On Missing Labels, Long-tails and Propensities in Extreme Multi-label Classification

Erik Schultheis
Aalto University
Helsinki, Finland
erik.schultheis@aalto.fi

Marek Wydmuch
Poznan University of Technology
Poznan, Poland
mwydmuch@cs.put.poznan.pl

Rohit Babbar
Aalto University
Helsinki, Finland
rohit.babbar@aalto.fi

Krzysztof Dembczyński*
Yahoo! Research
New York, USA
kdembczynski@cs.put.poznan.pl

ABSTRACT

The propensity model introduced by Jain et al. [18] has become a standard approach for dealing with missing and long-tail labels in extreme multi-label classification (XMLC). In this paper, we critically revise this approach showing that despite its theoretical soundness, its application in contemporary XMLC works is debatable. We exhaustively discuss the flaws of the propensity-based approach, and present several recipes, some of them related to solutions used in search engines and recommender systems, that we believe constitute promising alternatives to be followed in XMLC.

CCS CONCEPTS

• **Computing methodologies** → **Supervised learning by classification**.

KEYWORDS

extreme classification, multi-label classification, propensity model, missing labels, long-tail labels, recommendation

ACM Reference Format:
Erik Schultheis, Marek Wydmuch, Rohit Babbar, and Krzysztof Dembczyński. 2022. On Missing Labels, Long-tails and Propensities in Extreme Multi-label Classification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3534678.3539466

1 INTRODUCTION

Extreme multi-label classification (XMLC) is a supervised learning problem where only a few labels from an enormous label space, reaching orders of millions, are relevant per data point. Notable examples are tagging of text documents [1], content annotation for multimedia search [13], and diverse types of recommendation, including webpages-to-ads [6], ads-to-bid-words [25], users-to-items [42], queries-to-items [22], or items-to-queries [10]. These practical applications pose statistical challenges, including: 1) long-tailed distribution of labels—infrequent (tail) labels are much harder to predict than frequent (head) labels due to data imbalance, and a model completely ignoring the tail labels can get very high scores on standard performance metrics; 2) missing relevant labels in the observed training data—since it is nearly impossible to check the whole set of labels when it is so large.

To address the latter issue, propensity-scored versions of popular measures (i.e., precision@k and nDCG@k) were introduced by Jain et al. [18]. Under the propensity model, it is assumed that an assignment of a label to an example is always correct, but the supervision may skip some positive labels, and propensity of a label refers to the probability of not skipping that label. Under the implicit assumption that the chance for a label to be missing is higher for tail than for head labels, the propensity-scored measures were used to evaluate the prediction performance on tail labels. Despite being originally introduced to study the phenomenon of missing labels in XMLC, over the years, they have found their way into the literature as default performance metrics on tail labels [3, 16, 40].

In this work, we take a step back and thoroughly investigate the validity of the propensity model of Jain et al. [18], further referred to as JPV (from the first letters of authors' names), for the dual usage of missing and long-tail labels in XMLC. We start our discussion by recalling the definition of the XMLC problem, stating the problem of missing labels, and bringing closer the issues with long-tail labels (Section 2). We recall the JPV propensity model and highlight its shortcomings in Section 3, both in terms of the model itself and in regard to its current usage in XMLC. In particular, we demonstrate that this model: (i) does not fulfill natural conditions that may be desired of a reasonable propensity model, (ii) falls short on reliable and reproducible estimation of the model hyper-parameters, and (iii) leads to implausible results exceeding substantially the natural range of the metrics (e.g., precision@k > 300%).

After formally studying the above short-comings of the JPV model, we propose a suite of alternatives (c.f. Section 4) which are promising to follow for a more principled approach in (i) evaluating machine learning systems trained on data with incomplete user feedback, and (ii) disentangling the individual contribution of missing and tail labels in XMLC. We suggest using unbiased sets

*Also with Poznan University of Technology.

KDD '22, August 14–18, 2022, Washington, DC, USA.

ACM ISBN 978-1-4503-9385-0/22/08.
https://doi.org/10.1145/3534678.3539466

for validating models designed to deal with missing labels. Alternatively, one should use a set with a controlled bias, on which one can obtain unbiased estimates of performance metrics. Thereafter, we discuss alternative propensity models, which possess desirable analytical properties, and compare them with the JPV model empirically, confirming its shortcomings. We also show the efficacy of a framework in which label propensities and parameters of the learning model are learned jointly. Towards disambiguating the phenomenon of missing and long-tail labels in XMLC, we finally highlight other metrics as possible options for measuring tail-label performance instead of conflating these with missing labels.

It should be noted that, unlike most contemporary advances in XMLC, our goal in this work is not algorithmic. Instead, we take a critical viewpoint and study the commonly-used propensity model, explicating the consequences when it is used in real-world production environments.

2 PROBLEM STATEMENT

We first define the problem of XMLC, then the problem of missing labels, and finally the problem of long-tail labels.

2.1 Extreme multi-label classification

The goal of XMLC is to find a mapping between instances $X \in \mathcal{X}$ and a finite set of m non-mutually-exclusive class labels.[1] This means that any specific realization x of X is associated with a (possibly empty) subset $\mathcal{L}(x) \subset [m]$ of the labels called the *relevant* or *positive* labels, with the complement, $[m] \setminus \mathcal{L}(x)$, of the *irrelevant* or *negative* ones. We identify the relevant labels with a binary vector $\boldsymbol{y} \in \mathcal{Y}$ through $y_j = \mathbb{1}[j \in \mathcal{L}(x)]$,[2] where $\mathcal{Y} \coloneqq \{0,1\}^m$ is called the *label vector space*. In the classical setting, we assume that observations $(X, \boldsymbol{Y})$ are generated independently and identically according to a probability distribution $\mathbb{P}$ on $\mathcal{X} \times \mathcal{Y}$. In case of XMLC we assume m to be a large number (e.g., $\geq 10^5$), and $\|\boldsymbol{Y}\|_1$ to be much smaller than m, $\|\boldsymbol{Y}\|_1 \ll m$.

The problem of XMLC can be defined as finding a *classifier* $h : \mathcal{X} \longrightarrow \mathbb{R}^m$ which minimizes the *task risk*:

$$\mathrm{Risk}_{\ell_{\text{task}}}[h; X, \boldsymbol{Y}] \coloneqq \mathbb{E}[\ell_{\text{task}}(\boldsymbol{Y}, h(X))] \,, \tag{1}$$

where $\ell_{\text{task}} : \mathcal{Y} \times \mathbb{R}^m \longrightarrow \mathbb{R}_{\geq 0}$ is the (*task*) *loss*. The optimal (*Bayes*) classifier for a loss ℓ_{task} is given by

$$h^*(x) = \operatorname*{arg\,min}_{\hat{\boldsymbol{y}} \in \mathbb{R}^m} \mathbb{E}[\ell_{\text{task}}(\boldsymbol{Y}, \hat{\boldsymbol{y}}) \mid X = x] \,. \tag{2}$$

The above definitions follow the standard statistical learning framework. Let us notice, however, that in XMLC, instead of loss functions, one often uses performance metrics, which are rather maximized than minimized. Moreover, these definitions correspond to the most natural setting in which a decision is made based on a single x. Later in the paper, we also consider more general metrics that cannot be optimized with respect to individual instances.

Typically, a task loss is hard to optimize and one chooses instead a *surrogate loss* that is easier to cope with, e.g., because it is differentiable and convex. Furthermore, instead of a probability distribution, a learning algorithm operates on a finite i.i.d. sample and minimizes the corresponding empirical risk.

[1] We use capital letters for random variables, and calligraphic letters for sets.
[2] $\mathbb{1}[\cdot]$ is the indicator function.

2.2 Missing labels

In XMLC, the observed data might not follow the distribution we want to learn about. As an illustrative example, take the Wikipedia-500k dataset. The content of a Wikipedia article should be matched with a set of categories the article belongs to. Such a dataset can be easily created by scraping existing Wikipedia annotations. However, there are about 500 000 categories on Wikipedia, and it is clear that the original authors and curators have never checked every single category for each article.[3] On the other hand, each category that has been assigned to an article has been verified by a human to be relevant. Therefore, the labeling error can be assumed to be strongly one-sided: There may be many missing labels, but spurious labels should be uncommon.

To contrast ground-truth labels $\boldsymbol{Y}$ with those actually available, we denote the *observed labels* $\tilde{\boldsymbol{Y}}$ using a tilde.[4] Mathematically, the setting studied in this paper is defined by

$$\mathbb{P}\big[\tilde{\boldsymbol{Y}} \preceq \boldsymbol{Y} \mid X\big] = 1, \qquad \mathbb{P}\big[\tilde{\boldsymbol{Y}} \not\preceq \boldsymbol{Y} \mid X\big] = 0, \tag{3}$$

where $\tilde{\boldsymbol{Y}} \preceq \boldsymbol{Y}$ means that $\tilde{Y}_j \leq Y_j$ for all $j \in [m]$, and $\tilde{\boldsymbol{Y}} \not\preceq \boldsymbol{Y}$ means that there is at least one label for which $\tilde{Y}_j > Y_j$. Notice that the above equations cover also the no noise case, as we may have $\mathbb{P}\big[\tilde{\boldsymbol{Y}} = \boldsymbol{Y} \mid X\big] = 1$.

