[3] · **Johannes Fürnkranz**[4] · **Michael Rapp**[3]

Received: 11 May 2021 / Revised: 14 October 2021 / Accepted: 17 October 2021

Abstract

The idea to exploit label dependencies for better prediction is at the core of methods for multi-label classification (MLC), and performance improvements are normally explained in this way. Surprisingly, however, there is no established methodology that allows to analyze the dependence-awareness of MLC algorithms. With that goal in mind, we introduce a class of loss functions that are able to capture the important aspect of label dependence. To this end, we leverage the mathematical framework of non-additive measures and integrals. Roughly speaking, a non-additive measure allows for modeling the importance of correct predictions of label subsets (instead of single labels), and thereby their impact on the overall evaluation, in a flexible way. The well-known Hamming and subset 0/1 losses are rather extreme special cases of this function class, which give full importance to single label sets or the entire label set, respectively. We present concrete instantiations of this class, which appear to be especially appealing from a modeling perspective. The assessment of multi-label classifiers in terms of these losses is illustrated in an empirical study, clearly showing their aptness at capturing label dependencies. Finally, while not being the main goal of this study, we also show some preliminary results on the minimization of this parametrized family of losses.

Keywords Multi-label classification · Loss function · Non-additive measures · Analysis · Label dependence

1 Introduction

The setting of multi-label classification (MLC), which generalizes standard multi-class classification by relaxing the assumption of mutual exclusiveness of classes, has received a lot of attention in the recent machine learning literature—we refer to Tsoumakas et al. (2010) and Zhang and Zhou (2014) for survey articles on this topic. The motivation for

Editors: Annalisa Appice, Sergio Escalera, Jose A. Gamez, Heike Trautmann.

✉ Eyke Hüllermeier
eyke@lmu.de

Extended author information available on the last page of the article

Published online: 13 January 2022

MLC originated in the field of text categorization (Hayes and Weinstein 1990; Lewis 1992; Apté et al. 1994), but nowadays multi-label methods are used in applications as diverse as music categorization (Trohidis et al. 2011), semantic scene classification (Boutell et al. 2004), protein function classification (Diplaris et al. 2005), natural language processing (Wu et al. 2019), and remote sensing (Yessou et al. 2020).

Formally, the task of a multi-label classifier is to assign a subset of a given set of candidate labels to any query instance. A straightforward approach for learning such a predictor is via a reduction to binary classification, i.e., by training one binary classifier per label and combining the predictions of these classifiers into an overall multi-label prediction. This approach, known as *binary relevance* (BR) learning, is often criticized for ignoring possible label dependencies, because each label is predicted independently of all other labels. Indeed, the idea of exploiting statistical dependencies between the labels in order to improve predictive performance on the level of the entire label set is a major theme in research on multi-label classification, and many MLC methods proposed in the literature are motivated by this idea.

Of course, the usefulness of such methods very much depends on whether or not such dependencies are indeed present in the data. Besides, it turns out that the underlying loss function used to evaluate predictions plays an important role (Dembczynski et al. 2012). Since a subset of predicted labels can be compared with a ground-truth subset in many ways, various loss functions have been proposed in the literature. Two simple but commonly used examples are the Hamming and the subset 0/1 loss. While the former assesses the quality of predictions in terms of the percentage of incorrectly predicted labels, the latter merely checks whether the entire label subset is predicted correctly or not. As will be explained in more detail later on, capturing label dependencies is crucial for performing well in terms of the subset 0/1 loss, but—at least in theory—not necessary in the case of the Hamming loss, which evaluates each label prediction independently of all others.

Despite their widespread use and interesting theoretical properties, both Hamming and subset 0/1 can be criticized for various reasons, especially for being rather extreme (Dembczynski et al. 2010). The Hamming loss is often close to 0, simply because the label cardinality (average percentage of relevant labels per example) is very small in typical MLC data sets. Thus, even the default classifier that predicts all labels as irrelevant will usually perform well according to Hamming loss, and is indeed often difficult to beat. Even if an improvement is possible, the performance differences are typically small, and therefore difficult to test for statistical significance. On the other side, the subset 0/1 loss is typically quite high and may appear overly stringent, especially in the case of many labels. Moreover, since a mistake on a single label incurs the same penalty as erring on all predicted labels, it does not discriminate well between "almost correct" and completely wrong predictions. For these reasons, the evaluation of MLC algorithms in terms of Hamming loss or subset 0/1 loss is arguable and often of little practical relevance.

In this paper, we introduce a new class of MLC loss functions that are able to capture the important aspect of label dependence in a sophisticated and controllable manner. In a sense, these losses "interpolate" between Hamming and subset 0/1 loss. Although they could play the role of the target loss to be optimized by the learner, or could be used as a surrogate loss facilitating the optimization of another target loss (as will also be seen in Sect. 6), their main purpose (at least from our perspective) is different: we consider them as a tool for *analyzing* MLC algorithms, helping to gain insight into their learning behavior and their (alleged) "dependence-awareness".

The construction of our loss functions is based on the mathematical framework of non-additive measures and integrals (Sect. 3). Roughly speaking, the overall loss is obtained by integrating over the errors on individual labels. This integration is done with respect to a non-additive measure, which allows for controlling the dependence-awareness of the loss, i.e., for modeling the importance of correct predictions of label subsets. In Sect. 4, we present concrete instantiations of this type of loss functions, which allow for controlling dependence-awareness by means of a single parameter. The assessment of the dependence-awareness of multi-label classifiers in terms of these losses is illustrated in an empirical study (Sect. 5). Finally, in Sect. 6, we also show first results that illustrate the effects on minimizing this family of losses for different parametrizations.

2 Multi-label classification

Let $\mathcal{X}$ denote an instance space, and let $\mathcal{L} = \{\lambda_1, \dots, \lambda_K\}$ be a finite set of class labels. We assume that an instance $\boldsymbol{x} \in \mathcal{X}$ is (probabilistically) associated with a subset of labels in $\mathcal{L}$, often called the set of relevant labels, while the complement is considered as irrelevant for $\boldsymbol{x}$. We identify the subset with a binary vector $\boldsymbol{y} = (y_1, \dots, y_K) \in \mathcal{Y} = \{0, 1\}^K$, in which $y_k = 1$ indicates relevance of λ_k.

We assume observations to be realizations of random variables generated independently and identically (i.i.d.) according to a probability measure P on $\mathcal{X} \times \mathcal{Y}$ (with density/mass function p), i.e., an observation $\boldsymbol{y} = (y_1, \dots, y_K)$ is the realization of a corresponding random vector $\mathbf{Y} = (Y_1, \dots, Y_K)$. We denote by $p(\mathbf{Y} \,|\, \boldsymbol{x})$ the conditional distribution of $\mathbf{Y}$ given $\mathbf{X} = \boldsymbol{x}$, and by $p_i(Y_i \,|\, \boldsymbol{x})$ the corresponding marginal distribution of the ith label Y_i, i.e.,

$$p_i(b \,|\, \boldsymbol{x}) = \sum_{\boldsymbol{y} \in \mathcal{Y} : y_i = b} p(\boldsymbol{y} \,|\, \boldsymbol{x}) \tag{1}$$

for $b \in \{0, 1\}$. Given training data in the form of a finite set of observations

$$\mathcal{D} = \{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^{N} \subset \mathcal{X} \times \mathcal{Y}, \tag{2}$$

drawn independently from $P(\mathbf{X}, \mathbf{Y})$, the goal in MLC is to learn a predictive model that generalizes well beyond these observations, i.e., which yields predictions that minimize the expected risk with respect to a specific loss function.

2.1 Predictive models in MLC

A multi-label classifier $\boldsymbol{h}$ is a mapping $\mathcal{X} \longrightarrow \mathcal{Y}$ that assigns a (predicted) label subset to each instance $\boldsymbol{x} \in \mathcal{X}$. Thus, the output of a classifier $\boldsymbol{h}$ is a vector

$$\boldsymbol{h}(\boldsymbol{x}) = (h_1(\boldsymbol{x}), \dots, h_K(\boldsymbol{x})) \in \{0, 1\}^K . \tag{3}$$

In the following, predictions of this kind will also be denoted as $\hat{\boldsymbol{y}} = (\hat{y}_1, \dots, \hat{y}_K)$.

2.2 MLC loss functions

In the literature, various MLC loss functions have been proposed (Wu and Zhou 2017). Commonly used are the Hamming loss ℓ_H and the subset 0/1 loss ℓ_S, which both generalize the standard 0/1 loss for multi-class classification:

$$\ell_H(\boldsymbol{y},\hat{\boldsymbol{y}}) = \frac{1}{K}\sum_{k=1}^{K} [\![y_k \neq \hat{y}_k]\!], \tag{4}$$

$$\ell_S(\boldsymbol{y},\hat{\boldsymbol{y}}) = [\![\boldsymbol{y} \neq \hat{\boldsymbol{y}}]\!], \tag{5}$$

where $[\![\cdot]\!]$ is the indicator function, i.e., $[\![A]\!] = 1$ if the predicate A is true and $= 0$ otherwise. Besides, other performance metrics are often reported in experimental studies, for example the F-measure.

