A General Online Algorithm for Optimizing Complex Performance Metrics

Wojciech Kotłowski [1] **Marek Wydmuch** [1] **Erik Schultheis** [2] **Rohit Babbar** [2 3] **Krzysztof Dembczyński** [1 4]

Abstract

We consider sequential maximization of performance metrics that are general functions of a confusion matrix of a classifier (such as precision, F-measure, or G-mean). Such metrics are, in general, non-decomposable over individual instances, making their optimization very challenging. While they have been extensively studied under different frameworks in the batch setting, their analysis in the online learning regime is very limited, with only a few distinguished exceptions. In this paper, we introduce and analyze a general online algorithm that can be used in a straightforward way with a variety of complex performance metrics in binary, multi-class, and multi-label classification problems. The algorithm's update and prediction rules are appealingly simple and computationally efficient without the need to store any past data. We show the algorithm attains $\mathcal{O}(\frac{\ln n}{n})$ regret for concave and smooth metrics and verify the efficiency of the proposed algorithm in empirical studies.

1. Introduction

Many modern applications of machine learning involve optimization of complex performance metrics that, unlike misclassification error, do not decompose into expectation over instance-wise quantities. Examples of such measures include F-measure (Lewis, 1995), the area under the ROC curve (AUC) (Drummond & Holte, 2005), geometric (Drummond & Holte, 2005; Wang & Yao, 2012; Menon et al., 2013; Cao et al., 2019) and harmonic mean (Kennedy et al., 2009), and Matthews coefficient (Baldi et al., 2000).

Complex performance metrics have been studied in binary (Ye et al., 2012; Koyejo et al., 2014; Busa-Fekete et al., 2015; Dembczynski et al., 2017; Singh & Khim, 2022), multi-class (Narasimhan et al., 2015b; 2022), and multi-label classification (Waegeman et al., 2014; Koyejo et al., 2015; Kotłowski & Dembczyński, 2017) under different theoretical frameworks. The population utility (PU) framework focuses on building a model being close to the optimal one on the population level (Ye et al., 2012). The expected test utility (ETU) framework concerns the optimization of the expected utility computed over a given (known) test set (Lewis, 1995; Jansche, 2007; Ye et al., 2012). These two frameworks are different in the formulation of the objectives (more precisely, they differ in the order in which the expectation and the metric is computed), but they turn out to be asymptotically equivalent (Dembczynski et al., 2017). Still, most of the existing works concern a batch setting.

In this paper, we consider an online setting for optimization of complex performance metrics. In our framework, an algorithm observes a sequence of instances, one at a time, which are drawn *i.i.d.* from some distribution. At each iteration, after receiving the input vector, the algorithm makes a prediction and observes the true label (or vector of labels). After observing the entire sequence of data, the algorithm is evaluated by means of a performance metric computed from its empirical confusion matrix over the data. We evaluate an algorithm in our setting by means of a *regret*, which is the expected (over the data sequence) difference between the algorithm's performance and that of the optimal classifier. The goal is to design no-regret algorithms, that is, algorithms that guarantee regret converges to zero as the sequence length grows.

Contrary to most of the online learning work, we do not assume that predictions of the algorithm are obtained by means of some parametric function of the input vector (e.g., linear or generalized linear models). Instead, we assume that the algorithm has access to a *conditional probability estimator* (CPE), which returns an estimate of the true label conditional distribution at the input vector, which the algorithm will use to issue its prediction. Such a setup would be trivial for decomposable performance metrics, where the optimal decision is fully determined by the label conditional probabilities. For non-decomposable performance metrics, however, the optimal decision on different instances are no longer independent, yet the online setting requires the classi-

[*]Equal contribution [1]Poznan University of Technology, Poznan, Poland [2]Aalto University, Helsinki, Finland [3]University of Bath, Bath, UK [4]Yahoo Research, New York, United States. Correspondence to: Wojciech Kotłowski <wkotlowski@cs.put.poznan.pl>, Marek Wydmuch <mwydmuch@cs.put.poznan.pl>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

fier to commit to a prediction as soon as it gets the instance, without any chance to revise decisions that turn out to be suboptimal in retrospect.

In this paper, we introduce and analyze a general online algorithm called Online Metric Maximization Algorithm (OMMA), which can be used in a straightforward way with a variety of complex performance metrics in binary, multi-class, and multi-label classification problems, with or without budgeted predictions (i.e., requiring exactly k predictions per instance). The algorithm's prediction rule is appealingly simple: at any given trial, the algorithm issues the prediction that maximizes the expected value (with respect to the current, not yet seen label, obtained by means of the CPE) of a *linearized* version of the metric. Thus, OMMA is effectively a cost-sensitive classification rule with costs determined by the gradient of the utility at the current confusion matrix, and computing its prediction boils down to thresholding or sorting linear functions of the conditional probabilities. The algorithm is computationally efficient and does not need to store any past data or predictions beyond the entries of the confusion matrix, update of which can be done online in a straightforward manner. The algorithm attains $\mathcal{O}(\frac{\ln n}{n})$ regret for concave and smooth metrics (up to the estimation error of its CPE), and $\mathcal{O}(\sqrt{\frac{\ln n}{n}})$ regret without smoothness.

