A Blended Metric for Multi-label Optimisation and Evaluation

Laurence A. F. Park[1] and Jesse Read[2]

[1] School of Computing, Engineering and Mathematics,
Western Sydney University, Australia.
lapark@scem.westernsydney.edu.au
[2] DaSciM team, LIX Laboratory,
École Polytechnique, 91120 Palaiseau, France.
firstname.lastname@polytechnique.edu

Abstract. In multi-label classification, a large number of evaluation metrics exist, for example Hamming loss, exact match, and Jaccard similarity – but there are many more. In fact, there remains an apparent uncertainty in the multi-label literature about which metrics should be considered and when and how to optimise them. This has given rise to a proliferation of metrics, with some papers carrying out empirical evaluations under 10 or more different metrics in order to analyse method performance.
We argue that further understanding of underlying mechanisms is necessary. In this paper we tackle the challenge of having a clearer view of evaluation strategies. We present a blended loss function. This function allows us to evaluate under the properties of several major loss functions with a single parameterisation. Furthermore we demonstrate the successful use of this metric as a surrogate loss for other metrics. We offer experimental investigation and theoretical backing to demonstrate that optimising this surrogate loss offers best results for several different metrics than optimising the metrics directly. It simplifies and provides insight to the task of evaluating multi-label prediction methodologies.

1 Introduction

The major challenge in multi-label classification is dealing with multiple output labels simultaneously, which has important ramifications on building models, and also evaluating them. There have been an impressive number of new methods proposed in recent years, proposing different ways of modelling labels together, but relatively little investigation into the study of *which* loss functions methods optimise, which functions they *should* optimise, and how well they can be expected to achieve this for a given problem. As a result, empirical studies look at up to a dozen evaluation metrics. Our study is targeted at bringing new clarity and insight to this situation.

In multi-label classification, optimisation as part of a predictive model inherently involves multiple dimensions; one for each label. A review of multi-label

classification is given in [11]. A number of common benchmark algorithms are reviewed in [15] and [5]. Often, a vector is used to represent the labelling, e.g., $\mathbf{y}_k = [y_{k1}, y_{k2}, \ldots, y_{kL}]$ where $y_{kj} = 1$ iff the j-th of L labels is relevant to the k-th example $\mathbf{x}_k$ (and $y_{kj} = 0$ otherwise). To evaluate the performance of a multi-label classifier, typically predicted vectors $\mathbf{y}_k$ must be compared the vector of true labels $\mathbf{t}_k$ over all examples $k = 1, \ldots, K$ (for a test set of K examples).

Three of the most common similarity functions used in multi-label learning and evaluation are the Jaccard index, Hamming loss, and 0/1 loss. Jaccard index is known as accuracy in some publications, e.g., [3, 9], Hamming loss and 0/1 loss are known often as Hamming score and exact match in their payoff-form (higher is better), respectively [7]. However the basic principal of all multi-label metrics is the same for any metric: provide a single number indicating the *similarity* of the set (or vector[3]) of *predicted* labels compared to the set of *true* labels, i.e., a score that may be normalised to between 0 and 1. We henceforth refer to each of these mentioned metrics as a *similarity* or *loss* function, interchangeably. There is a large number of multi-label criteria, including rankings and micro and macro evaluation; a recent survey of multi-label metrics and unified view of them is given by [14].

Consider the examples in Table 1. The similarity functions are defined, for the k-th instance, as

$$\text{Hamming} := \frac{1}{L} \sum_{j=1}^{L} \mathbb{I}[y_{kj} = t_{kj}] \tag{1}$$

$$\text{Exact} := \mathbb{I}[\mathbf{y}_k = \mathbf{t}_k] \tag{2}$$

$$\text{Jacard} := \frac{|\mathbf{y}_k \wedge \mathbf{t}_k|}{|\mathbf{y}_k \vee \mathbf{t}_k|} \tag{3}$$

for L possible labels, where $\vee$ and $\wedge$ are the logical OR and AND operations, applied vector-wise, and $\mathbb{I}[\cdot]$ is an indicator function returning 1 if the inner condition holds (0 otherwise).

Each of these similarity functions measure different accuracy qualities of a multi-label classification system. Hamming similarity provides the proportion of labels predicted correctly, Exact similarity provides the proportion of label sets predicted correctly, while Jaccard similarity only examines the proportion of correctly predicted positive labels out of the potential positive set (predicted positive and actually positive). Therefore, a multi-label classification system should be optimised according to the desired similarity function.

The problem of optimisation in multi-label classification is complicated by the label dimension and the interdependence of labels. Resulting search spaces are usually non-convex and non-differentiable, and all the difficulties of such a search are inherited. A solution may fall into a local maximum/minimum and provide a non-optimal solution. We can transform the problem into simpler problems which

[3] Notation alternates among papers, since given a set of labels $\mathcal{L}$, a (sub)set $Y \subseteq \mathcal{L}$ can be represented as vector $\mathbf{y} = [y_1, \ldots, y_L]$ where $y_j = 1 \Leftrightarrow y_j \in Y$

Table 1. An example of multi-label evaluation. The average Hamming, Exact, and Jaccard similarity is 0.80, 0.40, and 0.67, respectively. Note that real-world multi-label data sets are typically much sparser with respect to labelling.

	$\mathbf{t}_k$	$\mathbf{y}_k$	Hamming	Exact	Jaccard
$\mathbf{x}_1$	[1 0 1 0]	[1 0 0 1]	0.5	0	0.33
$\mathbf{x}_2$	[0 1 0 1]	[0 1 0 1]	1	1	1
$\mathbf{x}_3$	[1 0 0 1]	[1 0 0 1]	1	1	1
$\mathbf{x}_4$	[0 1 1 0]	[0 1 0 0]	0.75	0	0.5
$\mathbf{x}_5$	[1 0 0 0]	[1 0 0 1]	0.75	0	0.5

provide convex search spaces, but even in these case, optimisation is typically much more difficult than a traditional single-label problem.