2.3 Label dependence

Predictions of an MLC classifier can often be regarded as an approximation of the conditional probability $p(Y = \hat{y} \,|\, \boldsymbol{x})$, i.e., the probability that $\hat{y}$ is the true label for the given instance $\boldsymbol{x}$. The idea to improve such predictions by capturing dependencies between $Y_1, \ldots, Y_K$ is a driving force in research on MLC. In this regard, a distinction between *unconditional* and *conditional independence* of labels can be made (Dembczynski et al. 2012). In the first case, the joint distribution $p(\mathbf{Y})$ in the label space factorizes into the product of the marginals $p(Y_k)$, i.e.,

$$p(\mathbf{Y}) = p(Y_1) \times \cdots \times p(Y_K)\,,$$

whereas in the latter case, the factorization

$$p(\mathbf{Y} \,|\, \boldsymbol{x}) = p(Y_1 \,|\, \boldsymbol{x}) \times \cdots \times p(Y_K \,|\, \boldsymbol{x})$$

holds conditioned on $\boldsymbol{x}$, for every instance $\boldsymbol{x}$. In other words, unconditional dependence is a kind of global dependence, whereas conditional dependence is a dependence locally restricted to a single point in the instance space.

2.4 Decomposable loss functions

It turns out that there is a close connection between label dependence and the *decomposability* of loss functions: a decomposable loss can be expressed in the form

$$\ell(\boldsymbol{y},\hat{\boldsymbol{y}}) = \sum_{k=1}^{K} \ell^{(k)}(y_k, \hat{y}_k) \tag{6}$$

with suitable binary loss functions $\ell^{(k)} : \{0,1\}^2 \longrightarrow \mathbb{R}$ for each label y_k, whereas a non-decomposable loss does not permit such a representation. It can be shown that knowledge about the marginals $p_k(Y_k \,|\, \boldsymbol{x})$ is sufficient for producing loss minimizing predictions $\hat{\boldsymbol{y}} = \boldsymbol{h}(\boldsymbol{x})$ in the decomposable case but not in the case of a non-decomposable loss (Dembczynski et al. 2012). Instead, if a loss is non-decomposable, higher-order probabilities are needed, and in the extreme case even the entire distribution $p(\mathbf{Y} \,|\, \boldsymbol{x})$ (as in the case of the subset 0/1 loss). On an algorithmic level, this means that MLC with a decomposable loss can be tackled by *binary relevance* (BR) learning (i.e., learning one binary classifier for each label individually), whereas non-decomposable losses call for more sophisticated learning methods that take label dependencies into account.

3 Multi-label loss functions based on non-additive measures

The Hamming and the subset 0/1 loss are often considered as prototypical examples of losses which, respectively, do and do not impel the learner to take label dependencies into account: Hamming is label-wise decomposable and can principally be optimized by learning algorithms like BR. The subset 0/1 loss, on the other side, is not label-wise decomposable. Therefore, this loss is often used to quantify the learner's ability or propensity to capture label dependencies. For example, consider the following (conditional) ground-truth distribution $p(\cdot \,|\, \boldsymbol{x})$ on the label space $\mathcal{Y} = \{0, 1\}^3$:

$\boldsymbol{y}$	(0, 0, 0)	(1, 1, 1)	(0, 1, 1)	(1, 0, 1)	(1, 1, 0)
$p(\boldsymbol{y} \,|\, \boldsymbol{x})$	$1/4$	$3/16$	$3/16$	$3/16$	$3/16$

For each of the three labels, the individual probability of relevance ($9/16$) is higher than the probability of irrelevance, and indeed, in the case of Hamming, $\hat{\boldsymbol{y}} = (1, 1, 1)$ is the Bayes-optimal prediction (minimizing the loss in expectation). For the subset 0/1 loss, however, the Bayes-optimal prediction is $\hat{\boldsymbol{y}} = (0, 0, 0)$. In general, the Bayes-optimal prediction is given by the *marginal mode* of the distribution $p(\cdot \,|\, \boldsymbol{x})$ in the case of Hamming and by the *joint mode* in the case of subset 0/1.

Both Hamming and subset 0/1 can be criticized for being rather extreme: the Hamming loss is often very low, because it does not take any dependencies into account, and the distribution between relevant and irrelevant instances for each label is quite unbalanced. Therefore, even a classifier that only predicts the absence of all labels may have a low Hamming loss. Conversely, the subset 0/1 loss is normally quite high, since an entirely correct prediction becomes very unlikely with increasing K. It is an "all or nothing" measure, for which a mistake on a single label is as bad as a mistake on many labels, and which does not reward correct predictions on larger subsets of the labels.

In the following, we introduce a new class of loss functions for MLC, which allows for a more gradual assessment of an algorithm's ability to model label dependencies. These loss functions are able to assess a learner's dependence-awareness, i.e., its aptness at capturing label dependencies, in a more flexible way. To this end, we leverage the mathematical framework of non-additive measures and integrals, the essentials of which are recalled in Sects. 3.2 and 3.3. Roughly speaking, a non-additive measure is used for modeling the importance of correct predictions of label subsets (instead of single labels), and thereby their impact on the overall evaluation (cf. Sect. 3.4). Hamming and subset 0/1 will be recovered as special ("boundary") cases of our family.

3.1 General idea

For didactic reasons, let us anticipate the basic construction principle of our family of loss functions, which will be introduced step by step, alongside with several other (auxiliary) functions. To this end, let us assume that an MLC classifier produces predictions in the form of scores $s_i \in [0, 1]$ for the individual labels λ_i—this covers binary predictions $s_i \in \{0, 1\}$ as a special case, though most methods produce real-valued scores in a first step, for example probability estimates, which are then turned into binary relevance predictions in the end (e.g., via thresholding).

Given a score s_i indicating the predicted relevance of the label λ_i and the ground truth $y_i \in \{0,1\}$, it is natural to consider the difference $|s_i - y_i|$ between these two as a measure of incorrectness of the prediction, or, vice versa, $1 - |s_i - y_i|$ as a measure of correctness. Considering these degrees of correctness of predictions on individual labels as evaluation criteria c_i, a loss $\ell_\mu(\boldsymbol{y}, \boldsymbol{s})$ will be defined as a suitably weighted aggregation (namely as an integral) of the correctness degrees

$$f(c_i) = 1 - |s_i - y_i| \in [0,1]\,, \tag{7}$$

Thus, $f(c_i) \in [0,1]$ can be seen as the degree to which the criterion c_i, being correct on the ith label, is fulfilled. Allowing for predictions in terms of a score vector $\boldsymbol{s} = (s_1, \ldots, s_K) \in [0,1]^K$ is more general than a binary prediction $\hat{\boldsymbol{y}} = (\hat{y}_1, \ldots, \hat{y}_K) \in \{0,1\}^k$, but obviously comprises the latter as a special case. Moreover, such kind of "soft" predictions are produced by many MLC algorithms. The loss will then be specified in terms of an integral of the "correctness function" f given by (7), i.e., as an aggregated (in-)correctness

$$\ell_\mu(\boldsymbol{y}, \boldsymbol{s}) = 1 - \int f \, d\mu\,, \tag{8}$$

To this end, two main ingredients, the measure μ for weighting and the integral for aggregation, are needed:

- A *measure* μ assigns a weight $\mu(A)$ to every subset A, in our case to a subset of labels, which can be interpreted as the importance of that subset. Formally, a measure μ is a mapping from subsets to the unit interval, which can be equivalently represented by its *Moebius transform* m_μ [see (10) below].
- The aggregation in (8) is accomplished with the so-called (discrete) *Choquet integral* $\mathcal{C}_\mu$, which is a weighted aggregation of the values of a function (here f) with respect to the underlying measure μ.

The Choquet integral has already been used as a mathematical tool in machine learning, for example in multi-class classification (Tehrani et al. 2012a) and preference learning (Tehrani et al. 2012b), but also in the context of MLC (Tehrani and Ahrens 2017). In all these cases, however, it has been used as a model for representing the predictor, for example a classifier h, that is, as a function to aggregate the input features $\boldsymbol{x}$. We are not aware of any work on leveraging non-additive measures and integrals for defining MLC loss functions, i.e., for applying it as an aggregation function on the predictions (rather than the features). In the following, we discuss the components on the right-hand side of (8) in more detail, for the specific case of MLC.

3.2 Non-additive measures

Let $C = \{c_1, \ldots, c_K\}$ be a finite set of (desirable) "criteria" and $\mu : 2^C \longrightarrow [0,1]$ a measure on this set. For each $A \subseteq C$, we interpret $\mu(A)$ as the *weight* or the *importance* of the subset of criteria A. In the context of MLC, we can think of a criterion c_i as the correctness of the prediction on the ith label λ_i. Thus, $\mu(\{c_1\})$ is the importance of predicting the first label correctly, and $\mu(\{c_1, c_2\})$ is the importance of *jointly* predicting the first and the second label correctly.

A standard assumption on a measure μ, which is at the core of probability theory, is *additivity*: $\mu(A \cup B) = \mu(A) + \mu(B)$ for all $A, B \subseteq C$ such that $A \cap B = \emptyset$. Unfortunately, additive measures cannot model any kind of "interaction": extending a set of elements A by a set of elements B always increases the weight $\mu(A)$ by the weight $\mu(B)$, regardless of A and B. For example, we cannot express that predicting λ_1 and λ_2 correctly, i.e., both together, has a higher value than the sum of getting both of them individually right, simply because additivity implies $\mu(\{\lambda_1, \lambda_2\}) = \mu(\{\lambda_1\}) + \mu(\{\lambda_2\})$.