Reconstruction of the ground truth distribution from the observed one, in the general case, is not a trivial task from the statistical and computational perspective, as it requires an exponential number of parameters. Let $\eta_{\boldsymbol{y}}(x) \coloneqq \mathbb{P}[\boldsymbol{Y} = \boldsymbol{y} \mid X = x]$ and $\tilde{\eta}_{\boldsymbol{y}}(x) \coloneqq \mathbb{P}\big[\tilde{\boldsymbol{Y}} = \boldsymbol{y} \mid X = x\big]$. We have then

$$\tilde{\eta}_{\tilde{\boldsymbol{y}}}(\boldsymbol{x}) = \sum_{\boldsymbol{y}} p_{\tilde{\boldsymbol{y}}}(\boldsymbol{y}, x) \eta_{\boldsymbol{y}}(\boldsymbol{x}) \,, \tag{4}$$

where $p_{\tilde{\boldsymbol{y}}}(\boldsymbol{y}, x) \coloneqq \mathbb{P}\big[\tilde{\boldsymbol{Y}} = \tilde{\boldsymbol{y}} \mid \boldsymbol{Y} = \boldsymbol{y}, X = x\big]$ is a propensity of observing $\tilde{\boldsymbol{y}}$ for ground-truth labels $\boldsymbol{y}$ and instance x. Notice that from (3) we have $p_{\tilde{\boldsymbol{y}}}(\boldsymbol{y}, x) = 0$ for $\tilde{\boldsymbol{y}} \not\preceq \boldsymbol{y}$. Furthermore, let $\boldsymbol{\eta}_{\mathcal{Y}}(x)$ and $\tilde{\boldsymbol{\eta}}_{\mathcal{Y}}(x)$ be vectors of $\eta_{\boldsymbol{y}}(x)$ and $\tilde{\eta}_{\boldsymbol{y}}(x)$, respectively, for all $\boldsymbol{y} \in \mathcal{Y}$ given in some predefined order π. Let C be a matrix containing all propensities $p_{\tilde{\boldsymbol{y}}}(\boldsymbol{y}, x)$, with rows and columns corresponding to $\tilde{\boldsymbol{Y}}$ and $\boldsymbol{Y}$, respectively, and organized according to π. Then, we get:

$$\tilde{\boldsymbol{\eta}}_{\mathcal{Y}}(x) = \mathrm{C} \boldsymbol{\eta}_{\mathcal{Y}}(x) \,, \tag{5}$$

and, finally:

$$\boldsymbol{\eta}_{\mathcal{Y}}(x) = \mathrm{C}^{-1} \tilde{\boldsymbol{\eta}}_{\mathcal{Y}}(x) \,, \tag{6}$$

where we need to assume that C is invertible.

Because of the practical reasons, a much simpler, label-wise, propensities are commonly used that are defined for each label separately:

$$p_j(X) \coloneqq \mathbb{P}\big[\tilde{Y}_j = 1 \mid Y_j = 1, X\big] \,. \tag{7}$$

Let $\tilde{\eta}_j(x) \coloneqq \mathbb{P}\big[\tilde{Y}_j = 1 \mid X = x\big]$ and $\eta_j(x) \coloneqq \mathbb{P}\big[Y_j = 1 \mid X = x\big]$. We have then:

$$\tilde{\eta}_j(x) = p_j(x) \eta_j(x) \,, \quad \eta_j(x) = \tilde{\eta}_j(x) / p_j(x) \,. \tag{8}$$

If propensities are known, then they can be used to construct an unbiased, task or surrogate, loss $\tilde{\ell}$ [35] in the sense that

$$\forall h : \ \mathrm{Risk}_{\ell}[h; X, \boldsymbol{Y}] = \mathrm{Risk}_{\tilde{\ell}}[h; X, \tilde{\boldsymbol{Y}}]. \tag{9}$$

[3] If it took a human one second to check a category for an article, then annotating a single article fully would take almost 6 days.
[4] Note that other papers, including [18], use often a slightly different notation.

Table 1: Imbalance characteristics of typical XMLC datasets.

Dataset	Instances	min IR	ILIR	Pos-80%
Eurlex-4K	$1.55 \cdot 10^4$	15.0	$1.01 \cdot 10^3$	19.9
AmazonCat-13K	$1.19 \cdot 10^6$	3.3	$3.55 \cdot 10^5$	4.8
Wiki10-31K	$1.41 \cdot 10^4$	1.2	$1.14 \cdot 10^4$	19.6
Delicious-200K	$1.97 \cdot 10^5$	3.0	$6.45 \cdot 10^4$	4.0
WikiLSHTC-325K	$1.78 \cdot 10^6$	6.1	$2.94 \cdot 10^5$	20.7
Wikipedia-500K	$1.81 \cdot 10^6$	6.5	$2.80 \cdot 10^5$	25.1
Amazon-670K	$4.90 \cdot 10^5$	268.0	$1.83 \cdot 10^3$	54.3
Amazon-3M	$1.72 \cdot 10^6$	143.0	$1.20 \cdot 10^4$	26.1

The construction of the unbiased counterpart depends on the form of propensities, e.g., the label-wise propensities (7) are sufficient for losses decomposable over labels [24] like Hamming loss or binary cross-entropy, but might not be for more complex losses without additional assumptions [30]. The unbiased losses can be used in training procedures [18, 26] or for estimating the performance of classifiers. For some losses, such as Hamming loss or precision@k, the Bayes classifier can be written as a function of the conditional label distributions $\eta_j(x)$. In this case, one can adjust existing inference procedures to use (8) to obtain estimates of $\eta_j(x)$ from estimates of $\tilde{\eta}_j(x)$ [37].

2.3 Long-tailed label distribution

A defining characteristic of extreme classification data is that the label distribution is highly imbalanced. In the binary case, the amount of imbalance is completely determined by the imbalance ratio $\frac{\mathbb{P}[Y=0]}{\mathbb{P}[Y=1]}$. In this sense, almost every binary problem corresponding to a label is highly imbalanced in XMLC, i.e., only a small fraction of training instances will be associated with that label. However, in XMLC, the data are also imbalanced when comparing different labels. In analogy to the binary case, we can define an inter-label imbalance ratio through ILIR $= \frac{\max\{\mathbb{P}[Y_i=1]:i\in[m]\}}{\min\{\mathbb{P}[Y_j=1]:j\in[m]\}}$.[5] Nevertheless, the imbalance factor does not cover an important property of the label distribution. It could be that most labels have a large number of positives, but some have very few, or vice-versa. The latter case is what happens in XMLC, where the label distribution is said to be *long-tailed* [2, 8].

In Table 1, these imbalance measures are shown for several XMLC datasets. We use *min IR* to denote the binary imbalance ratio of the head label, i.e., the label with the largest fraction of positive instances (therefore, its IR is the smallest), and Pos-80% to indicate the minimum fraction of class labels that retain 80% of positive labels (i.e., $y_j = 1$) in the training set. For example, in Delicious-200K, only four percent of the class labels contain 80% of the positive labels.

In addition to the number of positive instances of a sparse label, their distribution within the feature space can be very important. If Y_j has few positives, but in a small pocket $\mathcal{X}_+$ of the feature space it still fulfills $\mathbb{P}\left[Y_j = 1 \mid X \in \mathcal{X}_+\right] \approx 1$, then a learning algorithm might still learn a reasonable decision boundary, especially if the overall number of samples is large enough [15, p. 23]. In contrast, if $\mathbb{P}\left[Y_j = 1 \mid X\right] \ll 1$ everywhere (called *uniform class imbalance* by Singh and Khim [31]), learning to recognize the given class might be infeasible.

[5] This concept is also used in *multiclass* classification, e.g. the *imbalance factor* of Cui et al. [11].

3 CRITICAL VIEW ON THE CURRENT APPROACH TO SPARSE LABELS IN XMLC

In this section we present an overview on the current state of addressing the long-tail and missing-labels problems in XMLC. This is in large parts based on the work of Jain et al. [18], so we start by a recap of their findings.

3.1 Current approach to missing labels and long tails

The goal of Jain et al. [18] was to develop loss functions for XMLC that

> (a) prioritize predicting the few relevant labels over the large number of irrelevant ones; (b) do not erroneously treat missing labels as irrelevant [...] (c) promote the accurate prediction of infrequently occurring, hard to predict, but rewarding tail labels.

There are two main contributions of the paper that are relevant for our discussion: First, the development of unbiased loss functions that allow compensating for missing labels if their propensities are known, and second, an empirical model to estimate these propensities on XMLC data.