One solution is to optimise each label individually and independently. This approach is known as Hamming similarity, and defines the so-called *binary relevance* method, widely known across the multi-label literature as a baseline approach. Indeed we see that Hamming similarity is proportional to the sum of individual label similarities, therefore, when optimising for Hamming similarity, we can simply optimise the accuracy of each label. An example of this is to model each label with a logistic regression; resulting in L tasks of convex-optimisation.

However, it is typical to motivate methods that model labels together (e.g., [3, 8, 9, 15, 12] and many references therein). As shown thoroughly in those papers and many others, it is desirable to model an explicit or implicit dependence among labels. And this, in turn, motivates the *evaluation* of labels together also. Exact similarity does not decompose across labels (i.e., it requires optimisation of all labels jointly) and therefore encourages modelling labels together. In fact, for this metric it is theoretically optimal to model label combinations as single class values in a large multi-*class* problem [2]. This transformation is often called the *label powerset* method. Hence each vector/set $\mathbf{y}_k$ is treated as a *single* value that may be assigned to each instance. This can also be formulated as a differentiable and convex optimization problem, such as multi-class logistic regression, since the multiple classes can be modelled under a single softmax function. Nevertheless, optimisation is still not expected to be easy or necessarily effective in practice (see, e.g., [10, 12]): most label combination (class) are likely to be very sparse or not present in the training data and the metric is particularly sensitive to noise in the data, since even 1 of the L labels being incorrectly predicted will lead to a 0 score for that particular instance. Other methods such as classifier chains [9, 1] provide an alternative, which divides the optimisation across labels. However, unlike the binary relevance method, the labels are linked together in a chain cascade and thus can no longer be optimized separately (at least, not in terms of optimising exact similarity), and thus becomes thus a non-convex problem and remains a difficult task, hence paving the way for several extensions (e.g., [8, 1, 2]) with approximate inference, and alternatives (e.g., [12, 10]) with exact inference on small sub-problems.

Jaccard similarity is also used for evaluating labels together. It is often preferred since it can be seen as a midpoint between Hamming and exact similarity as two extremes. However, fewer theoretical results exist for its optimisation, although relationships with F-measure has been outlined [13]. It remains popular, and it is typically assumed that good results in both Hamming and exact similarity will reflect good results also in Jaccard similarity. Numerous empirical studies also show this [8, 5].

Therefore, to summarise so far: Hamming similarity is easy to optimise, but may not correspond to labellings that we would find desirable in real-world problems. On the other hand, optimising a multi-label classification system with respect to Jaccard or Exact similarity is difficult, since it requires optimisation of all labels at once, leading to a large optimisation search space. Solutions involve:

- Problem transformation (i.e., binary relevance, or label powerset)
- Non-convex optimisation such as local search and hill-climbing methods
- Using a surrogate loss
- Smoothing and regularisation of the loss to improve results.

In this paper we propose a blended metric, combining Jaccard and Exact similarity functions with the Hamming function as a surrogate metric for both Jaccard and Exact similarity, providing a simpler search space and thus more efficient and effective optimisation. Under a series of investigations we show the benefit of this blended metric. In particular, we obtain a very important result: best results for exact match can be obtained using our surrogate blended metric, rather than the exact match minimizer (label powerset) itself. In general, we are able to conclude making the recommendation of exploring different surrogate metrics, rather than optimising separately across an ever-growing set of metrics.

Our novel contribution is covered as follows:

- In Section 2 we formulate our surrogate blended metric: a single function that is a blend of the three similarity functions (Hamming, Exact, and Jaccard).
- In Section 3 we derive the gradient for optimisation of this function.
- In Section 4 and Section 5 we examine how optimisation over the blended function affects the accuracy with regard to the maximisation of the three similarity functions, on a number of real-world multi-label datasets; and we identify the spectrum where each of the three similarity functions are maximised and we discuss in detail and draw conclusions and recommendations.

2 Multi-label Evaluation and Undetermined Predictions

To begin our investigation, we formulate the blended Hamming, Jaccard, Exact similarity function and show that it behaves well when using undetermined label values (such as votes or probabilistic predictions $\in [0, 1]$ for each label).

2.1 A Blended Metric

Hamming and Jaccard similarity can be represented in terms of true/false positive/negative counts[4]:

$$\text{Hamming} = \frac{\text{TP} + \text{TN}}{L} \qquad \text{Jaccard} = \begin{cases} \frac{\text{TP}}{\text{TP+FP+FN}} & \text{if TN} \neq L \\ 1 & \text{otherwise} \end{cases}$$

where $\text{TP} + \text{TN} + \text{FP} + \text{FN} = L$ (the total number of labels). We also note that exact similarity can also be represented in terms of Hamming and Jaccard similarity.

$$\text{Exact} = \lim_{\beta \to \infty} \text{Hamming}^{\beta} = \lim_{\beta \to \infty} \text{Jaccard}^{\beta}$$

Of the three similarity functions, Hamming similarity is the simplest to optimise over. Hamming similarity is the mean of the accuracy of each label prediction and therefore, each label can be optimised individually (details in [2]). Both Jaccard and Exact similarity depend on the collection of labels, so the optimisation is non-decomposable.

We reason that a blending of Hamming similarity with either Jaccard or Exact similarity provides a greater optimisation of Jaccard or Exact similarity. The insight to this is that the Hamming optimisation search space is likely to be less chaotic than the Jaccard and Exact optimisation search spaces, providing a simpler optimisation path. On the other hand, this may lead to non-optimal solutions. Therefore we will investigate this effect throughout the rest of this paper.