Non-additive measures, also called capacities or fuzzy measures, are simply normalized and monotone, but not necessarily additive (Sugeno 1974):

$$\begin{aligned} &\mu(\emptyset) = 0, \mu(C) = 1, \text{ and} \\ &\mu(A) \leq \mu(B) \text{ for all } A \subseteq B \subseteq C \, . \end{aligned} \tag{9}$$

Thus, a set of criteria B is always at least as important as any of its subsets. Yet, the way in which $\mu(B)$ differs from $\mu(A)$ is not immediately determined by $\mu(B \setminus A)$.

A useful representation of non-additive measures is in terms of the *Moebius transform*:

$$\mu(B) = \sum_{A \subseteq B} \mathrm{m}_\mu(A) \tag{10}$$

for all $B \subseteq C$, where the Moebius transform m_μ of the measure μ is defined as follows:

$$\mathrm{m}_\mu(A) = \sum_{B \subseteq A} (-1)^{|A|-|B|} \mu(B) \, . \tag{11}$$

3.3 The Choquet integral

Suppose that $f : C \longrightarrow \mathbb{R}_+$ is a non-negative function that assigns a *value* to each criterion c_i. As already said, in the case of MLC, we can think of $f(c_i)$ as the correctness of a prediction on the label λ_i. An important question, then, is how to *aggregate* the evaluations of individual criteria, i.e., the values $f(c_i)$, into an overall evaluation, in which the criteria are properly weighted according to the measure μ. Mathematically, this overall evaluation can be considered as an integral $\mathcal{C}_\mu(f)$ of the function f with respect to the (possibly non-additive) measure μ. Indeed, if μ is an additive measure, the standard integral just corresponds to the *weighted mean*

$$\mathcal{C}_\mu(f) = \sum_{i=1}^{K} w_i \cdot f(c_i) = \sum_{i=1}^{K} \mu(\{c_i\}) \cdot f(c_i) \, , \tag{12}$$

which is a natural aggregation operator in this case. For example, in the context of MLC, the Hamming loss is a special case of (12), with $f(c_i) \in \{0, 1\}$ depending on whether the prediction on λ_i is right or wrong, and uniform weights $w_i = 1/K$.

The question of how to generalize (12) and define the integral of a function with respect to a *non-additive measure* is answered by the *Choquet integral* (Choquet 1954), which, in the discrete case, is formally defined as follows:

$$\mathcal{C}_\mu(f) = \sum_{i=1}^{K} \left(f(c_{(i)}) - f(c_{(i-1)}) \right) \cdot \mu(A_{(i)}) \, ,$$

where $(\cdot)$ is a permutation of $[K]$ such that $0 \leq f(c_{(1)}) \leq f(c_{(2)}) \leq \ldots \leq f(c_{(K)})$ (and $f(c_{(0)}) = 0$ by definition), and $A_{(i)} = \{c_{(i)}, \ldots, c_{(K)}\}$. It can also be shown that, in terms of the Moebius transform of μ, the Choquet integral can also be expressed as follows (see e.g. Klement et al. 2002):

$$\mathcal{C}_\mu(f) = \sum_{T \subseteq C} \mathrm{m}_\mu(T) \times \min_{i \in T} f(c_i) \,. \tag{13}$$

3.4 MLC losses based on non-additive measures

In the context of MLC, non-additive measures and generalized integrals can be used to define flexible loss functions: Each criterion c_i corresponds to the (correct) prediction on a label λ_i, and $\mu(A)$ quantifies the importance to be correct on the subset of labels A *as a whole*. Moreover, the function to be integrated is the correctness function (7). Thus, $u_i = f(c_i) = 1 - |s_i - y_i|$ is the degree of correctness on the label λ_i, where $s_i \in [0,1]$ is the score predicted for label λ_i and $y_i \in \{0,1\}$ the corresponding ground truth: $u_i = 1$ for a perfectly correct prediction and $u_i = 0$ for a wrong prediction.

Now, given the u_i as values on the criteria c_i (the higher the better), the idea is to aggregate these values with the Choquet integral (based on the measure μ) into an overall degree of correctness, and to define a loss as the complement $(1 - (\cdot))$ of this degree of correctness. Formally, this leads to the following loss function, wherein the permutation $(\cdot)$ is such that $0 \leq u_{(1)} \leq u_{(2)} \leq \ldots \leq u_{(K)}$, and $A_{(i)} = \{c_{(i)}, \ldots, c_{(K)}\}$:

$$\ell_\mu(\boldsymbol{y}, \boldsymbol{s}) = 1 - \sum_{i=1}^{K} \left(u_{(i)} - u_{(i-1)}\right) \cdot \mu\left(A_{(i)}\right) \,. \tag{14}$$

Note that, given random access to the measure μ, this loss can be computed in time $O(K \log K)$.

4 Concrete instantiations of the framework

In the following, we show that important existing MLC losses can be recovered as special cases of (14). Moreover, we introduce two parametrized families of loss functions based on so-called counting measures.

4.1 Special cases

The Hamming loss and the subset 0/1 loss, which are commonly used measures that represent the extreme ends of dependence-awareness, can be obtained as special cases of (14):

- The *Hamming loss* results from the additive measure $\mu(A) = |A|/K$, for which the sum in (14) takes the form

  $K^{-1}(Ku_{(1)} + (K-1)(u_{(2)} - u_{(1)}) + \ldots + 1(u_{(K)} - u_{(K-1)})) = K^{-1} \sum_i u_{(i)} = K^{-1} \sum_i u_i$,

  and hence

$$\ell_\mu(\boldsymbol{y},\boldsymbol{s}) = 1 - \frac{1}{K}\sum_{i=1}^{K} u_i = \frac{1}{K}\sum_{i=1}^{K} |s_i - y_i| \, .$$

Strictly speaking, allowing for real-valued scores $s_i \in [0,1]$, we obtain a generalization of the Hamming loss.

- The *subset 0/1 loss* results from

$$\mu(A) = \begin{cases} 1 & \text{if } A = C \\ 0 & \text{if } A \subsetneq C \end{cases} ,$$

for which the sum (14) takes the form $\sum_{i=1}^{K} \left(u_{(i)} - u_{(i-1)}\right) = u_{(1)} = \min_i u_i$, and hence

$$\ell_\mu(\boldsymbol{y},\boldsymbol{s}) = 1 - \min_{1 \le i \le K} u_i = \max_{1 \le i \le K} |y_i - s_i| \, ,$$

which, in case $\boldsymbol{s} \notin \{0,1\}^K$, is again a generalization.

Another interesting special case is the *covering error* introduced by Amit et al. (2007), which is defined as the sum of subset 0/1 losses on a family of predefined label subsets, called a covering. The connection to this loss can nicely be seen based on the representation (13) of the Choquet integral in terms of the Moebius transform. Here, the min -terms correspond to subset 0/1 losses on subsets T. In contrast to the covering error, where these losses are weighted equally, they are weighted by the values of the Moebius function in our case.

4.2 Counting measures

The two measures above are examples of so-called *counting measures*, which only depend on the cardinality of A. In other words, μ is a counting measure if it can be expressed as $\mu(A) = v({}^{|A|}/_{K})$ for a suitable function $v : [0,1] \longrightarrow [0,1]$, which means that the measure of a set only depends on its cardinality but not on the elements of the set. For example, $\mu(\{c_1, c_2\}) = v(2/K) = \mu(\{c_3, c_4\})$. This kind of symmetry property is certainly meaningful in MLC, where the different labels are normally considered as equally important – or, stated differently, the performance metric is normally invariant under permutation of the labels. Here, $v(k/K)$ can be interpreted as the importance of a correct prediction on a subset of k labels, which means that the loss function (14) is completely specified by the values

$$0 = v(0), v({}^{1}/_{K}), \ldots, v(1) = 1 \, .$$

Obviously, compared to the general case, this is an enormous reduction from an exponential to a linear number of degrees of freedom that need to be specified.