We also verify the efficiency of the proposed algorithm in empirical studies, where we evaluate it on a wide range of multi-label benchmarks and performance measures. Our results show that the proposed algorithms achieve attractive performance in comparison to similar algorithms that can only be applied to a limited set of measures or require more computations and memory.

Our contribution can be summarized as follows:

- We formulate an online learning framework in which the classifier itself is not parametric but based on a potentially learnable label probability estimator,
- We provide a simple algorithm with a constant memory footprint that can optimize general non-decomposable performance metrics in an online manner across a wide range of learning tasks, including binary, multi-class, multi-label, and budgeted-at-k predictions,
- We prove regret bounds for this algorithm, showing that for concave and smooth metrics, we converge to the population-optimal at a rate of $\mathcal{O}(\frac{\ln n}{n})$.
- We show that in data sets with large variance, it can be beneficial for the decision process to *ignore* the available label feedback, and instead rely entirely on the biased, but much more stable, CPE instead.

Related work. Despite an extensive and long line of research on complex performance metrics in the batch setting, the analysis of these metrics in the online setting is quite limited, with only a few distinguished exceptions. Below we briefly describe the main results obtained so far.

The online problem has been tackled by optimizing a surrogate loss/reward function convex with respect to model parameters, which are updated in an online fashion. Kar et al. (2014) proposed an online gradient descent method for structural surrogates (Joachims, 2005). Narasimhan et al. (2015a) considered optimizing metrics being functions of true positive and true negative rate, which are then replaced by convex surrogate rewards. Such approaches may have high memory cost and do not converge to an optimal model for a given metric, but to a model optimal for the structural surrogate, and these two do not necessarily coincide (within the considered parametric class of functions).

The setup closest to ours has been considered in the context of optimizing the F-measure, where it is known that the optimal classifier is obtained by thresholding label conditional probabilities. In particular, Busa-Fekete et al. (2015) have introduced an online algorithm to simultaneously train a CPE model and tune the threshold. They have shown that the algorithm is asymptotically consistent, but the convergence rate has not been established. Interestingly, this algorithm can be seen as a special case of our general algorithm. In a follow-up paper, Liu et al. (2018) have designed an algorithm in which the threshold is updated using stochastic gradient descent of a strongly convex function, which reflects a specific property of the optimal threshold. The algorithm has a convergence rate of $\mathcal{O}(\frac{\ln n}{\sqrt{n}})$.

For a wider class of functions, one can directly exploit the connection to cost-sensitive classification. Yan et al. (2017) has considered simultaneously learning multiple classifiers with different cost vectors together with an online algorithm that learns to select the best one. An alternative approach is to devise a method that finds the optimal costs in the space of confusion matrices, for example, by adapting the Frank-Wolfe algorithm previously used in batch learning for smooth and concave metrics (Narasimhan et al., 2015b; 2022; Schultheis et al., 2024). Such an approach can be then naïvely applied to the online setting by running it in exponentially growing time intervals to amortize the cost of rerunning the algorithm on the entire batch. We use such an approach as a baseline in our experimental studies.

Let us notice that our contribution does not coincide with any of the online/stochastic Frank-Wolfe methods known in the literature (Hazan & Kale, 2012; Reddi et al., 2016), as these concern minimization of an objective that is either a finite sum or an expectation, and thus linearly decomposes between the trials. In our case, the objective is a global, non-linear function of all past predictions and labels.

Online algorithms and regret minimization have been extensively studied in the online convex optimization (OCO) framework (Cesa-Bianchi & Lugosi, 2006; Hazan, 2016; Shalev-Shwartz, 2012). Our setup, however, substantially differs from OCO. On the one hand, we aim at optimizing a more difficult, non-linear function of the entire sequence of prediction, and we need stochastic (i.i.d.) assumption on the data sequence, as otherwise, no algorithm can guarantee vanishing regret (Appendix C). On the other hand, our algorithm is equipped with the CPE, which gives an approximate distribution of labels at the moment of prediction.