3.1.1 Propensity-scored losses. Popular XMLC performance metrics focus on the highest scored labels by the prediction algorithm. Examples of such metrics are *precision at k* (P@k), *recall at k* (R@k), or *(normalized) discounted cumulative gain* nDCG@k. For these metrics, unbiased estimates in the sense of (9) can be calculated, which are called the *propensity-scored* (PS) variants of these metrics (more examples in [18, Table 1]). Table 2 gives the formal definitions, where top_k maps a vector to the indices of its top-k components, $\text{r}_j(\hat{\boldsymbol{y}})$ gives the ranking of the j-th element in the vector, and p_j is the propensity for label j. Let us notice that, in general, p_j shall depend on x, but Jain et al. [18] practically assume p_j to be a constant value for each label j. Moreover, of the above unbiased estimates, only PSP and PSnDCG have found widespread use [7], because PSR still requires the knowledge of the total number of relevant labels $\|\boldsymbol{y}\|_1$.

Table 2: Definitions of popular XMLC performance metrics and their unbiased estimates on missing labels.

Measure	Definition	Unbiased estimate
P@$k(\boldsymbol{y}, \hat{\boldsymbol{y}})$	$k^{-1} \sum_{j\in\text{top}_k(\hat{\boldsymbol{y}})} y_j$	$k^{-1} \sum_{j\in\text{top}_k(\hat{\boldsymbol{y}})} \tilde{y}_j / p_j$,
R@$k(\boldsymbol{y}, \hat{\boldsymbol{y}})$	$\|\boldsymbol{y}\|_1^{-1} \sum_{j\in\text{top}_k(\hat{\boldsymbol{y}})} y_j$	$\|\boldsymbol{y}\|_1^{-1} \sum_{j\in\text{top}_k(\hat{\boldsymbol{y}})} \tilde{y}_j / p_j$
nDCG@$k(\boldsymbol{y}, \hat{\boldsymbol{y}})$	$\frac{\sum_{j\in\text{top}_k(\hat{\boldsymbol{y}})} \frac{y_j}{\log(\text{r}_j(\hat{\boldsymbol{y}})+1)}}{\sum_{j=1}^{k} \frac{1}{\log(j+1)}}$	$\frac{\sum_{j\in\text{top}_k(\hat{\boldsymbol{y}})} \frac{\tilde{y}_j}{p_j \log(\text{r}_j(\hat{\boldsymbol{y}})+1)}}{\sum_{j=1}^{k} \frac{1}{\log(j+1)}}$

Because Jain et al. [18] observed that the unbiased estimates could results in values larger than one, they suggest a normalized

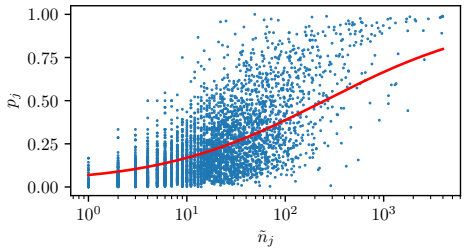

Figure 1: Reproduced estimates of propensities for Wikipedia-500K dataset using labels hierarchy and propensity function ϕ_{JPV} with $a = 0.5$ and $b = 0.4$ as estimated by Jain et al. [18] for this dataset.

version of these metrics to be reported (cf. Section 3.4). In subsequent literature, the distinction between the unbiased metrics and the normalized versions is not always preserved, e.g., Bhatia et al. [7] present unbiased formulas but lists normalized values.

3.1.2 Empirical propensity model. In order to use the propensity-scored loss functions, one needs to have the propensities available for the individual labels. Since true propensities are unknown for the XMLC benchmark datasets, Jain et al. [18] proposed to model propensities as a function of labels frequencies, resulting in propensities being a constant value for each label.

Let ϕ denote a propensity model. The model defined in [18] can be expressed via label priors $\tilde{\pi}_j := \mathbf{P}[\tilde{Y}_j = 1]$:

$$p_j = \phi_{\text{JPV}}(\tilde{\pi}_j; n, a, b) := \frac{1}{1 + (\log n - 1)(b+1)^a e^{-a \log(n\tilde{\pi}_j + b)}}, \quad (10)$$

where n is the number of training instances, and a and b are dataset-dependent parameters.

In order to arrive at this model and determine values for a and b, Jain et al. [18] investigated two datasets in which ancillary information could be used to identify some missing labels.

For a Wikipedia-based dataset, the parameters of the model have been estimated with the help of a label hierarchy. They assumed that if a label is relevant to an article, then all its ancestors in the hierarchy should also be relevant. If not present, they are counted as missing. This allows plotting the fraction of instances in which the label is missing over the number of instances in which it appears. This seems to follow a sigmoidal shape as described by (10), see Figure 1. The parameters a and b were then determined by fitting the model against the estimated values, where only labels with more than 4 descendants were used to improve robustness. The obtained values are $a = 0.5, b = 0.4$.

For the Amazon data set, which is an item-to-item recommendation task, missing labels have been approximated using "also viewed" and "also bought" information. It was assumed that a label j (an item) is relevant to all the items viewed along with items that were also bought with label j, as proposed by McAuley et al. [21]. The obtained values are $a = 0.6$ and $b = 2.6$.

For other data sets the authors propose, if there is no other possibility of estimating parameters a and b, to use averages of the values obtained for Wikipedia and Amazon data sets (which are $a = 0.55$, $b = 1.5$). This, in fact, has become a standard followed in many papers without questioning its rationality.

The above propensity model is then typically used in the metric of choice for model selection and evaluation. It has also been incorporated into training procedures. For example, decision tree methods can directly use the propensity-scored variants of metrics such as precision@k or nDCG@k [18]. Alternatively, one can use unbiased or upper-bounded propensity-scored surrogate losses [26].

3.1.3 Propensity and long tails. The form of (10) implies that tail labels are assigned lower propensities, which means that in metrics like those in Table 2 these tail labels, if predicted correctly, will be weighted more strongly than head labels. In particular, the resulting weightings resemble existing weighting schemes used for long-tailed learning tasks, leading the authors to conclude:

> Such weights arise naturally as inverse propensities in the unbiased losses developed in this paper. [...] This not only provides a sound theoretical justification of label weighting heuristics for recommending rare items but also leads to a more principled setting of the weights.

As a result, propensity-scored variants are also viewed as metrics in their own right, and are currently used both to counteract missing labels (as unbiased estimates) and to weigh tail labels (as independent metrics), becoming established performance metrics commonly used in XMLC.[6]

3.2 Discussion of missing-labels assumptions

In order to derive unbiased loss functions, we need to impose assumptions on the process of how labels go missing, as initially discussed in Section 2.2. Unfortunately, Jain et al. [18] sent a potentially misleading message in this regard. Their Theorem 4.1 proves

$$\mathbb{E}[\ell(\boldsymbol{Y}, \hat{\boldsymbol{y}})] = \mathbb{E}\left[\tilde{\ell}(\tilde{\boldsymbol{Y}}, \hat{\boldsymbol{y}})\right], \quad (11)$$

for any **fixed** prediction $\hat{\boldsymbol{y}}$ without a clear dependence on X. This also implies that the assumptions behind the propensities are unclear. Even if we assume the propensities to be constant for label j, the exact form of this assumption is necessary to properly prove unbiasedness in the sense of (9). Notice that $\mathbb{P}\left[\tilde{Y}_j = 1 \mid Y_j = 1\right] = p_j$ does not imply $\mathbb{P}\left[\tilde{Y}_j = 1 \mid Y_j = 1, X\right] = p_j$. Moreover, for more complex functions, such as recall@k, this assumption may take the form of $\mathbb{P}\left[\tilde{Y}_j = 1 \mid Y_j = 1, \boldsymbol{Y}_{\neg j}, X\right] = p_j$, where $\boldsymbol{Y}_{\neg j}$ represents ground-truth labels without label j (see Appendix for an example).

In general, we cannot expect the independence of missing labels from the instance's features to hold. Consider, for example, cases where the feature and label space are of a similar origin [12], such as matching Wikipedia titles or articles to categories. It seems unlikely that a label such as "Italy" would be missing for articles containing the word "Italy" in the subject, but it might be missing for articles that still pertain to Italy but do not feature the word "Italy" in

[6] We list several examples of references to propensity-scored losses: "We examined the performance on tail labels by PSP@k" [40]; "We achieve high precision and propensity scores, thus demonstrating the effectiveness of our method even on infrequent tail labels." [16]; "capture prediction accuracy of a learning algorithm at top-k slots of prediction, and also the diversity of prediction by giving higher score for predicting rarely occurring tail-labels" [3]; "propensity scored precision@k which has recently been shown to be an unbiased, and more suitable, metric" [17]; "which leads to better performance on tail labels." [39]; "propensity scored variant which is unbiased and assigns higher rewards for accurate tail label predictions", "evaluate prediction performance on tail labels using propensity scored variants" [20]; "replacing the nDCG loss with its propensity scored variant and using additional classifiers designed for tail labels" [32].

the title. The assumption that propensities are constant for each label simplifies the model significantly and leads to much simpler computational procedures. Unfortunately, if this assumption is not satisfied, then one may get implausible results as discussed later.