Formally, for an increasing function $v : [0,1] \longrightarrow [0,1]$ such that $v(0) = 0$ and $v(1) = 1$, we obtain an OWA (ordered weighted averaging (Yager and Filev 1999; Yager and Kacprzyk 2012)) aggregation of the degrees of correctness u_i, namely

$$\sum_{i=1}^{K} w_i \cdot u_{(i)} \tag{15}$$

with

$$w_i = v\left(\frac{K-i+1}{K}\right) - v\left(\frac{K-i}{K}\right).$$

In other words, we obtain an OWA loss function

$$\ell_\mu(\boldsymbol{y},\boldsymbol{s}) = \sum_{i=1}^{K} w_i \cdot |y_{(i)} - s_{(i)}| \tag{16}$$

with $w_1 + \ldots + w_K = 1$. Again, Hamming is obtained for the special case $v : x \mapsto x$ and subset 0/1 for v such that

$$v(x) = \begin{cases} 1 & \text{if } x = 1 \\ 0 & \text{otherwise} \end{cases}.$$

Let us highlight that, in spite of a somewhat involved derivation (based on non-additive measures and integrals) and the flexibility our class of loss functions in general, the form (16) we end up with in the case counting measures is both intuitively appealing and easy to compute. In principle, it is nothing than a weighted average of the errors on individual labels, with the important difference that the weights w_i now pertain, not to the ith label, but to the ith order statistic of the error, i.e., the ith largest error. Let us illustrate this with a simple example, in which the ground-truth labeling is $\boldsymbol{y} = (0, 1, 1, 0, 0, 0)$ and the prediction $\boldsymbol{s} = (0.2, 0.3, 0.9, 0.1, 0.4, 0.3)$. Here, the errors on the individual labels are given, respectively, by 0.2, 0.7, 0.1, 0.1, 0.4, 0.3. Sorting these from lowest to highest yields the increasing sequence 0.1, 0.1, 0.2, 0.3, 0.4, 0.7. Different weight vectors $\boldsymbol{w}$ will then emphasize different values in this sequence and hence yield different losses, for example:

Error	0.1	0.1	0.2	0.3	0.4	0.7	$\ell_\mu(\boldsymbol{y},\boldsymbol{s})$
Weight	1/6	1/6	1/6	1/6	1/6	1/6	0.30
Weight	0	0	0	0	0	1	0.70
Weight	0	1/15	2/15	3/15	4/15	5/15	0.43

The first case with uniform weights corresponds to Hamming loss and yields a simple averaging of the errors. In the second case, the full weight is given to the largest error, which corresponds to the subset 0/1 loss. The third case is in-between these two extremes.

Let us also note that the computation is further simplified in the case of binary predictions, i.e., where the scores s_i and hence also the individual errors are either 0 or 1. In this case, the loss merely depends on the total number of errors k, and is given by

$$\ell_\mu(\boldsymbol{y},\boldsymbol{s}) = \sum_{i=K-k+1}^{K} w_i,$$

i.e., by the sum of the k largest weights.

4.3 Parametrized families

In the following, we present two families of loss functions (16), which allow for modeling the dependence-awareness in terms of a single parameter.

- *Polynomial loss*: First, one could think of using a convex function of the form

$$v : x \mapsto x^{\alpha} \tag{17}$$

for $\alpha \geq 1$. The larger α, the more important it becomes to predict larger subsets correctly, and subset 0/1 is recovered for the limit case $\alpha \to \infty$. In other words, α can be used to smoothly interpolate between Hamming and subset 0/1.

- *Binomial loss*: To motivate a second family, suppose we are only interested in getting k-subsets of labels right, whereas a correct prediction on a subset of size $< k$ should not be rewarded. This could be reflected by a Moebius function

$$m(A) = \begin{cases} 1/\binom{K}{k} & \text{if } |A| = k \\ 0 & \text{otherwise} \end{cases}$$

In this case, we obtain

$$v\left(\frac{j}{K}\right) = \frac{\binom{j}{k}}{\binom{K}{k}}. \tag{18}$$

Again, the Hamming and subset 0/1 loss can be recovered by setting, respectively, $k = 1$ and $k = K$, while interpolations are obtained in-between.

In principle, non-symmetric measures could of course be used in MLC as well, for example to express that different labels or different label subsets are of different importance. Yet, as already said, symmetry appears to be a natural property. Moreover, as it significantly reduces the number of degrees of freedom, this property facilitates the specification of a measure-based loss function (14).

Nevertheless interesting would be a weighting of label subsets in proportion to the number of relevant labels they contain. More concretely, starting from a "base measure" μ, the Moebius mass $m_{\mu}(A)$ could be adjusted depending on the number of relevant labels in A—increased if A contains many and reduced if it contains only few relevant labels. Thereby, more emphasis could be put on correct predictions for relevant labels. The resulting loss function would then depend on the ground truth $\boldsymbol{y}$. We leave this generalization for future work.

4.4 Examples

We end this section with a few examples illustrating the different roles that generalized losses may play, namely, the role of an analytic tool (first example), as a target loss (second example), and as a surrogate loss (third example). As a family of losses, we consider the binomial loss parametrized by k, i.e., the loss defined by (16) with v given by (18).

As a first simple example consider a 3-label problem, where (for a given instance $\boldsymbol{x}$) the labels are either all relevant ($\boldsymbol{y} = (1, 1, 1)$) or all irrelevant ($\boldsymbol{y} = (0, 0, 0)$). Suppose that learner A trains a binary predictor for each label individually and achieves an accuracy of 0.8 on each of the three problems. Learner B, on the other side, is trained using the label powerset approach (Tsoumakas et al. 2010) and only predicts (1, 1, 1) or (0, 0, 0), providing the right answer with probability 0.5. Obviously, learner B is more dependence-aware than A. Yet, assuming B's predictions are independent, both have a subset 0/1 loss around

0.5, and A even a bit less. This example clearly shows that, to compare learners in terms of their dependence-awareness, it is not enough to look at a single loss measure: a learner may perform quite well in terms of subset 0/1 because it succeeds at capturing dependencies, but also because it performs strong on each label individually (while ignoring any dependencies). As an aside, the example also suggests that capturing dependencies does not automatically imply improvements, also because it normally complicates the learning task (see also our third example below).

More insight is gained by looking at several losses simultaneously. For learner B, the subset 0/1 loss coincides with its Hamming loss and also with the binomial loss for $k = 2$, simply because learner B is either completely right or completely wrong on all labels. That is, learner B is very stable over the entire spectrum of losses. Looking at the performance of learner A in terms of the binomial loss, we get 0.2 for $k = 1$ (Hamming) and 0.32 for $k = 2$. Thus, increasing k from 1 to 3, we obtain different *performance profiles* (sequence of loss values) for the two learners:

$$
\begin{aligned}
A &: (0.2, 0.32, 0.488) \\
B &: (0.5, 0.5, 0.5)
\end{aligned}
$$

Comparing these profiles provides much more information and clearly shows that learner A, albeit better in absolute performance, is less dependence-aware than B: its performance is more affected when increasing the dependence-awareness of the loss, and it loses performance relative to B. As shown by this example, what is really informative is to look at how a loss *changes*, both in absolute terms but also relative to the loss of another learner, when varying the dependence-awareness. This is why we consider parametrized families like those introduced above to be so useful. We shall elaborate on this way of using generalized losses in the experiments in Sect. 5.

The second example is simply meant to show that changing the target (loss) may also change the (Bayes-)optimal prediction. In principle, this is of course nothing new, and we have already seen an example where the loss minimizer for Hamming is the complete opposite of the minimizer for subset 0/1. Yet, we would like to show that, by varying the dependence-awareness, corresponding effects can also be achieved "in-between". To this end, consider the distribution on labelings $\boldsymbol{y}$ (given an instance $\boldsymbol{x}$) shown in Table 1. One can verify (e.g., through simple enumeration) that the Bayes-optimal predictions for the binomial loss with different parameters k are given as follows:

$k = 1$	1	0	0	1	0
$k = 2$	0	0	1	1	1
$k = 3$	0	0	1	1	1
$k = 4$	1	1	0	0	0
$k = 5$	1	0	1	1	0

As can be seen, by changing the parameter of the loss, the optimal prediction may change quite drastically. For example, the prediction of three of the five labels changes when going from $k = 1$ to $k = 2$, and even all five labels change when passing from $k = 3$ to $k = 4$.

The previous example has shown that varying the degree of label-awareness may serve as an incentive for the learner to produce different predictions. For this reason, we argue that a generalized loss may also be useful as a surrogate loss. Imagine, for example, that we are interested in minimizing subset 0/1 loss (as a target) in a problem with 5 labels.

Table 1 Hypothetical probability distribution over labels $y_1, \dots, y_5$ for a given instance $\boldsymbol{x}$

y_1	y_2	y_3	y_4	y_5	$p(\boldsymbol{y} \mid \boldsymbol{x})$
0	0	0	0	0	0.046
0	0	0	0	1	0.003
0	0	0	1	0	0.034
0	0	0	1	1	0.048
0	0	1	0	0	0.025
0	0	1	0	1	0.052
0	0	1	1	0	0.036
0	0	1	1	1	0.050
0	1	0	0	0	0.022
0	1	0	0	1	0.011
0	1	0	1	0	0.006
0	1	0	1	1	0.059
0	1	1	0	0	0.041
0	1	1	0	1	0.023
0	1	1	1	0	0.013
0	1	1	1	1	0.012
1	0	0	0	0	0.044
1	0	0	0	1	0.023
1	0	0	1	0	0.018
1	0	0	1	1	0.011
1	0	1	0	0	0.003
1	0	1	0	1	0.022
1	0	1	1	0	0.062
1	0	1	1	1	0.054
1	1	0	0	0	0.056
1	1	0	0	1	0.059
1	1	0	1	0	0.040
1	1	0	1	1	0.022
1	1	1	0	0	0.029
1	1	1	0	1	0.013
1	1	1	1	0	0.038
1	1	1	1	1	0.025

Moreover, suppose the problem is such that we manage to train a predictor that is correct with probability 0.9 on individual labels, for example using binary relevance learning and hence Hamming as a surrogate loss. When training with the label powerset method on label-quintuples (i.e., solving a classification problem with 32 classes), a correctness of at most 0.4 can be achieved. Now, the subset 0/1 loss is ≈ 0.4 in the first case (assuming independence of the predictions) and 0.6 in the third case, showing that binary relevance learning might be better than label powerset, despite the dependence-awareness of the target loss. The case would be different, of course, with a correctness of only 0.8 instead of 0.9 on the binary problems. Thus, whether it is better to incentivize the learner to perform well on single labels or on the entire set of 5 labels strongly depends on how manageable these tasks are. In the first case, the learner ends up with simpler tasks but will tend to ignore

Table 2 Overview of datasets with statistics of their main properties

Dataset	#Instances	#Labels	Label-to-Instance Ratio	Unique Label Combinations	Cardinality
Birds	645	19	0.0295	133	1.01
Emotions	593	6	0.0101	27	1.87
Enron-f	1702	53	0.0311	753	3.38
Flags	194	12	0.0619	103	4.12
Genbase	662	27	0.0408	32	1.25
Llog-f	1460	75	0.0514	304	1.18
Medical	978	45	0.0460	94	1.25
Scene	2407	6	0.0025	15	1.07
Yeast	2417	14	0.0058	198	4.24

label-dependencies. These dependencies are accounted for in the second case, but this may not be helpful if it makes the problem unduely difficult to solve. Quite plausibly, the optimal compromise is somewhere in-between these two extremes, so that training the learner with a properly parametrized generalized loss (as a surrogate) might be the best way to go. Using such a loss, it is possible to incentivize the learner to produce correct predictions on label subsets of different size, so that a good balance between dependence-awareness and manageability of the problem might be achieved. We shall elaborate on the use of generalized losses as surrogates (and targets) for learning in Sect. 6.