We propose the blended Hamming, Jaccard similarity function:

$$s(\mathbf{y}, \mathbf{t}; \alpha) = \frac{\text{TP} + \alpha\text{TN}}{\text{TP} + \text{FN} + \text{FP} + \alpha\text{TN}} \tag{4}$$

$$= \frac{\text{TP} + \alpha\text{TN}}{L - (1 - \alpha)\text{TN}} \tag{5}$$

given the vector of true labels $\mathbf{t}$, the vector of determined or undetermined predictions $\mathbf{y}$, and a parameter α which controls the blending. This metric provides Hamming similarity when $\alpha = 1$ and Jaccard when $\alpha = 0$. We additionally note that this form avoids the divide-by-zero problem of the Jaccard similarity. By taking the limit as $\alpha \to 0$, the equation results in $s(\mathbf{y}, \mathbf{t}; \alpha \to 0) = \text{TN}/\text{TN} = 1$, when TP, FN and FP are zero.

2.2 Behaviour using undetermined labels

The blended similarity function is a function of true/false positive/negative counts. Optimising over these functions is difficult due to the discrete nature

[4] Throughout this work we refer to F-measure and Jaccard in an instance-wise evaluation context (noting that these metrics can also be used in a micro- or macro-averaging context)

of these counts, which leads to a discontinuous optimisation function and a gradient containing values of either zero of infinity.

Of course, many multi-label classifiers compute undetermined label values z_{kj} in the $[0, 1]$ range, that are used to obtain the label predictions $y_{kj} \in \{0, 1\}$, such that

$$y_{kj} = \mathbb{I}\left[z_{kj} \geq \tau_j\right] \tag{6}$$

is the class prediction for the j-th label ($\mathbb{I}[\cdot] = 1$ if the inner condition holds), where τ_j is some threshold, typically set to 0.5 but may also be tuned (see, e.g., [4, 9]) either per label or the same for all labels $j = 1, \ldots, L$. The vector $\mathbf{z}_k = [z_{k1}, \ldots, z_{kL}]$ contains the undetermined label values and in many cases $z_{kj} \approx P(y_{kj} = 1|\mathbf{x}_k)$ is related to probabilistic models or approximations[5], or is a normalised sum of ensemble votes (e.g., [12, 10, 1]).

Using these undetermined label values z_{kj} to obtain undetermined true/false positive/negative values, provides a continuous optimisation space, and a gradient for gradient-based optimisation. For example, given the three labels $\mathbf{y}_k =$ [1 0 1] and the predicted undetermined label values $\mathbf{z}_k =$ [0.6 0.4 0.2], the undetermined values would be rounded to obtain determined the values $\mathbf{y}_k =$ [1 0 0], giving TP = 1, FP = 0, FN = 1, TN = 1. Using the undetermined values instead, we get TP = 0.8, FP = 0.4, FN = 1.2, TN = 0.6. For both cases TP + TN + FP + FN = 3.

The undetermined values can also be used for evaluation, such as in the case of the log loss metric (applied to multi-label evaluation in, e.g., [9]). Nevertheless, such metrics are relatively less popular in the multi-label literature, perhaps because not all classifiers can provide them. Nevertheless, they help considerably in the optimisation to smooth out the search space.

2.3 Blending the Exact Similarity

TP and TN are dependent only on the correctness of the predictions (the correctly predicted 1 and 0 values), so we instead use the vector of correctness values $\mathbf{p}_k = [p_{k1}, \ldots, p_{kL}]$, where p_{ki} (each element of $\mathbf{p}_k$) is equal to

$$\begin{cases} z_{ki} & \text{when } t_{ki} = 1 \\ 1 - z_{ki} & \text{when } t_{ki} = 0 \end{cases} \tag{7}$$

where t_{ki} is the true label. The combined Hamming-Jaccard score using undetermined labels can now be represented as

$$s(\mathbf{z}_k, \mathbf{t}_k; \alpha) = \frac{\mathbf{p}_k^\top \mathbf{a}_k}{L + \mathbf{p}_k^\top \mathbf{b}_k}$$

where $\mathbf{a}_k$ contains 1 when true label $t_{ki} = 1$ and α when true label is 0; and $\mathbf{b}_k$ contains 0 when the true label $t_{ki} = 1$ and $\alpha - 1$ when the true label is 0. The set of vectors $\mathbf{a}_k$ and $\mathbf{b}_k$ are constant for the optimisation.

[5] Note, however, that metrics like F-measure and Jaccard measure cannot be optimised only under consideration of marginal probabilities.

To include the Exact similarity into this spectrum, we add the parameter β

$$s(\mathbf{z}_k, \mathbf{t}_k; \alpha, \beta) = \left[\frac{\mathbf{p}_k^\top \mathbf{a}_k}{L + \mathbf{p}_k^\top \mathbf{b}_k} \right]^\beta$$

which is equivalent to Hamming similarity when $\alpha = 1, \beta = 1$, and equivalent to Jaccard similarity when $\alpha \to 0$, $\beta = 1$ and to Exact when $\beta \to \infty$. A continuous spectrum exists between the three similarity functions for $\alpha \in (0, 1]$ and $\beta \in [1, \infty)$.