The assumption that the propensities do not depend on other labels going missing does not need to hold in practice as well. For example, a user that tagged the article for "Italy" with "Member states of the European Union" might be primed to think of more examples of organizations in which Italy is a member, and thus e.g., "Current member states of the United Nations" might be less likely to be forgotten than if the EU membership had been forgotten. Fortunately, in many cases, the unbiased estimate does not actually require this dependence — if the loss function can be written as a sum over contributions from each label individually, then the labels do not interact with each other and the label-wise properties are sufficient to obtain unbiased losses. This is the case for the popular PSP and PSnDCG metrics.

3.3 Shortcomings of the JPV propensity model

Let us discuss several issues of the JPV model, concerning theoretical and empirical shortcomings, as well as some problems in the way the parameters of the model have been established.

Scaling behavior. Let us first observe that (10) does not preserve propensity estimates if the amount of data is changed, without changing its characteristics, e.g., by sub- or over-sampling the dataset. In particular, if one increases the amount of available data by making multiple copies of the dataset, which should not change the estimates of label priors $\tilde{\pi}_j$ given by $\tilde{n}_j/n$ (with $\tilde{n}_j$ being the number of positive instances of label j in the observed, noisy training set), the JPV model will estimate propensities to be equal one, i.e., no missing labels, as the amount of data goes to infinity:

$$\lim_{n\to\infty} \phi_{\text{JPV}}(\tilde{\pi}_j, n) = \frac{1}{1+(b+1)^a \lim_{n\to\infty}(\log n - 1)e^{-a\log(\tilde{\pi}_j n+b)}} = \frac{1}{1+(b+1)^a \lim_{n\to\infty}(\log n)(\tilde{\pi}_j n)^{-a}} = 1. \quad (12)$$

This means that we cannot interpret a and b as parameters of some underlying (unknown) process that describes the labeling process. As we cannot even have fixed a and b when the data come from the same process, this very much calls into question the approach of using values for a and b across datasets as is current practice.

Estimation process. Setting aside structural concerns about (10), the estimation of the parameters a and b still remains an issue. First, by identifying missing labels based on meta-data as described in Section 3.1.2, only an upper-bound on the propensity is estimated, since labels may also be missing in other ways. For example, we tried to reproduce the procedure of propensity estimation on the Wikipedia dataset. We have found that only around 40 000 out of 500 000 labels meet the criteria of the sufficient number of descendants selected by the authors, and around 300 000 labels are without descendants, so they would never be considered missing by this protocol.

Further, one might argue that in cases in which missing labels can be identified by some side-channel information such as label hierarchies, then one can directly impute these missing labels and need not worry about training with missing labels.

Propensity as a function of frequency. This still leaves the question of whether such estimates are sensible. Even though there is clearly a trend that labels within a given range of frequency have – on average – a certain propensity, for each individual label the actual propensity can fluctuate widely around this mean, as shown in Figure 1 that we obtained following the original procedure for estimating propensities.

Reproducibility. The description of the process of propensity estimation in [18] is rather sparse on details. While meta-data for Wikipedia is easily obtainable, it is not clear what is the source of ancillary information that has been used for the Amazon dataset. Additionally, depending on the preprocessing steps and criteria, such as the number of descendants in the label hierarchy, one can achieve very different estimates of parameters a and b.

3.4 Implausible results and normalization

Despite being unable to verify the correctness of the assumptions and the JPV model without actual clean ground truth data, we are still able to show that the approach of Jain et al. [18] leads to implausible results. For example, PSP@k, as an unbiased estimate of P@k on the ground-truth data, should be bounded between zero and one. However, when calculating this measure for a real classifier, the result may exceed this range substantially. Of course, for an individual instance or a small subset of them, the unbiased estimate does not need to fall into that range, but a large deviation from the true value becomes exceedingly unlikely when averaging over the entire dataset.

To circumvent this issue, Jain et al. [18] suggest to report a normalized version of PSP@k, also calling this measure "propensity-scored precision". The normalization is realized by dividing the metrics value by the largest possible value that any prediction could have achieved on that data:

$$\text{Norm PSP@}k = \frac{\sum_{i=1}^{n} \text{PSP@}k(\tilde{\boldsymbol{y}}_i, \hat{\boldsymbol{y}}_i)}{\sum_{i=1}^{n} \max_{\mathbf{z}} \text{PSP@}k(\tilde{\boldsymbol{y}}_i, \mathbf{z})}. \quad (13)$$

The normalization introduces a factor that is constant over the entire dataset, and thus does not influence model selection. However, it removes the interpretation of the received value as an unbiased estimate of the metric on clean data, and it hides the model misspecification. Table 3 reports the values of both variants of PSP@k, showing how severe this issue is.

Table 3: Normalized and unnormalized propensity-scored precision of PfastreXML [18], when using the JPV model, with $a = 0.5$, $b = 0.4$ for WikiLSHTC-325K and $a = 0.6$, $b = 2.6$ for Amazon-670K.

	WikiLSTHC-325K			Amazon-670K		
PSP(%)	@1	@3	@5	@1	@3	@5
Normalized	31.16	31.80	33.35	29.93	31.26	32.80
Unnormalized	196.96	118.54	85.28	326.47	282.28	250.57

3.5 The current use of propensity metrics

It seems that the current use of propensity metrics mixes up, in a not entirely clear way, two different issues, missing and tail labels. As mentioned in Section 3.1.3, these metrics might be used for the purpose of giving more weight to tail labels. In this case, the normalization step seems to be a valid procedure. However, a propensity metric loses its original interpretation, and it is just one way of accounting for tail labels, without any concrete justification. For this use case, it would be preferable to have a metric that treats tail labels in a principled way. As a first step towards that goal, Section 4.4 provides some discussion on alternative task losses.

Only in the interpretation as a tail-performance promoting loss, it does make sense to speak of a trade-off in performance between vanilla and propensity-scored metrics, as these are conceptually different. In the missing-labels interpretation, taking the propensities into account is not calculating a different conceptual metric, but instead, the *correct* way of calculating the unweighted, but the true performance of a classifier. Of course, in XMLC, both interpretations can be combined, i.e., one would like to have a task loss that is adapted to tail labels, but calculate it in a way that takes missing labels into account. The closest to this in the literature is [26], where training uses a loss that combines unbiased estimates and class-rebalancing, but still, evaluation is performed using vanilla and propensity-scored metrics, instead of a propensity-scored variant of a tail-weighted metric.

4 RECIPES TO FOLLOW

In this section, we present several recipes on how to conduct research on missing and tail labels in XMLC. We start our discussion with a recommendation of using an additional dataset which is either unbiased or its bias is under control. We then discuss several alternatives for the JPV model, show how to fit the models to unbiased data, and how to compare them empirically. Next, we introduce methods that jointly train the prediction and the propensity model. Despite our critical remarks, we consider in this section only propensities being constant for each label. Finally, we discuss performance metrics for long-tail labels that might be a better choice than propensity-scored metrics.[7]

4.1 Bias-controlled validation and test sets

If indeed our training data are biased by missing labels, the best way to test, validate, and estimate the propensity model is to use unbiased data or data with controlled bias. In the latter case, the bias is controlled in such a way that unbiased estimates can be easily computed. Such data are definitely costly to get, but even a small set can be beneficial, helping in selecting the right model to be used in production and in estimating its real performance. This is a standard approach used in recommendation systems [29, 38] and search engines [19]. Even if, in a given application, it is not possible to obtain an unbiased data due to some constraints, we should investigate algorithms on benchmarks with unbiased or bias-controlled sets associated. Without this investigation, it is hard to verify which methods are indeed working and which are misleading. In many real-world applications, the final evaluation of the model is performed in A/B tests on real, unbiased data. We should avoid situations in which the results of A/B tests are against our expectations coming from offline experiments.

To follow the above recommendation, we have prepared several datasets of different types, which can be used for experimentation with missing labels in XMLC. The first type contains fully synthetic datasets. The second one is a modification of the standard XMLC benchmarks. The final dataset is a variant of the Yahoo R3 dataset,[8] transformed from a recommendation problem to multi-label classification. Statistics of these datasets along with additional information are given in Appendix.

The synthetic datasets are generated in a similar way as in [34]. They are parameterized by the number m of labels. In the experiments reported below, we use $m = 100$, which is not *very* extreme but suffices for investigating the propensity models. Each label j is represented by a d-dimensional hyper-ball S_j, whose radius and center are generated randomly. All those hyper-balls lay in a feature space being itself a d-dimensional hyper-ball S, big enough to contain all hyper-spheres S_j. Instances are uniformly generated in S and each instance x is associated with labels whose hyper-balls contain x, i.e., $\mathcal{L}(x) = \{j \in [m] : x \in S_j\}$. The popularity (or priors) of the labels is directly determined by the radius of S_j. We separately generate training, validation, and test sets from the above model. We then apply a propensity model of choice to the training set to generate missing labels. For some experiments, we also generate missing labels in validation and test sets.