5 An analysis of the dependence-awareness of MLC algorithms

In this section, we showcase how the proposed class of multi-label loss functions can be applied as an analysis tool for capturing the “dependence-awareness” of different multi-label classifiers, i.e., for assessing a learner’s ability or propensity to capture label dependence.[1] Actually, the study should mainly be conceived as an evaluation of our new loss functions and less as an evaluation of the methods. In fact, for the methods we analyze, we already have quite a good idea of their dependence-awareness. What we are interested in is whether this is also reflected by the loss, i.e., whether the loss is coherent with our expectations. In other words, the study is mainly intended as a proof of concept, showing that the loss functions behave as they are supposed to do.

5.1 Experimental setup

For the experimental study, we used nine benchmark datasets originating from a variety of different domains.[2] Table 2 provides an overview of the considered datasets together with

[1] The code of the experiments presented in this and the next section is provided at https://github.com/KIuML/DependenceAwareMLCLoss. An implementation of Boomer is available at https://github.com/mrapp-ke/Boomer.

[2] The datasets are available at http://mulan.sf.net/datasets-mlc.html and https://waikato.github.io/meka/datasets/.

their statistical properties. This includes the number of instances, the number of labels, the ratio of number of labels to number of instances, the absolute number of unique label combinations, and the average number of relevant labels per instance, also referred to as label cardinality.

We use paired 10-fold cross-validations to obtain out-of-sample predictions in the form of label relevance scores. Although we restrict our analysis to binary predictions $s_i \in \{0, 1\}$ in order to isolate from the ability of the classifiers to shape their scores, our methodology is in principle also suitable for comparing algorithms that output soft predictions $s_i \in [0, 1]$ and independent of the thresholding technique used.

5.2 Algorithms

In order to analyze their differences w.r.t. dependence-awareness, we experiment with several publicly available multi-label algorithms:

- *Binary relevance (BR)* is a reduction to binary classification, which learns one binary classifier for each label independently of the others (Boutell et al. 2004; Zhang et al. 2018). Despite its simplicity, BR has proven to be highly competitive in comparison to state-of-the-art multi-label learners in recent studies, especially regarding measures that are not dependence-aware (cf., e.g., Rivolli et al. 2020; Wever et al. 2020, 2018).
- *Classifier chains (CC)* take label dependencies into account, by imposing an order on the label set and using the predictions for the previous labels as additional feature information for the next label predictor (Read et al. 2009, 2021).
- *Label powerset (LP)* is a reduction to multi-class classification (Tsoumakas et al. 2010). It converts each possible label subset into a separate (meta-)class and then solves a standard classification problem. Thereby, it takes label dependence into account, though at the expense of treating similar label sets as independent classes.
- *Random k-labelsets (RAkEL)* randomly selects several label subsets of a given size k, learns a (LP) multi-label classifier for each subset, and combines their predictions (Tsoumakas and Vlahavas 2007). This may be viewed as a generalization of binary relevance (K classifiers with $k = 1$) and label powerset (1 classifier with $k = K$). Obviously, the larger k, the more dependence-aware this method should be.
- *Predictive clustering trees (PCT)* build up a multi-objective decision tree by using example variance and multi-label prediction quality for guiding the tree construction (Kocev et al. 2007). Full label vectors are predicted at the leafs, hence PCT allows a certain control over the dependence-awareness by setting the leaf and ensemble sizes.
- *BOOMER* is a rule-based multi-label boosting algorithm recently proposed by Rapp et al. (2020), which can be parametrized to optimize either Hamming or subset 0/1 loss, denoted in the experiments as BOOMER-1 and BOOMER-K respectively.

BR and LP are particularly interesting for our analysis, because they are known to follow opposite goals w.r.t. loss minimization and to be diametrical in their treatment of label dependencies, which we expect to be reflected by the proposed dependence-aware losses. RAkEL, CC, and PCT can be positioned in-between LP and BR with respect to their dependence-awareness, although their exact objectives are less obvious. Indeed, RAkEL and PCT allow for a certain control over the dependence-awareness by appropriate selection of parameters.

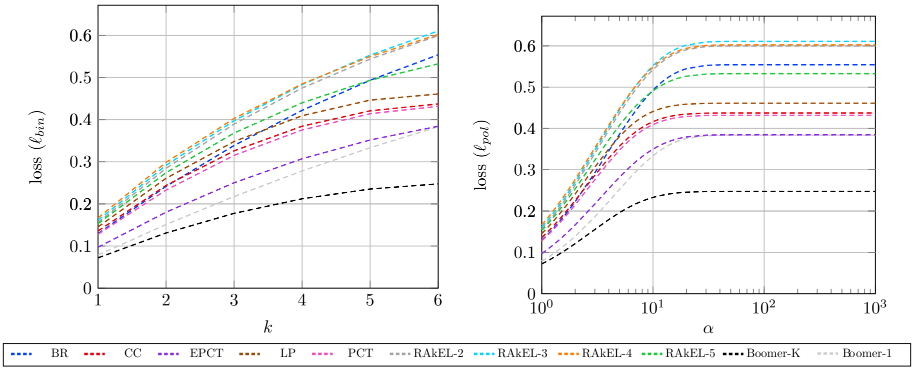

Fig. 1 Comparison of multi-label algorithms on the dataset `scene` w.r.t. parametrized instantiations of the binomial loss (left) and the polynomial loss (right)

For all algorithms we used the implementations of MEKA,[3] except PCTs, for which we used the original implementation CLUS[4] with a self-written to Mulan,[5] and BOOMER, which is available from Github.[6] Due to their favorable runtime, we used decision trees as single-label base learners in all MEKA methods. All hyper-parameters are set to their default values, except for RAkEL, which is evaluated for different values k and the number of ensemble members m, and PCT, which is used with single trees (PCT) and bagged ensembles of 10 trees (EPCT).

5.3 Results

We evaluate the performance of the considered multi-label algorithms in terms of the polynomial instantiation (17) of our loss function, as well as the binomial instantiation (18). We denote the former by ℓ_{poly} and the latter by ℓ_{bin}. While the (discrete) parameter k of ℓ_{bin} takes values in $\{1, \dots, K\}$, we vary the (continuous) parameter of the polynomial loss, α, between 1 and 1000. In both cases, the lowest parameter value 1 corresponds to the Hamming loss and the highest values to the subset 0/1 loss (in the case of ℓ_{poly}, strictly speaking, only for $\alpha \to \infty$), whereas intermediate values interpolate between these two extremes.

We start the analysis with a comparison of the evaluated algorithms for the `scene` dataset (Fig. 1). The graphs plot the value of the parameter k respectively α on the x-axis against the loss of the method on the y-axis. Naturally, the curves are increasing, because with an increasing dependence-awareness, the losses are becoming more demanding and more difficult to minimize.

On closer examination, we can observe some algorithms to work better than other methods for a small k or α, while the order may change as the parameter values increase and the losses demand more dependence-awareness. For example, we can see that BR performs favorably to LP for small k or α, but LP catches up with increasing parameter

[3] http://waikato.github.io/meka/.

[4] https://dtai.cs.kuleuven.be/clus/.

[5] https://mulan.sourceforge.net/.

[6] https://github.com/mrapp-ke/Boomer.