Given that we obtain the Exact similarity as $\beta \to \infty$, let us look at how α effects the rate at which the limit is approached, and if so which value of α provides the fastest rate of convergence. Namely, if α is adjusted to $\alpha + \delta$, where $\delta > 0$, the score changes to:

$$s(\mathbf{z}_k, \mathbf{t}_k; \alpha + \delta, \beta) = \left[\frac{\mathbf{p}_k^\top \mathbf{a}_k + \delta(1 - \mathbf{t}_k)}{L + \mathbf{p}_k^\top \mathbf{b}_k + \delta(1 - \mathbf{t}_k)} \right]^\beta \tag{8}$$

This value is bound between 0 and 1, so this addition of $\delta(1-\mathbf{t}_k)$ to the numerator and denominator will increase the score. And, thus,

$$s(\mathbf{z}_k, \mathbf{t}_k; \alpha, \beta) \leq s(\mathbf{y}_k, \mathbf{t}_k; \alpha + \delta, \beta)$$

for $0 \leq \alpha \leq [\alpha+\delta] \leq 1$. The limit as $\beta \to \infty$ will approach the correct Exact score faster when any score that is not 1, is closer to zero (since Exact requires that all fractional scores should be mapped to zero). Since decreasing α decreases the score, the limit will approach the correct score faster for smaller α. Therefore, we expect that the best estimate of Exact will occur when $\alpha \to 0$ and $\beta \to \infty$.

3 Optimising the combined metric

In this section, we present the linear model, the optimisation function used to fit the linear model, a derivation of the optimisation gradient and inclusion of a penalty to deter over-fitting.

3.1 Optimisation function

Suppose the linear model

$$\log\left(\frac{\mathbf{z}_k}{1 - \mathbf{z}_k}\right) = W\mathbf{x}_k$$

with unknown matrix W, undetermined prediction vector $\mathbf{z}_k \in (0, 1)^L$ and feature vector $\mathbf{x}_k \in \mathbb{R}^M$. This model resembles a set of L parallel logistic regressions, but unlike logistic regression, we are not determining the W that maximises the likelihood of the data. We compute W that maximises the chosen evaluation

function score $s(\mathbf{z}_k, \mathbf{t}_k; \alpha, \beta)$ for all k. To optimise the average score, the optimisation problem becomes:

$$\max_W \frac{1}{K} \sum_{k=1}^{K} s(\mathbf{z}_k, \mathbf{t}_k; \alpha, \beta) \tag{9}$$

where W is the weight matrix containing the elements w_{ij}, $s_k \in [0, 1)$ is the score for each of the K objects.

$$s(\mathbf{z}_k, \mathbf{t}_k; \alpha, \beta) = s_k = \left[\frac{\sum_{i=1}^{L} p_{ki} a_{ki}}{L + \sum_{i=1}^{L} p_{ki} b_{ki}} \right]^{\beta}$$

L is the number of labels, $p_{ki} \in (0, 1)$ is the correctness of the undetermined label prediction z_{ki} (recall Eq. (7))

$$p_{ki} = z_{ki}^{t_{ki}} (1 - z_{ki})^{(1 - t_{ki})}$$

and $z_{ki} \in (0, 1)$ is the sigmoid of the mapped feature vector $\mathbf{x}_k$

$$z_{ki} = \frac{1}{1 + \exp\left(-\sum_j w_{ij} x_{kj}\right)}$$

where x_{kj} is an element of the feature vector $\mathbf{x}_k$, and the element 1 is appended to $\mathbf{x}_k$ as a bias term. The constants $a_{ki} \in \{\alpha, 1\}$ and $b_{ki} \in \{0, 1 - \alpha\}$ depend only on the optimisation parameter α and the true label values t_{ki}.

$$a_{ki} = t_{ki} + \alpha(1 - t_{ki}) \qquad b_{ki} = (\alpha - 1)(1 - t_{ki})$$

$\alpha \in (0, 1]$ and $\beta \in [1, \infty)$ are parameters to set the desired optimisation (e.g. $\alpha = 1$, $\beta = 1$ for Hamming similarity). Also note that $a_{ki} - b_{ki} = 1$.

The optimisation of Eq. (9) provides us with W which we can use to make determined predictions by

$$\mathbf{y} = \operatorname{sign}(W \mathbf{x}_k)$$

where sign returns the L signs of the L elements in $W\mathbf{x}_k$, making sure that 1 is appended to $\mathbf{x}$ if it was used to compute a bias term during optimisation. Note that the bias term has the same role as the threshold described in Section 2.2.

3.2 Metric gradient

We can optimise the similarity function using gradient descent or stochastic gradient descent for large problems. To do so, we need the gradient of the function with respect to each elements of the matrix W. The derivative of the sigmoid function is

$$\frac{dz_{ki}}{dw_{ij}} = x_{kj} z_{ki} (1 - z_{ki})$$

The derivative of the correctness function is

$$\frac{dp_{ki}}{dz_{ki}} = p_{ki}\left[\frac{t_{ki}}{z_{ki}} - \frac{1-t_{ki}}{1-z_{ki}}\right]$$

and the derivative of the score function with respect to the correctness is

$$\frac{ds_k}{dp_{ki}} = \beta s_k \left[\frac{a_{ki}}{\sum_{i=1}^{L} p_{ki}a_{ki}} - \frac{b_{ki}}{L + \sum_{i=1}^{L} p_{ki}b_{ki}}\right]$$

Combining the partial derivatives provides the derivative of the score function with respect to the unknown weights;

$$\frac{ds_k}{dw_{ij}} = \frac{ds_k}{dp_{ki}}\frac{dp_{ki}}{dz_{ki}}\frac{dz_{ki}}{dw_{ij}} = \beta s_k p_{ki}(t_{ki} - z_{ki})\left[\frac{a_{ki}}{\sum_{i}^{L} p_{ki}a_{ki}} - \frac{b_{ki}}{L + \sum_{i}^{L} p_{ki}b_{ki}}\right] x_{kj} \tag{10}$$

Using gradient descent, we update weights W until the optimisation function stops increasing,

$$w_{ij} \leftarrow w_{ij} + \lambda \frac{ds_k}{dw_{ij}}$$

where λ controls the rate of convergence.