For the original XMLC benchmark datasets, we assume that there are no missing labels. We then merge the original train and test sets, and take labels having at least s positive instances. We perform this step to select labels for which one can apply the noise models without removing all positive labels. We then split the data again into training and test sets, and apply, similarly as above, a propensity model of choice to the training set to generate missing labels. For some experiments, we extract a validation set from the test set.

In the original Yahoo R3 dataset, records are organized in a format of user-item ratings, and each record contains a user ID, an item ID, and the user's rating for the item (from 1 to 5). The training set contains over 300K ratings from 15.4K users to 1K items. This set is biased as users select items from a limited list of options recommended by some algorithm. The bias-controlled test set is obtained by collecting ratings from a subset of 5.4K users to rate $r = 10$ randomly selected items. To create a multi-label dataset we treat each item as a label ($m = 1K$). We consider ratings greater or equal to 4 as positive feedback and others as irrelevant. We take users unique to the original training set (10K of users) to create a biased multi-label training set. For each user, we randomly split positive feedback into equal halves. We take the first half as features x and the later half as labels $\boldsymbol{y}$. Next, we use users present in both the original training and test set to create a test set with a controlled bias. We again randomly select half of the positive feedbacks from the training set as features x (to keep the same distribution of features) and all positive feedback from the test set as labels $\boldsymbol{y}$. We can also extract a validation set from the test set, usually consisting of a half of users from the test set.

[7] Repository with the code to reproduce all experiments: https://github.com/mwydmuch/missing-labels-long-tails-and-propensities-in-xmlc

[8] https://webscope.sandbox.yahoo.com/

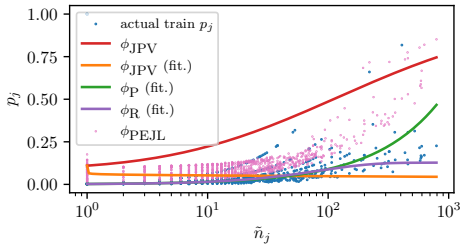

Figure 2: Propensity models on the Yahoo R3 dataset. Annotation (fit.) **denotes that the parameters have been fitted to the actual training-set propensities. The ϕ_{PEJL} model is described in Section 4.3.**

For a dataset created in such a way, we can calculate estimates of training-set propensities p_j^{tr} as:

$$\hat{p}_j^{\text{tr}} = \phi_{\text{direct}} = \hat{\pi}_j^{\text{tr}} p_j^{\text{c}} \left(\hat{\pi}_j^{\text{val}}\right)^{-1}, \tag{14}$$

where $\hat{\pi}_j^{\text{tr}}$ and $\hat{\pi}_j^{\text{val}}$ are, respectively, training- and validation-set estimates of the prior probability of label j, and p_j^{c} is the controlled propensity used for the validation and test set. To estimate the label priors, one can use relative frequencies of labels in training and validation sets. For p_j^{c} we use a ratio of r labels used for labelling to all m labels, i.e., $p_j^{\text{c}} = \frac{r}{m} = 0.01$ for the Yahoo R3 dataset.

4.2 Alternative propensity models

The JPV propensity model (10) is not the only one to consider. In fact, many different forms have been introduced in other domains [19, 29, 38]. We express the propensity models as functions of observed label priors $\tilde{\pi}_j := \mathbb{P}\left[\tilde{Y}_j = 1\right]$, without direct relation to n, which by construction avoids convergence issues discussed in Section 3.3.[9]

A propensity model used frequently in recommendation systems is given by the following power-law formulation:

$$p_j = \phi_{\text{P}}(\tilde{\pi}_j; \beta, \gamma) := \left(\beta \tilde{\pi}_j\right)^{\gamma}. \tag{15}$$

With $\beta = \max_j \tilde{n}_j / n$ we receive a model used, for example, in [29, 38], while for $\beta = \gamma = 1$ we get a very simple model which might be used, if estimation of the parameters is infeasible due to lack of unbiased or bias-controlled data. Another solution could be to use the generalized logistic function, also called Richard's curve [28], which is very flexible and its shape resembles (10):

$$p_j = \phi_{\text{R}}(\tilde{\pi}_j; c, d, e, f, g, h) := c + \frac{d - c}{\left(e + f \exp(-g\tilde{\pi}_j)\right)^{1/h}}. \tag{16}$$

The parameters of the models can be either set up according to a domain knowledge or fit using an additional unbiased or bias-controlled dataset. In the latter case, one can use standard non-linear optimization methods [4]. We fit the models to inverse propensities, minimizing squared errors $\|\hat{p}_j^{-1} - \phi(\tilde{\pi}_j)^{-1}\|^2$ using the Levenberg-Marquardt method [23]. The errors are reported in the Appendix.

[9] The simplest estimate of the priors are relative frequencies of labels, i.e., $\hat{\tilde{\pi}}_j = \tilde{n}_j / n$. As we deal with many very sparse labels, we should rather use more robust estimates, for example, $\hat{\tilde{\pi}}_j = (\tilde{n}_j + \alpha)/(n + \alpha)$.

Table 4: Actual precision@{1,3,5} (and their standard errors) on the Yahoo R3 dataset. The best results are marked in bold. The last row presents the results of the PEJL method from Section 4.3. Each experiment was repeated 25 times.

Method	P@1 (%)	P@3 (%)	P@5 (%)
$\phi_{\mathbf{1}}$	60.83 ± 2.02	54.30 ± 0.89	51.20 ± 0.75
ϕ_{JPV}	66.03 ± 1.70	56.17 ± 0.91	52.02 ± 0.70
ϕ_{JPV} (fit.)	48.58 ± 2.13	43.26 ± 0.86	40.47 ± 0.68
ϕ_{P} (fit.)	63.53 ± 1.90	54.53 ± 0.97	50.50 ± 0.73
ϕ_{R} (fit.)	71.23 ± 1.71	61.02 ± 0.77	54.03 ± 0.49
ϕ_{direct}	**73.72 ± 2.26**	**66.14 ± 0.97**	**59.59 ± 0.74**
ϕ_{PEJL}	68.09 ± 1.53	58.15 ± 1.04	53.62 ± 0.72

Figure 2 illustrates the results of fitting the different propensity models for the Yahoo R3 dataset. The actual training-set propensities have been obtained using (14). The plot clearly shows that the JPV model with $a = 0.55$ and $b = 1.5$, suggested as default values, is not a good fit to the actual propensities. The same model, but with a and b fitted to the data gives a degenerated solution because many values are out of codomain of (10) when n is that small. On the other hand, ϕ_{P} and ϕ_{R} seem to give a good fit, but still actual propensities are widely spread, suggesting that a model solely depending on label priors might not be the best choice.

We have also trained prediction models using the above propensities to see whether they help in improving (actual) precision@k on the unbiased test set. We use the one-vs-all approach in which probabilistic model $f_j(x)$, for label j, is obtained by minimizing the unbiased variant of logistic loss [26, 29]:

$$\ell(\tilde{y}_j, p_j, f_j(x)) = -\frac{\tilde{y}_j}{p_j} \log(f_j(x)) - \left(1 - \frac{\tilde{y}_j}{p_j}\right) \log(1 - f_j(x)). \tag{17}$$

The results are given in Table 4. As a baseline we also use a vanilla logistic loss which corresponds to $\phi_1(\tilde{\pi}_j) = 1$. We can observe that all propensities models, except the degenerated variant of JPV, give slightly better results than the baseline, with ϕ_{R} being clearly the best among them. On the other hand, ϕ_{direct}, which directly estimates propensity for each label using (14), significantly improves the performance (particularly for P@3 and P@5). Nevertheless, ϕ_{direct} can only work well if the unbiased or bias-controlled data are substantial. If this is not the case, one might need to use a parametric model, but the above results suggest that the dependence on label priors might not be sufficient.

Finally, we illustrate a problem of propensity mismatch on synthetic and modified benchmark datasets. We introduce noise to training data according to either the ϕ_{JPV} or ϕ_{P} model, train prediction functions using both propensities models, and report actual precision@k, computed on the unbiased test set, along with propensity-based precision@k for the same ϕ_{JPV} or ϕ_{P} model, computed on the biased test set (i.e., with the noise model applied). The results in Table 5 show that relying on propensity-based metrics can be misleading. As it should be expected, in the majority of cases,

models compatible with the metric are obtaining the best performance. However, selecting a model based on a chosen propensity-based metric can be wrong as the actual precision might be driven by a completely different propensity model.

4.3 Propensity estimation via joint learning

To minimize an unbiased loss function, such as the unbiased logistic loss (17), one needs to know propensities in advance. However, estimating them might be difficult in practice. As demonstrated above, the use of inaccurate estimates can lead to results being far away from the optimal ones.