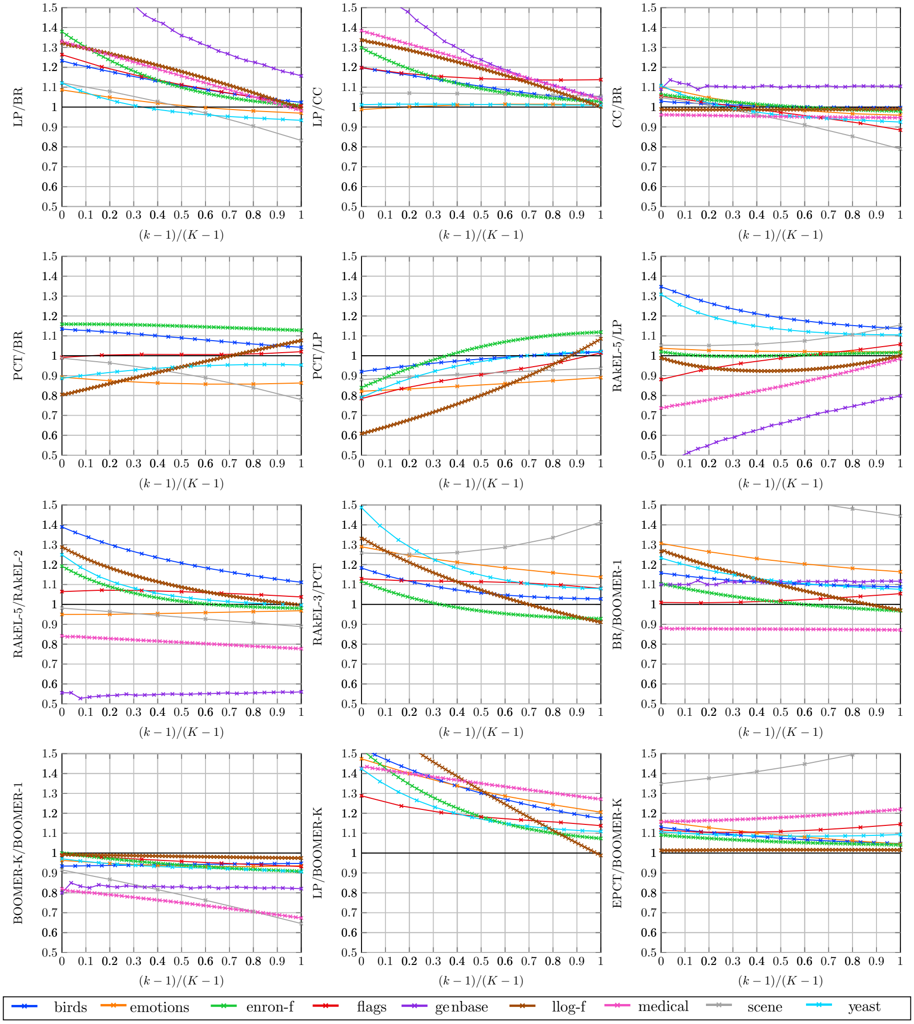

Fig. 2 Pair-wise comparison of selected multi-label algorithms (BR, CC, LP, PCT, EPCT, RAkEL, BOOMER) for the binomial loss with $^{k-1}/_{K-1}$ on the x-axis and the ratio between one method's loss and the loss of another method on the y-axis

values until it finally outperforms BR, which is the expected behavior because LP captures label dependencies while BR does not. In general, the dependence-awareness of a learner is reflected by the slope of the performance curve (the flatter the better): With increasing k or α, the loss is getting more strict, because label dependencies are getting more important (as already stated, this is why the curves are monotone increasing). Moreover, the more a learner is affected by the increased strictness of the loss, the steeper the slope. While the parameter of ℓ_{bin} has a simpler interpretation, as k corresponds to the number of labels that for which a jointly correct prediction is required, α allows for a more fine-grained analysis of dependence-awareness. However, with both families, we can observe intersections between the loss curves of the algorithms, explicitly showing when the order of the methods changes.

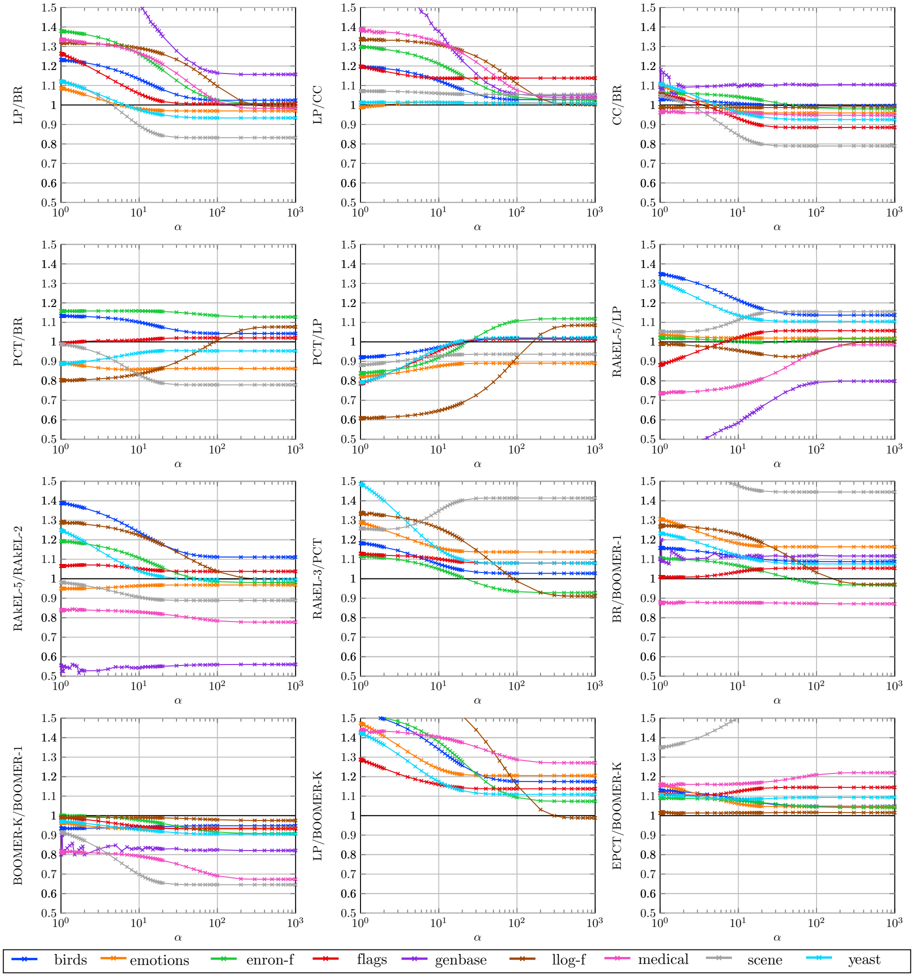

Fig. 3 Pair-wise comparison of selected multi-label algorithms (BR, CC, LP, PCT, EPCT, RAkEL, BOOMER) for the polynomial loss with α on the x-axis and the ratio between one method's loss and the loss of another method on the y-axis

The visualizations chosen in Figs. 2 and 3 allow for a more focused comparison between two methods over several datasets. The graphs shown (one per dataset) are produced by plotting the parameter of the loss (on the x-axis) against the ratio of losses l_2/l_1 of two learners (on the y-axis). Again, we vary the values of the parameters ($1 \leq k \leq K$, since K differs from dataset to dataset rescaled to the range between 0 and 1 for better visualization) for ℓ_{bin} and ($1 \leq \alpha \leq 1000$) for ℓ_{poly}. To interpret these plots, let us highlight the following properties:

- A point on the graph above the horizontal $y = 1$ indicates better performance of the first method l_1. A point below the horizontal indicates that the second method l_2 performs better. Thus, specifically interesting are the crossings of the graph with the horizontal.

- Also interesting is the derivative of the graph: A monotone decreasing (increasing) shape indicates better dependence-awareness of the first (second) method, as it improves relative to the second (first) method with an increasing demand of dependence-awareness. A parallel trajectory, in contrast, indicates a similar behaviour regarding the dependence-awareness.

Obviously, the graphs do not only depend on the methods being compared but also on the dataset. This is natural, because the performance of a method (and hence the performance ratio) differs from dataset to dataset, just like the label dependence. Thus, while dependence-awareness might be an advantage for one dataset, it could be a disadvantage for another one; for example, in the complete absence of dependencies, BR will be very effective, whereas LP will make the learning task unnecessarily difficult. This is why increasing and decreasing graphs can sometimes be observed simultaneously in Figs. 2 and 3. In general, however, there is a relatively clear trend, and the experimental results confirm our expectations: With an increasing dependence-awareness of the loss (increasing k respectively α), simple methods such as BR tend to perform worse than dependence-aware methods like LP, which is also shown by the late crossing of the horizontal by the graphs. The advantage for intermediate and high levels of dependence-awareness is diminished if we compare to CC, a method which is less extreme than LP in its attempt to correctly predict the entire label combination. This can also be observed in the direct comparison between CC and BR, where CC improves over BR with increasing k.

Taking these algorithms as a basis allows us to better understand the behaviour of methods for which the objectives are unknown or unclear. For instance, we can observe that the used parametrization of PCT is clearly more in line with BR regarding the dependence-awareness than with LP, where all trajectories show a clear curvature.

The RAkEL algorithm allows one to control its dependence-awareness with its hyper-parameter k. In fact, this hyper-parameter is very related to the subset size k of the binomial loss. This is also reflected by the ratio between the losses for RAkEL $k = 5$ and $k = 2$, which decreases with increasing subset sizes to match.