Note that if $\alpha = 1$ and $\beta = 1$ (giving Hamming similarity), we obtain $a_{ki} = 1$, $b_{ki} = 0$ and $s_k = \sum_i^L p_{ki}/L$ giving the gradient

$$\begin{aligned}\frac{ds_k}{dw_{ij}} &= \frac{p_{ki}(t_{ki} - z_{ki})x_{kj}}{L} \\ &= \begin{cases} \frac{z_{ki}(1-z_{ki})x_{kj}}{L} & \text{if } t_{ki} = 1 \\ \frac{-z_{ki}(1-z_{ki})x_{kj}}{L} & \text{if } t_{ki} = 0 \end{cases} = \frac{(-1)^{1-t_{ki}}}{L}\frac{dz_{ki}}{dw_{ij}}\end{aligned}$$

Thus we see that the gradient that optimises Hamming similarity with respect to w_{ij} is independent of the weights for other values of i, meaning that each label can be fitted independently.

3.3 Optimisation penalty

To reduce the chance of over-fitting the training data, we include a penalisation term. Thus Eq. (9) becomes

$$\max_W \frac{1}{K}\sum_{k=1}^{K} s(\mathbf{z}_k, \mathbf{t}_k; \alpha, \beta) - \underbrace{\frac{\gamma}{2}\sum_{i=1}\sum_{j=1} w_{ij}^2}_{\text{penalty}}$$

for some positive γ, typically chosen via cross-validation. We make a corresponding minor adjustment to the gradient, hence Eq. (10) becomes

$$\frac{ds_k}{dw_{ij}} = \beta s_k p_{ki}(t_{ki} - z_{ki})\left[\frac{a_{ki}}{\sum_{i}^{L} p_{ki}a_{ki}} - \frac{b_{ki}}{L + \sum_{i}^{L} p_{ki}b_{ki}}\right] x_{kj} - \gamma w_{ij}$$

4 Parameter investigation

Having proposed a parametrisable blended metric, and derived its optimiser, we now turn to explore the question: Does blending the Hamming similarity with either Jaccard or Exact similarity provide a greater optimisation of Jaccard or Exact similarity itself?

We answer this question by optimising the model over different data sets, using different α, β parameter combinations and examining the results using the three similarity functions.

Namely, we use the Emotions, Enron, Scene, Slashdot, Stare and Yeast multi-label data sets[6]. Each of these data sets are commonly used for multi-label machine learning except for Stare, which is a data set for detecting cardiovascular disease from retinal features [6]. A 50/50 train/test split is used and the models are optimised over the training portion. We experiment with values of α from 0.1 to 1 in increments of 0.1 and also $\alpha = 0.00001$ to approximate the limit of α approaching zero ($\alpha = 0$ was not used to avoid the problems associated to the Jaccard metric). We also experiment with eight values $\beta \in \{2^0, \ldots, 2^7\}$ (higher values lead to approximately zero gradients), giving 88 $\alpha \times \beta$ combinations. The penalty parameter γ was computed using cross validation. Each of the 88 optimised models for each data set was evaluated using Hamming, Jaccard and Exact similarity to examine the effect of the parameters on the evaluation scores. The optimisation was performed using undetermined scores, but the evaluation scores (shown in the figures) are computed using the final determined labels. The results for the training data are shown in Figure 1, and the results for the testing data are shown in Figure 2. The top six plots in each figure show the Hamming similarity on each of the six data sets, the middle six show the Jaccard similarity, and the bottom six show the Exact accuracy. The shade of each block in a given plot shows the accuracy of the model optimised using the set α, β parameters. For example, the first plot in Figure 1 shows the Hamming similarity on the training portion of the Emotions data; the bottom row of the plot shows the grey level transitioning from light to dark grey, meaning that when $\beta = 1$ ($\log_2 \beta = 0$) the Hamming accuracy increases as α changes from 0 to 1.

5 Results and Discussion

The top sections of Figures 1 and 2 show the mean Hamming similarity between the predicted and true label sets for the six data sets on the training and testing portions. Each block in the figures show the effect of changing α and β on the Hamming similarity. We would expect that the optimal configuration for Hamming similarity is $\alpha = 1$, $\beta = 1$ (the lower right corner of each block plot) since it optimises Hamming similarity. We can see that this is the case for most of

[6] All available from `http://mulan.sourceforge.net/datasets-mlc.html`, `https://sourceforge.net/projects/meka/files/Datasets/` (Slashdot), and `http://www.ces.clemson.edu/~ahoover/stare/` (Stare).