Therefore it would be useful if propensities could be estimated directly from a biased training set. Unfortunately, this is an ill-defined problem because the absence of a label can be explained by either a small conditional probability of the label or a low propensity or both. The additional assumption needed for the propensity to be identifiable were studied before, in the areas of learning from positive and unlabeled data [14], and novelty detection [9]. The overview of the possible assumption is given by Bekker and Davis [5], where the weakest of the assumptions requires that the true distribution of negative samples for a given label cannot contain the positive distribution [9]. In these areas and under compatible assumptions, many methods for estimating the error ratio or labels priors, both directly related to propensity estimates, were proposed [5]. Recently, [41] and [33] have introduced methods for estimating the unbiased conditional label probabilities and propensities jointly on the biased training set. We refer to such methods as Propensity Estimation via Joint Learning (PEJL).

Table 5: Mismatch of propensity models: actual P@$\{1,3,5\}$ (computed on unbiased test set) and PSP@1 (computed on biased test set) of prediction models trained on data biased by ϕ_{JPV} or ϕ_{P} models. Green highlights PSP@k compatible with the used propensity model, while red highlights incompatible PSP@k. The best value in each column for a given dataset is marked in bold.

Dataset	Method	P@k (%)			PSP@k(%)	
		@1	@3	@5	(ϕ_{JPV}) @1	(ϕ_{P}) @1
Artificial-ϕ_{JPV}	ϕ_{JPV}	**78.49**	**69.74**	**58.86**	**78.66**	102.02
	ϕ_{P}	70.22	64.89	56.66	70.34	**133.60**
Artificial-ϕ_{P}	ϕ_{JPV}	75.71	67.8	56.69	**77.59**	75.71
	ϕ_{P}	**77.79**	**69.14**	**58.04**	72.02	**77.20**
EUR-Lex-ϕ_{JPV}	ϕ_{JPV}	64.75	50.64	40.57	64.94	74.75
	ϕ_{P}	**66.51**	**51.91**	**41.50**	**66.84**	**80.63**
EUR-Lex-ϕ_{P}	ϕ_{JPV}	54.23	41.72	32.99	**44.33**	52.19
	ϕ_{P}	**55.07**	**42.07**	**33.04**	42.35	**53.08**
AmazonCat-ϕ_{JPV}	ϕ_{JPV}	**86.32**	64.58	48.53	**86.60**	182.71
	ϕ_{P}	78.87	**67.78**	**53.31**	79.38	**389.35**
AmazonCat-ϕ_{P}	ϕ_{JPV}	67.32	40.86	29.78	**44.94**	64.03
	ϕ_{P}	**82.80**	**55.72**	**40.33**	44.31	**82.22**
Wiki10-ϕ_{JPV}	ϕ_{JPV}	**82.57**	**68.72**	**59.18**	**82.97**	120.76
	ϕ_{P}	78.85	65.53	57.19	80.01	**228.02**
Wiki10-ϕ_{P}	ϕ_{JPV}	**80.44**	54.8	47.06	**87.34**	**80.43**
	ϕ_{P}	79.18	**59.89**	**49.05**	71.98	78.61

Let us briefly describe the method of Teisseyre et al. [33] (cf. Appendix for description of the method of Zhu et al. [41]). It uses the fact that minimization of logistic loss leads to estimation of the posterior probability. Therefore, we can define the loss in the following way:

$$\ell(\tilde{y}_j, p_j, f_j(x)) = -\tilde{y}_j \log(p_j f_j(x)) - (1-\tilde{y}_j)\log(1-p_j f_j(x)), \quad (18)$$

where $p_j f_j(x)$ can be seen as an estimate of the actual, ground-truth, conditional probability $\eta_j(x)$, with p_j being the propensity and $f_j(x)$ the estimate of the observed conditional probability, analogously to (8). This function can be optimized not only with respect to $f_j(x)$, but also to p_j. The outline of the alternative method of Zhu et al. [41] can be found in the appendix.

We evaluate this approach on Yahoo R3 dataset. The estimated values of p_j are plotted on the Figure 2 and the last row of Table 4 presents the promising results of this approach. While the obtained estimates are overestimated, they capture the true trend. The predictive performance also looks promising, being only slightly worst than the best propensity model ϕ_{R}. This is indeed encouraging as this method does not have access to the unbiased or bias-controlled data. Figure 2 also plots the obtained propensities for each label.

4.4 Task losses for long-tails

It seems that Jain et al. [18] have introduced the propensity-scored losses rather to "promote" long-tail labels than to deal with missing labels. As such, the propensities can be seen as a kind of weighing approach that gives higher importance to less popular labels. Unfortunately, it is not clear why the weighing scheme used in [18] should be preferred over other ones. Moreover, a weighing scheme does not have to be interpreted in terms of propensities. Let us consider a weighted variant of P@k:

$$\mathrm{P@}k(\boldsymbol{y}, \hat{\boldsymbol{y}}) = k^{-1} \sum_{j \in \mathrm{top}_k(\hat{\boldsymbol{y}})} w_j \tilde{y}_j \,. \quad (19)$$

This boils down to PSP@k when $w_j = \frac{1}{p_j}$ which also implies $w_j \geq 1$. But one can use any weights that would represent the importance or gain of labels. In such a case, the weighted P@k has a natural interpretation of being an unbiased estimate of the expected gain. If tail labels are of our interest, then they should get higher weights, but actual values are rather domain-specific without a direct relation to propensities.

To finalize our discussion, let us mention several other task losses that can be used as metrics that pay special attention to long-tail labels. The macro F_β-measure defined as:

$$\mathrm{F}_\beta^{\mathrm{macro}}\left(\{\hat{\boldsymbol{y}}_i, \boldsymbol{y}_i\}_1^n\right) = \frac{1}{m} \sum_j \frac{(1+\beta^2)\sum_i y_{ij}\hat{y}_{ij}}{\beta^2 \sum_i y_{ij} + \sum_i \hat{y}_{ij}}, \quad (20)$$

puts the same weight to each label and focuses on true positives. Therefore, positive predictions on long-tail labels are important to obtain high values on this metric. One can also consider an @k version of this metric, in which only top k predictions are taken into account.

Another interesting metric, originally proposed for search engines [27], is abandonment@k defined as:

$$\mathrm{abandonment@}k(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \mathbb{1}\left[\forall j \in \mathrm{top}_k(\hat{\boldsymbol{y}}) : y_j \neq 1\right], \quad (21)$$

which encounters no error if there is at least one correctly predicted label among the k ones in the predicted set. This untypical formulation enforces diversity in the predicted set. Always predicting the two most popular but correlated, labels might be less beneficial than predicting less popular but also non-overlapping labels.

Finally, we mention the coverage metric, which directly reflects the diversity of correctly predicted labels. It is defined as a fraction of labels with at least one correct positive prediction:

$$\operatorname{cov}\left(\{\hat{\boldsymbol{y}}_i, \boldsymbol{y}_i\}_1^n\right) = m^{-1}\left|\{j \in [m] : \exists i \in [n] \text{ s.t. } y_{ij} = \hat{y}_{ij} = 1\}\right| . \quad (22)$$

This metric has already been suggested in the literature as an alternative [3, 36]. It has also been used in the original paper of Jain et al. [18], but only to motivate the propensity-based metrics.

5 CONCLUSIONS

Despite our critical comments regarding [18], we still appreciate this contribution. It was the first paper that brought direct attention to the problem of missing and long-tail labels in XMLC. The original theoretical results concerning the propensity model have motivated a lot of research in this direction. Nevertheless, we believe that missing labels and long-tail labels are rather orthogonal problems that should be solved with different tools. Obviously, labels gone missing may cause labels to be sparse, but it does not mean that a blind propensity model may solve any of these problems.

ACKNOWLEDGMENTS

Academy of Finland grant: Decision No. 347707. Computational experiments have been performed in Poznan Supercomputing and Networking Center.