Also shown in Fig. 2 are comparisons for BOOMER, more specifically for extreme variants of this learner optimized for $k = 1$ and $k = K$, respectively. A first observation confirms the expected strong performance of BOOMER, especially the dependence-aware variant BOOMER-K (all points are below the horizontal $y = 1$). BOOMER-K is even surprisingly strong for lower k, and this advantage further increases with an increasing dependence-awareness of the loss. This can be seen from the fact that the lines move further away from the horizontal as the dependence-awareness increases. The comparison with LP suggests that, although LP is inferior in absolute terms, its dependence-awareness is even more pronounced, despite the fact that both approaches target the same loss. The reason might be that BOOMER uses a (smooth) surrogate of the subset 0/1 loss (to make it differentiable), which presumably also puts weight on label subsets of lower cardinality.

6 A case study in loss minimization

In the previous section, we used our new loss functions to analyze well-known MLC algorithms with respect to their dependence-awareness. Yet, as already mentioned in the introduction, the losses could in principle also be used for other purposes. In a second experimental study, we focus on their possible role as target and/or surrogate loss.

As a learning algorithm, we use a simple nearest-neighbor (NN) approach, mainly for two reasons. First, in contrast to all other MLC algorithms we are aware of, NN can be adapted to minimizing a generalized loss in a quite straight-forward way.[7] Second, compared to more sophisticated methods, the minimization of the loss is arguably more explicit in NN and less superimposed by other effects.

More specifically, we instantiate NN to minimize the representation of our loss function (14) in terms of the Moebius transform and leave the parameter k of ℓ_{bin} as a hyper-parameter to be configured. To make a prediction on an instance $\boldsymbol{x}$, we first retrieve the label sets $\boldsymbol{y}^{(i)} \in \mathcal{Y}, 1 \leq i \leq c$ of the c nearest neighbors according to the L_1-distance. Then, we enumerate all label sets in $\mathcal{Y} = \{0,1\}^K$ and predict the label set $\hat{\boldsymbol{y}}^*$ minimizing the loss with respect to the label sets $\boldsymbol{y}^{(i)}$ of the nearest neighbors:

$$\hat{\boldsymbol{y}}^* \in \operatorname*{arg\,min}_{\hat{\boldsymbol{y}} \in \mathcal{Y}} \sum_{i=1}^{c} \ell_{\text{bin}}(\boldsymbol{y}^{(i)}, \hat{\boldsymbol{y}}) \tag{19}$$

Since the number of elements in $\mathcal{Y}$ grows exponentially in the number of labels K, complete enumeration quickly becomes computationally infeasible. Fortunately, some pruning of the search space is possible: if all neighbors agree on a label y_k, then this label can (provably) be fixed and removed from search. The probability of this consensus rule to apply is especially high for the prevalent negative labels, but of course decreases with an increasing size of the neighborhood—in our experiments, we set the number of neighbors $c = 9$, for which the consensus rule still yields a considerable reduction.

We evaluated our NN classifier on the datasets birds, emotions, flags, scene, and yeast for all possible parameters k for both the target loss function and the internal (surrogate) loss function used for making predictions. The results of these experiments are summarized in Fig. 4 in terms of heatmaps. In each of the rows, the heatmaps visualize the absolute difference between the loss of any parametrization of the NN classifier and the minimum observed loss. For the purpose of a better and more fine-granular interpretation of these results, the absolute differences are clipped at 0.02, as some differences may become relatively large and distort the overall image. From left to right the NN classifier moves from optimizing Hamming to optimizing for subset 0/1 loss. In turn, a column represents a fixed parametrization of the NN approach and the loss changes top down from Hamming to subset 0/1 loss.

A general observation is that the best hyper-parameter k of the NN approach, i.e., the best parameter k for the surrogate loss locally minimized by NN, monotonically increases with the k of the target loss. That is, the more dependence-aware the target loss, the more dependence-aware the surrogate loss should be, which is quite plausible. This is especially true in the extreme case where the target is subset 0/1. In this case, optimizing for Hamming yields one of the worst two solutions in any of the datasets. That said, the best surrogate loss is not necessarily identical to the target loss. More specifically, the best parameter value k for the surrogate does not increase linearly in the parameter k of the target loss but rather saturates early (although the former also slightly exceeds the latter in a few cases, for example on the emotions, scene, and flags data, which nevertheless remain an exception). This may have several reasons, including those already explained in Sect. 4.4. In particular, note that the loss minimizer does not necessarily change with an increasing k, i.e., the

[7] In principle, BOOMER can also be tailored to any (differentiable) loss function, but the minimization over all possible label subsets poses a non-trivial combinatorial problem.

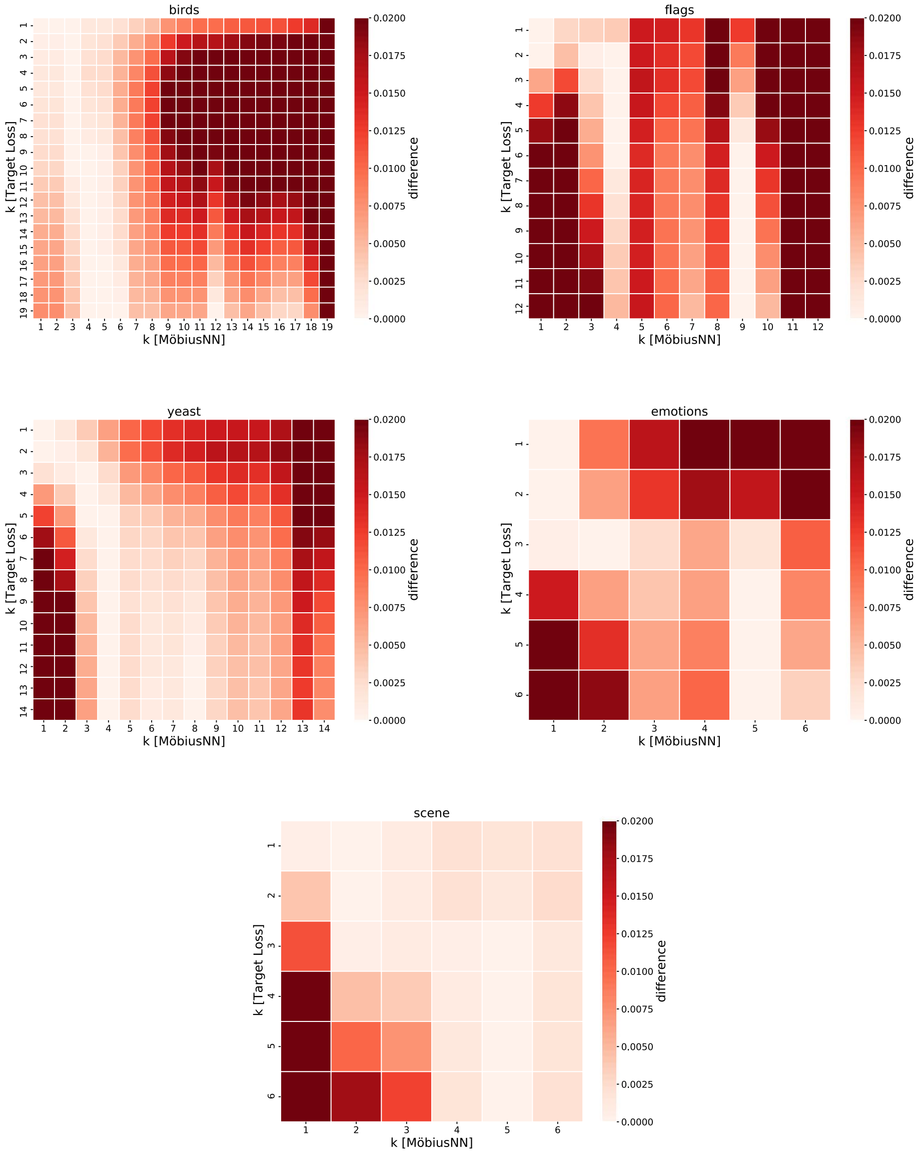

Fig. 4 In each row of a heatmap we consider a target loss function parametrized with a specific k. In each column, in turn we configure our nearest neighbor approach with a different value for its hyper-parameter k, which is used internally to choose a label set for prediction. In each row we plot the absolute different between any configuration and the best one regarding the respective parametrization of the target loss. A darker red thus means a performs closer to the best configuration. For simpler interpretation, the deltas are clipped at a maximum difference of 0.02

minimizer for $k < K$ could (and often will) be the same as for $k = K$. Moreover, learning tends to become more difficult with increasing dependence-awareness. In the case of NN, the training information contained in the neighborhood of a query becomes less redundant when increasing k, and there is a high chance to lose uniqueness. For example, for $k = K$, the risk minimizer is the mode of the empirical distribution on the labelings, which is often not unique because each label combination occurs at most once.

In any case, our results suggest that the generalized MLC losses might indeed be interesting candidates for surrogate losses to be optimized by a learner at training time. For example, in the case of the datasets birds and flags, optimizing for subset 0/1 loss on the training data turns out to be the worst configuration of all, even when subset 0/1 is the target loss. However, the hyper-parameter k (or, more generally, the best level of dependence-awareness) for the learner seems to depend on the data and needs to be tuned to the dataset at hand.

7 Conclusion and future work

We consider a multi-label loss function as "dependence-aware" if it puts emphasis on getting larger label combinations right in their entirety, instead of "merely" making correct predictions on individual labels. In this paper, we introduced a flexible class of loss functions that allows for modeling dependence-awareness by means of non-additive measures. More specifically, we define a loss function in terms of a Choquet integral of label-wise correctness with respect to such a measure. We also proposed two instantiations of our family, in which dependence-awareness can be controlled by a single parameter, thereby "interpolating" between Hamming and subset 0/1 loss.