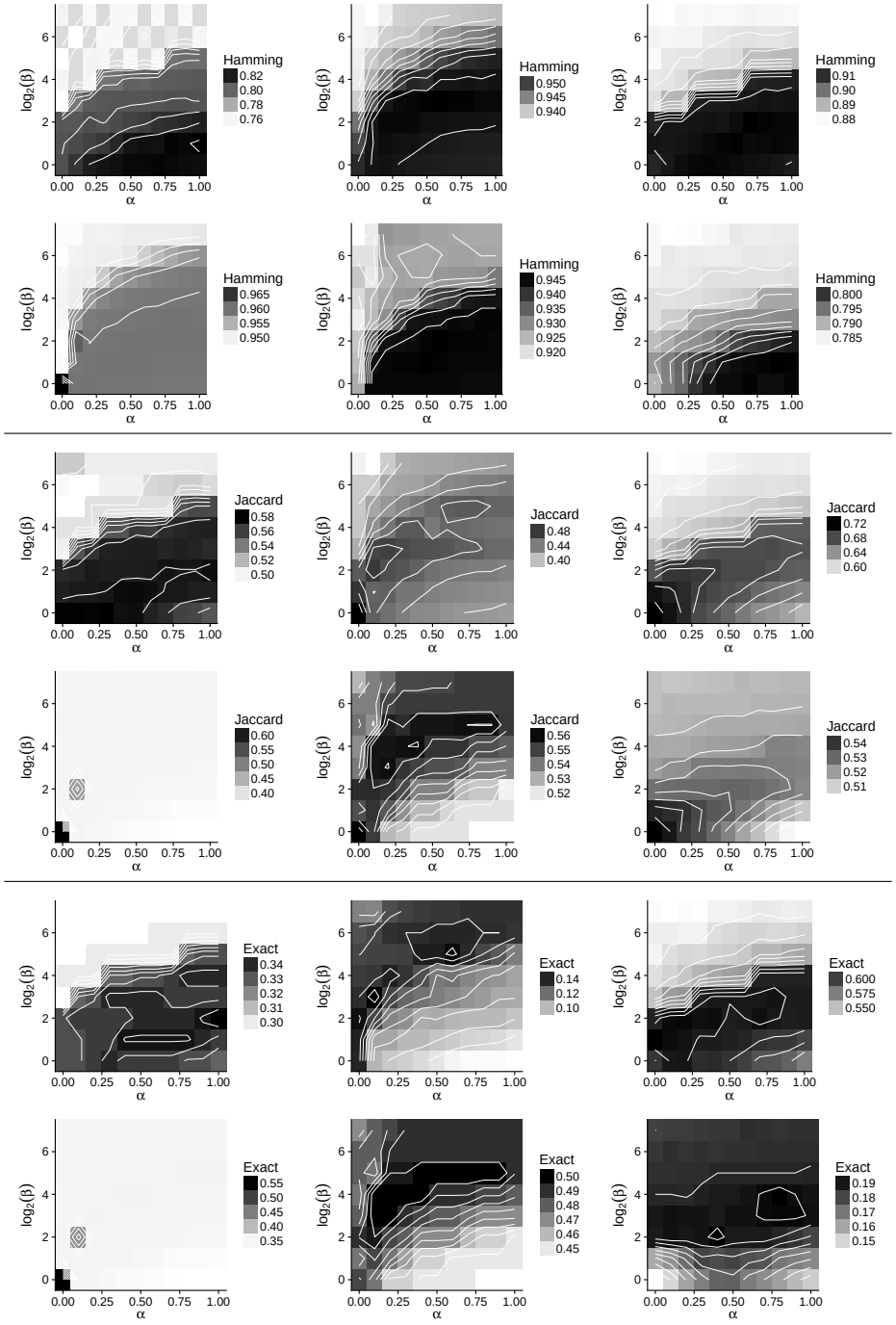

Fig. 1. The Hamming (top section), Jaccard (middle section) and Exact (bottom section) *training* similarity of the (top row from left to right in each section) Emotions, Enron, Scene (bottom row from left to right in each section) Slashdot, Stare and Yeast data, optimised using the shown values of α and β.