REFERENCES

[1] Rahul Agrawal, Archit Gupta, Yashoteja Prabhu, and Manik Varma. 2013. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *WWW*. 13–24.
[2] Rohit Babbar and Bernhard Schölkopf. 2017. DiSMEC: Distributed Sparse Machines for Extreme Multi-label Classification. In *WSDM*. 721–729.
[3] Rohit Babbar and Bernhard Schölkopf. 2019. Data scarcity, robustness and extreme multi-label classification. *Machine Learning* 108 (09 2019).
[4] Mokhtar S. Bazaraa, Hanif D. Sherali, and Chitharanjan M. Shetty. 2006. *Nonlinear Programming: Theory and Algorithms*. Wiley.
[5] Jessa Bekker and Jesse Davis. 2020. Learning from positive and unlabeled data: a survey. *Machine Learning* 109, 4 (2020), 719–760.
[6] Alina Beygelzimer, John Langford, Yury Lifshits, Gregory B. Sorkin, and Alexander L. Strehl. 2009. Conditional Probability Tree Estimation Analysis and Algorithms. In *UAI*. 51–58.
[7] Kush. Bhatia, Kunal. Dahiya, Himanshu Jain, Anshul Mittal, Yashoteja Prabhu, and Manik Varma. 2016. The extreme classification repository: Multi-label datasets and code. http://manikvarma.org/downloads/XC/XMLRepository.html
[8] Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain. 2015. Sparse Local Embeddings for Extreme Multi-label Classification. In *NeurIPS*. 730–738.
[9] Gilles Blanchard, Gyemin Lee, and Clayton Scott. 2010. Semi-Supervised Novelty Detection. *Journal of Machine Learning Research* 11, 99 (2010), 2973–3009.
[10] Wei-Cheng Chang, Hsiang-Fu Yu, Kai Zhong, Yiming Yang, and Inderjit S. Dhillon. 2020. Taming Pretrained Transformers for Extreme Multi-label Text Classification. In *KDD*. 3163–3171.
[11] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. 2019. Class-balanced loss based on effective number of samples. In *CVPR*. 9268–9277.
[12] Kunal Dahiya, Ananye Agarwal, Deepak Saini, K Gururaj, Jian Jiao, Amit Singh, Sumeet Agarwal, Purushottam Kar, and Manik Varma. 2021. SiameseXML: Siamese Networks meet Extreme Classifiers with 100M Labels. In *ICML*. 2330–2340.
[13] Jia Deng, Sanjeev Satheesh, Alexander C. Berg, and Fei-Fei Li. 2011. Fast and Balanced: Efficient Label Tree Learning for Large Scale Object Recognition. In *NeurIPS*. 567–575.
[14] Charles Elkan and Keith Noto. 2008. Learning classifiers from only positive and unlabeled data. In *KDD*. 213–220.
[15] Alberto Fernández, Salvador García, Mikel Galar, Ronaldo C Prati, Bartosz Krawczyk, and Francisco Herrera. 2018. *Learning from imbalanced data sets*. Springer.
[16] Chuan Guo, Ali Mousavi, Xiang Wu, Daniel N Holtmann-Rice, Satyen Kale, Sashank Reddi, and Sanjiv Kumar. 2019. Breaking the Glass Ceiling for Embedding-Based Classifiers for Large Output Spaces. In *NeurIPS*.
[17] Himanshu Jain, Venkatesh Balasubramanian, Bhanu Chunduri, and Manik Varma. 2019. Slice: Scalable Linear Extreme Classifiers Trained on 100 Million Labels for Related Searches. In *WSDM*. 528–536.
[18] Himanshu Jain, Yashoteja Prabhu, and Manik Varma. 2016. Extreme Multi-Label Loss Functions for Recommendation, Tagging, Ranking and Other Missing Label Applications. In *KDD*. 935–944.
[19] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2018. Unbiased Learning-to-Rank with Biased Feedback. In *IJCAI*. 5284–5288.
[20] Sujay Khandagale, Han Xiao, and Rohit Babbar. 2020. Bonsai: diverse and shallow trees for extreme multi-label classification. *Machine Learning* 109, 11 (2020), 2099–2119.
[21] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring Networks of Substitutable and Complementary Products. In *KDD*. 785–794.
[22] Tharun Kumar Reddy Medini, Qixuan Huang, Yiqiu Wang, Vijai Mohan, and Anshumali Shrivastava. 2019. Extreme Classification in Log Memory using Count-Min Sketch: A Case Study of Amazon Search with 50M Products. In *NeurIPS*. 13265–13275.
[23] Jorge J. Moré. 1978. The Levenberg-Marquardt algorithm: Implementation and theory. In *Numerical Analysis*, G. A. Watson (Ed.). Springer Berlin Heidelberg, 105–116.
[24] Nagarajan Natarajan, Inderjit S. Dhillon, Pradeep Ravikumar, and Ambuj Tewari. 2017. Cost-sensitive learning with noisy labels. *Journal of Machine Learning Research* 18, 1 (2017), 5666–5698.
[25] Yashoteja Prabhu and Manik Varma. 2014. FastXML: a fast, accurate and stable tree-classifier for extreme multi-label learning. In *KDD*. 263–272.
[26] Mohammadreza Qaraei, Erik Schultheis, Priyanshu Gupta, and Rohit Babbar. 2021. Convex Surrogates for Unbiased Loss Functions in Extreme Classification With Missing Labels. In *WWW*. 3711–3720.
[27] Filip Radlinski, Robert Kleinberg, and Thorsten Joachims. 2008. Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th international conference on Machine learning*. 784–791.
[28] Francis J. Richards. 1959. A Flexible Growth Function for Empirical Use. *Journal of Experimental Botany* 10, 2 (06 1959), 290–301.
[29] Yuta Saito, Suguru Yaginuma, Yuta Nishino, Hayato Sakata, and Kazuhide Nakata. 2020. Unbiased Recommender Learning from Missing-Not-At-Random Implicit Feedback. In *WSDM*. 501–509.
[30] Erik Schultheis and Rohit Babbar. 2021. Unbiased Loss Functions for Multilabel Classification with Missing Labels. *CoRR* abs/2109.11282 (2021).
[31] Shashank Singh and Justin Khim. 2021. Statistical Theory for Imbalanced Binary Classification. arXiv:2107.01777 [math.ST]
[32] Yukihiro Tagami. 2017. AnnexML: Approximate Nearest Neighbor Search for Extreme Multi-label Classification. In *KDD*. 455–464.
[33] Paweł Teisseyre, Jan Mielniczuk, and Małgorzata Łazęcka. 2020. Different Strategies of Fitting Logistic Regression for Positive and Unlabelled Data. In *ICCS*. 3–17.
[34] Jimena Tomás, Newton Spolaôr, Everton Cherman, and Maria-Carolina Monard. 2014. A Framework to Generate Synthetic Multi-label Datasets. *Electronic Notes in Theoretical Computer Science* 302 (02 2014), 155–176.
[35] Brendan Van Rooyen and Robert C. Williamson. 2017. A theory of learning with corrupted labels. *Journal of Machine Learning Research* 18, 1 (2017), 8501–8550.
[36] Tong Wei and Yu-Feng Li. 2020. Does Tail Label Help for Large-Scale Multi-Label Learning? *IEEE Transactions on Neural Networks and Learning Systems* 31, 7 (2020), 2315–2324.
[37] Marek Wydmuch, Kalina Jasinska-Kobus, Rohit Babbar, and Krzysztof Dembczynski. 2021. Propensity-Scored Probabilistic Label Trees. In *SIGIR*. 2252–2256.
[38] Longqi Yang, Yin Cui, Yuan Xuan, Chenyang Wang, Serge Belongie, and Deborah Estrin. 2018. Unbiased Offline Recommender Evaluation for Missing-Not-at-Random Implicit Feedback. In *RecSys*. 279–287.
[39] Ian En-Hsu Yen, Xiangru Huang, Wei Dai, Pradeep Ravikumar, Inderjit S. Dhillon, and Eric P. Xing. 2017. PPDsparse: A Parallel Primal-Dual Sparse Method for Extreme Classification. In *KDD*. 545–553.
[40] Ronghui You, Zihan Zhang, Ziye Wang, Suyang Dai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2019. AttentionXML: Label Tree-based Attention-Aware Deep Model for High-Performance Extreme Multi-Label Text Classification. In *NeurIPS*. 5812–5822.
[41] Ziwei Zhu, Yun He, Yin Zhang, and James Caverlee. 2020. Unbiased Implicit Recommendation and Propensity Estimation via Combinational Joint Learning. In *RecSys*. 551–556.
[42] Jingwei Zhuo, Ziru Xu, Wei Dai, Han Zhu, Han Li, Jian Xu, and Kun Gai. 2020. Learning Optimal Tree Models under Beam Search. In *ICML*. 11650–11659.

A LABEL FREQUENCY IN XMLC DATASETS

We show in Figure 3 a log-log plot of the distribution of label frequencies in popular benchmark datasets from the XMLC repository [7]. As also noted in other works [2, 8], the label frequencies are characterized by the long-tail.

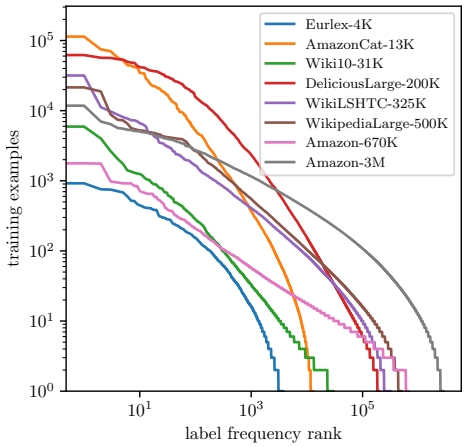

Figure 3: Label frequency in XMLC datasets. The X-axis shows the label rank when sorted by the frequency of positive instances and the Y-axis gives the number of the positive instances.

B STATISTICS OF MISSING-LABEL DATASETS

We present in Table 6 the statistics of datasets created to experiment with propensities models and missing labels. The description of these datasets is given in Section 4. Because the process of generating the biased training sets contains randomness, for the mean number of labels per example, we report the average value from all generated variants of the datasets.

Table 6: Characteristics of datasets used in the experiments. We report the size of the *biased* train set ($n^{\text{tr.}}$) and the size of the test set ($n^{\text{ts.}}$), the total number of labels (m), and the mean number of labels per example in the *biased* train set and the test set. Symbol $*$ denotes the average value over all generated variants of the dataset, and † the value corrected by $p_j^c = r/m$, where r is a number of labels sampled for labeling for each datapoint.