A first experimental study has shown the potential of our loss functions as a tool for analyzing the dependence-awareness of different MLC methods, i.e., their ability to capture label dependence. Moreover, as suggested by a second study, the losses are also interesting candidates for surrogate losses used for training an MLC classifier, even when the actual target is a different loss. In this regard, a natural next step is to develop new algorithms that are specifically tailored to our family of losses and can be customized for minimizing specific instantiations thereof. The boosting-based multi-label learner BOOMER (Rapp et al. 2020), which we also used in our experiments, is a generic method of that kind. However, as it proceeds from the Moebius representation of the loss, it is not very efficient. It would therefore be interesting to design learners that directly proceed from the integral representation (14). The main problem of this representation, namely that it involves a permutation of the predicted scores, could be overcome by recent work on differentiable approximations of the sorting operator (Blondel et al. 2020).

Apart from algorithmic issues, one may also think of interesting extensions of the loss itself, and further instantiations thereof. For example, the two instantiations we presented are both symmetric, i.e., the importance of a label subset solely depends on its size but not on the labels themselves. From a practical point of view, one may argue that some labels are more important than others, or that subsets including relevant labels are more important than subsets of only irrelevant labels (and recall is more important than precision). This could be taken into account by a suitable design of the non-additive measure, e.g., by specifying the weight of a subset as a function of the number of relevant labels included.

Author Contributions ELM motivated this work, while EH contributed the idea of leveraging non-additive measures and integrals to specify MLC loss functions. He also took the lead in writing. All other authors contributed to the content and the writing as well. The implementation and experiments were mainly conducted by MR and MW.

Funding Open Access funding enabled and organized by Projekt DEAL. This work was partially funded by the German Research Foundation (DFG) under grant numbers 400845550 and 160364472 (the latter being part of the Collaborative Research Center "On-The-Fly Computing", SFB 901/3)

Code availability The code of the experiments presented in this paper is provided at https://github.com/KIuML/DependenceAwareMLCLoss. An implementation of Boomer is available at https://github.com/mrapp-ke/Boomer. For basic MLC algorithms, we used the implementation in MEKA (http://waikato.github.io/meka/), except PCTs, for which we used the original implementation CLUS (https://dtai.cs.kuleuven.be/clus/) with a self-written to Mulan (https://mulan.sourceforge.net/), and BOOMER, which is available from Github https://github.com/mrapp-ke/Boomer.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

Amit, Y., Dekel, O., & Singer, Y. (2007). A boosting algorithm for label covering in multilabel problems. In *Proc. int. conf. artificial intelligence and statistics (AISTATS), PMLR* (pp. 27–34).

Apté, C., Damerau, F., & Weiss, S. M. (1994). Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems, 12*(3), 233–251.

Blondel, M., Teboul, O., Berthet, Q., & Djolonga, J. (2020). Fast differentiable sorting and ranking. In *Proc. international conference on machine learning (ICML)* (pp. 950–959).

Boutell, M. R., Luo, J., Shen, X., & Brown, C. M. (2004). Learning multi-label scene classification. *Pattern Recognition, 37*(9), 1757–1771.

Choquet, G. (1954). Theory of capacities. *Annales de l'nstitut Fourier, 5*, 131–295.

Dembczynski, K., Waegeman, W., Cheng, W., & Hüllermeier, E. (2010). Regret analysis for performance metrics in multi-label classification: The case of Hamming and subset zero-one loss. In *Proc. European conf. on machine learning (ECML/PKDD), Barcelona, Spain* (pp. 280–295).

Dembczynski, K., Waegeman, W., Cheng, W., & Hüllermeier, E. (2012). On label dependence and loss minimization in multi-label classification. *Machine Learning, 88*(1–2), 5–45.

Diplaris, S., Tsoumakas, G., Mitkas, P. A., & Vlahavas, I. P. (2005). Protein classification with multiple algorithms. In *Proc. Panhellenic conference on informatics* (pp. 448–456). Springer.

Hayes, P. J., & Weinstein, S. P. (1990). CONSTRUE/TIS: A system for content-based indexing of a database of news stories. In *Proc. 2nd conference on innovative applications of artificial intelligence (IAAI), AAAI* (pp. 49–64).

Klement, E., Mesiar, R., & Pap, E. (2002). *Triangular norms*. Kluwer Academic Publishers.

Kocev, D., Vens, C., Struyf, J., & Dzeroski, S. (2007). Ensembles of multi-objective decision trees. In *Proc. 18th European conference on machine learning (ECML/PKDD)* (pp. 624–631). Springer.

Lewis, D. D. (1992) An evaluation of phrasal and clustered representations on a text categorization task. In *Proc. 15th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR)* (pp. 37–50). ACM.

Rapp, M., Loza Mencía, E., Fürnkranz, J., Nguyen, V., & Hüllermeier, E. (2020). Learning gradient boosted multi-label classification rules. In *Proc. European conference on machine learning and knowledge discovery in databases (ECML/PKDD), Ghent, Belgium* (pp. 124–140).

Read, J., Pfahringer, B., Holmes, G., & Frank, E. (2009). Classifier chains for multi-label classification. In *Proc. European conference on machine learning and knowledge discovery (ECML/PKDD), Part II* (pp. 254–269). Springer.

Read, J., Pfahringer, B., Holmes, G., & Frank, E. (2021). Classifier chains: A review and perspectives. *Journal of Artificial Intelligence Research, 70*, 683–718.

Rivolli, A., Read, J., Soares, C., Pfahringer, B., & de Carvalho, A. C. (2020). An empirical analysis of binary transformation strategies and base algorithms for multi-label learning. *Machine Learning, 109*(8), 1509–1563.

Sugeno, M. (1974). *Theory of fuzzy integrals and its application*. Ph.D. thesis, Tokyo Institute of Technology.

Tehrani, A. F., & Ahrens, D. (2017). Modeling label dependence for multi-label classification using the Choquistic regression. *Pattern Recognition Letters, 92*, 75–80.

Tehrani, A. F., Cheng, W., Dembczynski, K., & Hüllermeier, E. (2012). Learning monotone nonlinear models using the Choquet integral. *Machine Learning, 89*(1), 183–211.

Tehrani, A. F., Cheng, W., & Hüllermeier, E. (2012). Preference learning using the Choquet integral: The case of multipartite ranking. *IEEE Transactions on Fuzzy Systems, 20*(6), 1102–1113.

Trohidis, K., Tsoumakas, G., Kalliris, G., & Vlahavas, I. P. (2011). Multi-label classification of music by emotion. *EURASIP Journal on Audio, Speech and Music Processing, 2011*, 4.

Tsoumakas, G., & Vlahavas, I. P. (2007). Random *k*-labelsets: An ensemble method for multilabel classification. In *Proc. 18th European conference on machine learning (ECML/PKDD)* (pp. 406–417). Springer.

Tsoumakas, G., Katakis, I., & Vlahavas, I. P. (2010). Mining multi-label data. In *Data mining and knowledge discovery handbook* (2nd edn., pp. 667–685). Springer.

Wever, M., Mohr, F., & Hüllermeier, E. (2018). Automated multi-label classification based on ml-plan. ArXiv preprint arXiv:1811.04060

Wever, M., Tornede, A., Mohr, F., & Hüllermeier, E. (2020). Libre: Label-wise selection of base learners in binary relevance for multi-label classification. In *Advances in intelligent data analysis XVIII (IDA)* (pp. 561–573). Springer.

Wu, J., Xiong, W., & Wang, W. Y. (2019). Learning to learn and predict: A meta-learning approach for multi-label classification. CoRR. http://arxiv.org/abs/1909.04176

Wu, X., & Zhou, Z. (2017). A unified view of multi-label performance measures. In *Proc. ICML, international conference on machine learning*.

Yager, R., & Filev, D. (1999). Induced ordered weighted averaging operators. *IEEE Transactions on Systems, Man, and Cybernetics, Part B, 29*(2), 141–150.

Yager, R., & Kacprzyk, J. (Eds.). (2012). *The ordered weighted averaging operators: Theory and applications*. Springer.

Yessou, H., Sumbul, G., & Demir, B. (2020). A comparative study of deep learning loss functions for multi-label remote sensing image classification. In *IEEE international geoscience and remote sensing symposium*.

Zhang, M., & Zhou, Z. (2014). A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering, 26*(8), 1819–1837.

Zhang, M., Li, Y., Liu, X., & Geng, X. (2018). Binary relevance for multi-label learning: An overview. *Frontiers of Computer Science, 12*(2), 191–202.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Authors and Affiliations

Eyke Hüllermeier[1] · Marcel Wever[2] · Eneldo Loza Mencia[3] · Johannes Fürnkranz[4] · Michael Rapp[3]

Marcel Wever
marcel.wever@upb.de

Eneldo Loza Mencia
research@eneldo.net

Johannes Fürnkranz
juffi@faw.jku.at

Michael Rapp
michael.rapp.ml@gmail.com

[1] University of Munich (LMU), Munich, Germany

[2] Heinz Nixdorf Institut, Paderborn University, Paderborn, Germany

[3] TU Darmstadt, Darmstadt, Germany

[4] Johannes Kepler University, Linz, Austria