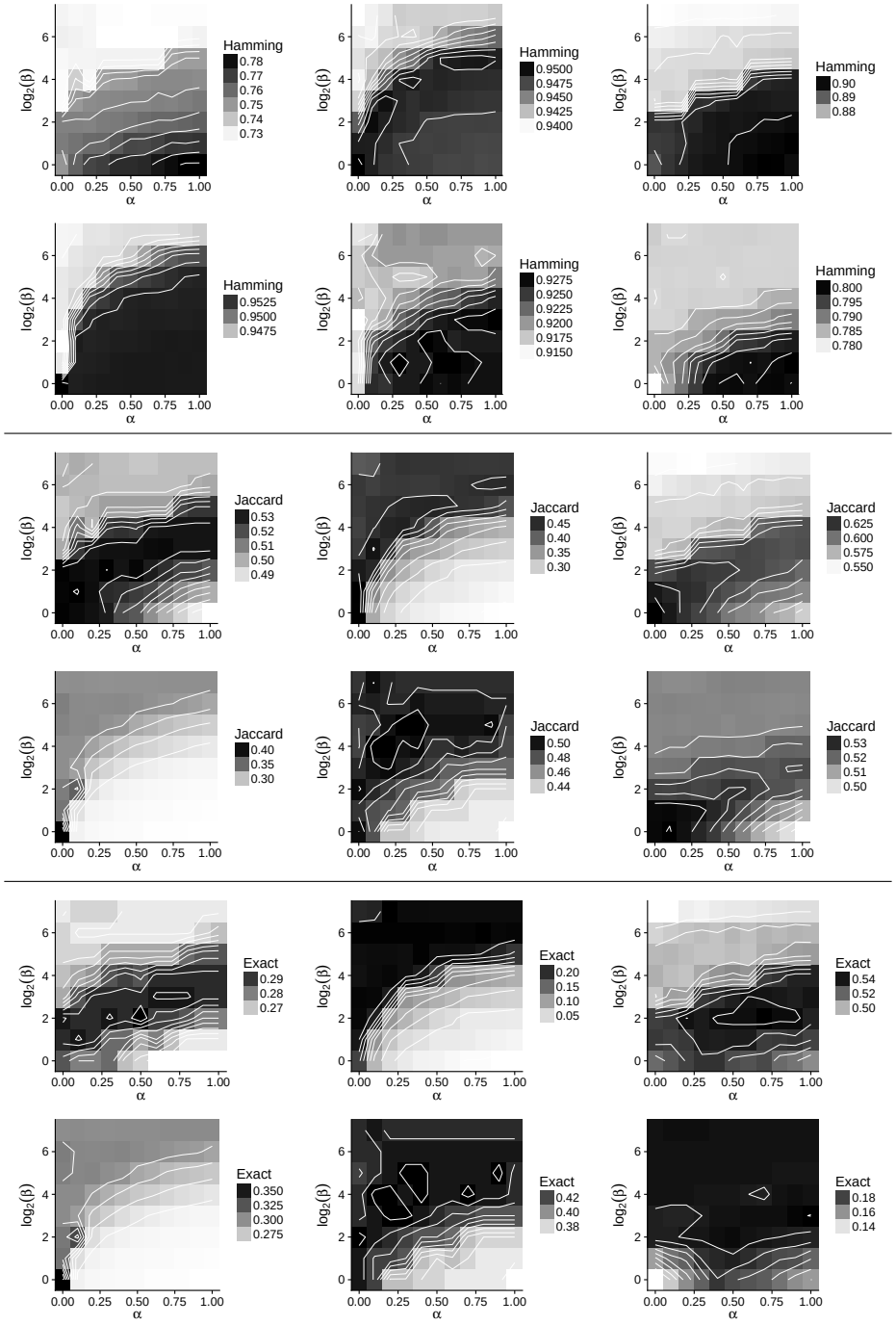

Fig. 2. The Hamming (top section), Jaccard (middle section) and Exact (bottom section) *testing* similarity of the (top row from left to right in each section) Emotions, Enron, Scene (bottom row from left to right in each section) Slashdot, Stare and Yeast data, optimised using the shown values of α and β.

Data	Type	α	β	Jaccard	Base
Emotions	Train	0.3	1	0.580	0.580
	Test	0.1	2	0.535	0.535
Stare	Train	1e-05	1	-	0.561
	Test	0.1	16	0.505	0.501
Scene	Train	1e-05	1	-	0.720
	Test	1e-05	1	-	0.643
Yeast	Train	1e-05	1	-	0.551
	Test	1e-05	2	0.534	0.533
Slashdot	Train	1e-05	1	-	0.628
	Test	1e-05	1	-	0.407
Enron	Train	1e-05	1	-	0.514
	Test	1e-05	2	0.489	0.488

Table 2. The location (α, β) of the optimal Jaccard scores, and their Base score ($\alpha = 0$, $\beta = 1$). A dash means that the optimal score was located at the Base score position.

the data sets for both training and testing. For the remaining cases, we find that the accuracy difference between the $\alpha = 1$ and $\beta = 1$ and optimal configuration is small (of the order of 0.01 or less).

The middle sections of Figures 1 and 2 show the Jaccard scores for each of the training and testing data sets. The model optimises over Jaccard similarity when $\alpha \to 0$ and $\beta = 1$. Therefore it is expected that the Jaccard scores are optimal or close to optimal for precisely these values (bottom left corners of the block plots).

The plots show that the Jaccard score is high in the lower left corner, but it also remains high as both α and β increase (moving diagonally along the block plot). Slashdot training is a notable exception, where the lower left corner seems to be a "sweet spot", obtaining an approximate 0.2 increase in Jaccard similarity over the rest of the plot. We note that this is not as dramatic for the Slashdot test data, but the lower left corner still provides a high Jaccard score relative to the remaining configurations.

The lower sections of Figures 1 and 2 show the Exact match scores for each of the training and testing sets. Exact similarity is being optimised when $\beta \to \infty$ (the top of the plots). However, these plots show interesting and surprising behaviour, with many of them being similar to the Jaccard plots. In other cases there does not seem to be consistency in the optimal regions, which seems to be located at about $\alpha = 0.5$, with fluctuating values of β.

We now proceed to examine the difference in accuracy of the optimal α, β configuration and expected α, β configuration. Tables 2, 3 and 4 contain these details for the Jaccard, Hamming and Exact similarity functions.

Table 2 shows the Jaccard scores, where Base is the score for the expected optimal configuration ($\alpha \to 0$, $\beta = 1$). We find that seven of the twelve data sets provide the optimal score at the expected configuration, while the remaining five provide a very slight increase (at most 0.004) over the Base score. Thus we

Data	Type	α	β	Hamming	Base
Emotions	Train	1	2	0.827	0.823
	Test	1	1	-	0.783
Stare	Train	0.6	4	0.946	0.944
	Test	0.3	2	0.927	0.927
Scene	Train	0.7	4	0.916	0.912
	Test	0.8	1	0.901	0.899
Yeast	Train	0.9	1	0.804	0.804
	Test	1	2	0.799	0.798
Slashdot	Train	1e-05	1	0.968	0.959
	Test	1e-05	1	0.954	0.953
Enron	Train	0.6	8	0.954	0.952
	Test	1e-05	1	0.950	0.947

Table 3. The location (α, β) of the optimal Hamming scores, and their Base score ($\alpha = 1$, $\beta = 1$). A dash means that the optimal score was located at the Base score position.

confirm that when evaluating using Jaccard similarity, we should optimise with respect to Jaccard similarity.

Table 3 shows the Hamming scores, where Base is the score for the expected optimal configuration ($\alpha = 1$, $\beta = 1$). Similar conclusions can me made for Hamming similarity: when evaluating using this metric, we should likewise optimise using Hamming similarity; as expected.

Table 4 shows the Exact scores, where BaseH and BaseJ give the scores for the expected optimal configuration ($\alpha = 1$ or $\alpha \to 0$, $\beta = 2^7$). Only one of the data sets provides the optimal score at the expected $\beta = 2^7$. We also find that there are large differences between the optimal scores and the BaseH and BaseJ scores. There are many values of small α (nine values ≤ 0.2). But it is clearly seen that the selection of α and β for optimal Exact similarity is dependent on the data.