Dataset	$n^{\text{tr.}}$	$n^{\text{ts.}}$	m	$\mathbb{E}[\lvert\mathcal{L}(x)\rvert]^{\text{tr.}}$	$\mathbb{E}[\lvert\mathcal{L}(x)\rvert]^{\text{ts.}}$
Artificial-ϕ_{JPV}	63,000	30,000	100	$\approx 3.72^*$	4.27
Artificial-ϕ_{P}	63,000	30,000	100	$\approx 2.44^*$	4.27
Eurlex-ϕ_{JPV}	12,188	5,805	976	$\approx 1.55^*$	4.46
Eurlex-ϕ_{P}	12,188	5,805	976	$\approx 0.81^*$	4.46
AmazonCat-13K-ϕ_{JPV}	$6.28 \cdot 10^5$	$2.99 \cdot 10^5$	1,531	$\approx 3.65^*$	4.43
AmazonCat-13K-ϕ_{P}	$6.28 \cdot 10^5$	$2.99 \cdot 10^5$	1,331	$\approx 0.69^*$	4.43
Wiki10-31K-ϕ_{JPV}	13,079	6,229	2,951	$\approx 6.22^*$	13.38
Wiki10-31K-ϕ_{P}	13,079	6,229	2,951	$\approx 1.46^*$	13.38
Yahoo-R3	11,946	2,436	1,000	$\approx 3.97^*$	$87.94^\dagger$

C ESTIMATION ERRORS OF PROPENSITY MODELS

Table 7 presents the mean squared errors (MSE) of inverse propensity estimates ($\|\hat{p}_j^{-1} - \phi(\tilde{\pi}_j)^{-1}\|^2$) on the Yahoo R3 dataset obtained by different propensity models described in Section 4. Models marked as (fit.) have been fitted to the same data the error has been calculated on. Since we repeated the experiment with the PEJL model several times, we report the average error.

Table 7: MSE on inverse propensity estimates of different propensity models for the Yahoo-R3 dataset. Symbol $*$ denotes the average value over several runs.

Model	MSE
$\phi_{\mathbf{1}}$	1663.94
ϕ_{JPV}	1557.04
ϕ_{JPV} (fit.)	1259.78
ϕ_{P} (fit.)	512.94
ϕ_{R} (fit.)	516.85
ϕ_{PEJL}	$\approx 1236^*$

D DETAILS OF MODEL TRAINING

We describe here the implementation and training procedures of classifiers used in the experiments in Section 4. For all the experiments, except the one with the PEJL model described in Section 4.3, we train a linear model using the unbiased binary-cross entropy loss (17) with propensities coming from different models ϕ. The weights of the linear models are initialized from the uniform distribution $\mathcal{U}(-\sqrt{k}, \sqrt{k})$, where $k = 1/d$ with d being the number of features.

In the case of the PEJL approach, the same linear architecture is used but trained using (18). To assure that estimated values of p_j are in $[0, 1]$, they are modeled as a sigmoid transformation ($\sigma(\cdot)$) of p'_j parameters (one per label). Parameters p'_j are initialized from the uniform distribution $\mathcal{U}(-e, e)$.

For each experiment, 10% of the biased training set serves as a validation set for the selection of hyperparameters and early stopping. All methods are implemented using PyTorch [44]. Optimization is performed with the Adam optimizer [43]. Only two hyperparameters are tuned on the biased validation test, the learning rate (from set {0.005, 0.01, 0.05, 0.1}) and the weight decay (from set {0, 1e-8, 1e-7, 1e-6}). Experiments have been performed on a single machine with Intel Xeon Gold 5115 and NVIDIA V100 16GB.

E AN ALTERNATIVE PEJL METHOD

Below we briefly describe the main idea behind the method of Zhu et al. [41], which can serve as an alternative to the approach of Teisseyre et al. [33] describe in Section 4.3.

Let $\boldsymbol{M} \in \{0, 1\}^m$ be a mask random variable that expresses relation between $\boldsymbol{Y}$ and $\tilde{\boldsymbol{Y}}$, i.e., $\tilde{\boldsymbol{Y}} = \boldsymbol{M} \odot \boldsymbol{Y}$. We then have $\mathbb{P}\left[\tilde{Y}_j = 1 | \boldsymbol{x}\right] = \mathbb{P}\left[M_j = 1\right] \cdot \mathbb{P}[Y = 1 | \boldsymbol{x}]$ and $\mathbb{P}\left[M_j = 1\right] = p_j$ and $\mathbb{P}[Y = 1 | \boldsymbol{x}] = \eta_j(\boldsymbol{x})$ [30]. Because M_j is a Bernoulli variable as well as Y_j and $\tilde{Y}_j$, we end up with two models, the first one for Y_j with known p_j, and the second for M_j with known $\eta_j(\boldsymbol{x})$. The first one can be obtained

by minimizing (17). For the second one, a parametric model $\phi(\theta_j)$ for p_j can be learned by minimizing the following logistic loss:

$$\ell(\tilde{y}_j, \eta_j(\boldsymbol{x}), \phi(\theta_j)) = -\frac{\tilde{y}_j}{\eta_j(\boldsymbol{x})} \log(\phi(\theta_j)) - \left(1 - \frac{\tilde{y}_j}{\eta_j(\boldsymbol{x})}\right) \log(1-\phi(\theta_j)) \,. \tag{23}$$

We can learn $f_j(\boldsymbol{x})$ and $\phi(\theta_j)$ jointly by replacing $\eta_j(\boldsymbol{x})$ by $f_j(\boldsymbol{x})$ in (23), and p_j by $\phi(\theta_j)$ in (17). Since both $f_j(\boldsymbol{x})$ and $\phi(\theta_j)$ can be updated on a single example $\boldsymbol{x}$, to avoid estimation-training overlap problem, the training data is split into two parts, and the training is performed with $\phi(\theta_j)$ fixed on one part and with $f_j(\boldsymbol{x})$ fixed on the second one.

F ADDITIONAL ASSUMPTIONS FOR COMPLEX METRICS

Here we show an example which demonstrates that for a non-decomposable metric, additional assumptions on the process of labels going missing are required.

Consider a setting with two class labels. Let the label-wise propensities for both labels be $p_1 = p_2 = 0.5$. The desired unbiased loss function of ℓ can be parametrized for each prediction $\hat{\boldsymbol{y}}$ by four real numbers, $v_{\tilde{\boldsymbol{y}}} \coloneqq \tilde{\ell}(\tilde{\boldsymbol{y}}, \hat{\boldsymbol{y}})$. We can consider two different scenarios. First, both labels always go missing at the same time, second they go missing complementarily, corresponding to the probability distributions $\mathbb{P}$ and $\mathbb{P}'$. As the unbiasedness needs to hold for all potential distributions of true labels, it needs to hold in particular in the four cases in which the true label distribution is concentrated on a single point. By explicitly calculating expectations through reading off the probabilities from Table 8, we can state the unbiasedness requirement which is $\mathbb{E}\left[\tilde{\ell}(\tilde{Y}, \hat{Y})\right] = \mathbb{E}\left[\ell(Y, \hat{Y})\right]$:

$$\begin{aligned}
\ell((1,1), \hat{\boldsymbol{y}}) &= 0.5(v_{11} + v_{00}) = 0.5(v_{10} + v_{01}) \,, \\
\ell((1,0), \hat{\boldsymbol{y}}) &= 0.5v_{00} + 0.5v_{10} \,, \\
\ell((0,1), \hat{\boldsymbol{y}}) &= 0.5v_{00} + 0.5v_{01} \,, \\
\ell((0,0), \hat{\boldsymbol{y}}) &= v_{00} \,.
\end{aligned} \tag{24}$$

These are five linear equations with only four variables, so in general, there is no solution. If we additionally assume that the labels go missing independently from each other, then the marginal propensities uniquely determine the full distribution. In that case, unbiased estimates can be derived for general loss functions [30].

Table 8: Different distributions with the same propensities.

Y_1	Y_2	$\tilde{Y}_1$	$\tilde{Y}_2$	$\mathbb{P}[\tilde{Y}_1, \tilde{Y}_2]$	$\mathbb{P}'[\tilde{Y}_1, \tilde{Y}_2]$	$\tilde{\ell}(\tilde{\boldsymbol{y}}, \hat{\boldsymbol{y}})$
1	1	1	1	0.5	0.0	v_{11}
1	1	1	0	0.0	0.5	v_{10}
1	1	0	1	0.0	0.5	v_{01}
1	1	0	0	0.5	0.0	v_{00}
1	0	1	0	0.5	0.5	v_{10}
1	0	0	0	0.5	0.5	v_{00}
0	1	0	1	0.5	0.5	v_{01}
0	1	0	0	0.5	0.5	v_{00}
0	0	0	0	1.0	1.0	v_{00}

APPENDIX REFERENCES

[43] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

[44] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *NeurIPS*. 8024–8035.