Therefore, our findings may be summarised as the following recommendations: if optimising Hamming: use $\alpha = 1$, $\beta = 1$; if Jaccard: use $\alpha \to 0$, $\beta = 1$; and – we emphasise – for Exact similarity: we should in fact explore the α and β space. This has important and far-reaching implications, since Exact similarity is a widely used metric, and often used to promote novel classifiers because classifiers that model labels together are more likely to outperform the independent baseline under this metric. With a more careful and exploratory optimisation scheme as we define, we have showed how it is possible to achieve even higher predictive performance.

5.1 Conclusions and Future Work

We have analysed multi-label evaluation, and outlined the difficulties in optimising several of the well-known loss metrics. To tackle the issues that arise in

Data	Type	α	β	Exact	BaseJ	BaseH
Emotions	Train	1	4	0.348	0.296	0.296
	Test	0.5	4	0.297	0.265	0.265
Stare	Train	0.1	8	0.500	0.470	0.490
	Test	1e-05	4	0.439	0.427	0.427
Scene	Train	1e-05	2	0.624	0.526	0.532
	Test	0.7	4	0.545	0.480	0.489
Yeast	Train	0.1	8	0.192	0.185	0.186
	Test	0.2	16	0.194	0.191	0.190
Slashdot	Train	1e-05	1	0.553	0.351	0.351
	Test	1e-05	1	0.360	0.300	0.301
Enron	Train	0.1	8	0.150	0.115	0.137
	Test	0.2	128	0.239	0.205	0.229

Table 4. The location (α, β) of the optimal Exact scores, and their Jaccard Base score BaseJ ($\alpha = 0$, $\beta = 128$) and Hamming Base score BaseH ($\alpha = 1$, $\beta = 128$).

multi-label optimisation, we proposed a surrogate loss, in the form of a blended metric. This blended metric forms a smooth spectrum between the Hamming and exact match metrics, where it falls depending on its parameterisation. Using particular parameterisations of this function, we show that optimisation is more effective, on account of its smoothness. Indeed, for example we demonstrated that one can obtain better results under exact match by optimising the blended metric, than by optimising exact match directly (which is difficult to do). This is important because exact match is a common metric in the literature and many empirical evaluations are based around it. We also made a series of other recommendations to multi-label researchers, in reflection of our findings.

To fully evaluate the potential of our proposal under a complete range of contexts, further experimental comparison will be necessary, for example with structured prediction like probabilistic classifier chains under different inference algorithms, and structural SVMs. We cannot claim that our proposal optimises in a Bayes optimal way, due to the approximation used; Further theoretical analysis is needed on this front, particularly under exact similarity, Jaccard, and F-measure. Finally, we can point out that with some modification, the blended metric could be used to maximize F-measure – a promising line of future investigation.

References

1. K. Dembczyński, W. Cheng, and E. Hüllermeier. Bayes optimal multilabel classification via probabilistic classifier chains. In *ICML '10: 27th International Conference on Machine Learning*, pages 279–286, Haifa, Israel, June 2010. Omnipress.
2. K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier. On label dependence and loss minimization in multi-label classification. *Mach. Learn.*, 88(1-2):5–45, July 2012.

3. S. Godbole and S. Sarawagi. Discriminative methods for multi-labeled classification. In *PAKDD '04: Eighth Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 22–30. Springer, 2004.
4. C. Largeron, M. Géry, and C. Moulin. MCut: A Thresholding Strategy for Multi-label Classification. In *Eleventh International Symposium on Intelligent Data Analysis (IDA 2012)*, pages 173–184, Helsinki, Finland, Oct. 2012.
5. G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. Džeroski. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, 45(9):3084–3104, Sept. 2012.
6. U. T. Nguyen, A. Bhuiyan, L. A. F. Park, R. Kawasaki, T. Y. Wong, J. J. Wang, P. Mitchell, K. Ramamohanarao, et al. An automated method for retinal arteriovenous nicking quantification from color fundus images. *IEEE Trans. Biomed. Engineering*, 60(11):3194–3203, 2013.
7. L. A. F. Park and S. Simoff. Using entropy as a measure of acceptance for multi-label classification. In E. Fromont, T. De Bie, and M. van Leeuwen, editors, *Advances in Intelligent Data Analysis XIV*, pages 217–228. Springer, 2015.
8. J. Read, L. Martino, and D. Luengo. Efficient Monte Carlo methods for multi-dimensional learning with classifier chains. *Pattern Recognition*, 47(3):1535–1546, 2014.
9. J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. *Machine Learning*, 85(3):333–359, 2011.
10. J. Read, A. Puurula, and A. Bifet. Multi-label classification with meta labels. In *ICDM'14: IEEE International Conference on Data Mining (ICDM 2014)*, pages 941–946. IEEE, December 2014.
11. G. Tsoumakas, I. Katakis, and I. Vlahavas. Mining multi-label data. In O. Maimon and L. Rokach, editors, *Data Mining and Knowledge Discovery Handbook*. 2nd edition, Springer, 2010.
12. G. Tsoumakas, I. Katakis, and I. Vlahavas. Random k-labelsets for multi-label classification. *IEEE Transactions on Knowledge and Data Engineering*, 23(7):1079–1089, 2011.
13. W. Waegeman, K. Dembczyński, A. Jachnik, W. Cheng, and E. Hüllermeier. On the bayes-optimality of f-measure maximizers. *J. Mach. Learn. Res.*, 15(1):3333–3388, Jan. 2014.
14. X. Wu and Z. Zhou. A unified view of multi-label performance measures. In *ICML*, volume 70, pages 3780–3788. PMLR, 2017.
15. M.-L. Zhang and Z.-H. Zhou. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837, 2014.