2. Problem Setup

Let $\boldsymbol{x} \in \mathcal{X}$ be the input instance and $\boldsymbol{y} \in \mathcal{Y} \subseteq \{0,1\}^m$ the output label vector, jointly distributed according to $(\boldsymbol{x}, \boldsymbol{y}) \sim \mathbb{P}$. We let $\mathbb{P}_{\mathcal{X}}$ denote the marginal distribution $\boldsymbol{x}$, and $\boldsymbol{\eta}(\boldsymbol{x})$ the label conditional probability, $\boldsymbol{\eta}(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{y}|\boldsymbol{x}}[\boldsymbol{y}]$. In *multi-class classification*, the label vector $\boldsymbol{y}$ is a one-hot encoding of one out of m classes, $\mathcal{Y} = \{\boldsymbol{y} \in \{0,1\}^m \colon \sum_j y_j = 1\}$, while $\boldsymbol{\eta}(\boldsymbol{x}) = (\eta_1(\boldsymbol{x}), \ldots, \eta_m(\boldsymbol{x}))$ is the label conditional distribution with $\eta_j(\boldsymbol{x}) = \mathbb{P}(y_j = 1|\boldsymbol{x})$ denoting the conditional probability of class j. Whereas in *multi-label classification* $\boldsymbol{y} \in \{0,1\}^m$ denotes a vector of relevant labels and $\boldsymbol{\eta}(\boldsymbol{x}) = (\eta_1(\boldsymbol{x}), \ldots, \eta_m(\boldsymbol{x}))$ is vector of *label marginal conditional probabilities*, $\eta_j(\boldsymbol{x}) = \mathbb{P}(y_j = 1|\boldsymbol{x})$. Note that in the multi-label case, $\boldsymbol{\eta}(\boldsymbol{x})$ *is not* the conditional distribution $\mathbb{P}(\boldsymbol{y}|\boldsymbol{x})$ (which is a *joint* distribution over $\{0,1\}^m$), but in the context of this paper, it will always suffice to only operate on its marginals; we will thus sometimes use $\boldsymbol{y} \sim \boldsymbol{\eta}(\boldsymbol{x})$ to denote a multi-label vector drawn from $\mathbb{P}(\boldsymbol{y}|\boldsymbol{x})$ with marginals $\boldsymbol{\eta}(\boldsymbol{x})$ if no other property of the distribution matters, and we will call $\boldsymbol{\eta}$ the conditional distribution to simplify the presentation.

We consider an online (stochastic) setting in which the algorithm observes a sequence of instances $(\boldsymbol{x}_1, \boldsymbol{y}_1), (\boldsymbol{x}_2, \boldsymbol{y}_2), \ldots, (\boldsymbol{x}_n, \boldsymbol{y}_n)$, drawn *i.i.d.* from $\mathbb{P}$. At each iteration $t = 1, 2, \ldots, n$, after receiving input $\boldsymbol{x}_t$ drawn from $\mathbb{P}_{\mathcal{X}}$, the algorithm makes a prediction $\widehat{\boldsymbol{y}}_t$, and next observes the true output $\boldsymbol{y}_t$ drawn from $\mathbb{P}(\boldsymbol{y}|\boldsymbol{x}_t)$. We assume that the algorithm has access to a *conditional probability estimator* (CPE) $\widehat{\boldsymbol{\eta}}_t$, which at trial t returns an estimate $\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)$ of the true conditional distribution $\boldsymbol{\eta}(\boldsymbol{x}_t)$, which the algorithm will use for issuing its prediction $\widehat{\boldsymbol{y}}_t$. We put the time subscript in $\widehat{\boldsymbol{\eta}}_t$ as we allow the CPE to change over time (e.g., it can be produced by an external online learner run on the same sequence of instances); we assume, however, that $\widehat{\boldsymbol{\eta}}_t$ can only depend on the observed data $\boldsymbol{x}_t$, $(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_{t-1}, \boldsymbol{y}_{t-1})$ (i.e., does not depend on $\boldsymbol{y}_t$ and all the future instance). The CPE could be, for instance, a neural network or boosted decision trees trained (possibly online) by minimizing some proper scoring loss. In this work, we do not focus on the way the CPE is learned, and all our bounds on the performance of the algorithm will depend on the *estimation error* of $\widehat{\boldsymbol{\eta}}_t$ with respect to true conditional distribution $\boldsymbol{\eta}$. We outline the online protocol in Figure 1.

for $t = 1, \ldots, n$ **do**
  Observe input instance $\boldsymbol{x}_t$ drawn from $\mathbb{P}_{\mathcal{X}}$
  Receive conditional probability estimate $\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)$
  Predict label $\widehat{\boldsymbol{y}}_t$ based on $\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)$
  Receive true label $\boldsymbol{y}_t$ drawn from $\mathbb{P}(\cdot|\boldsymbol{x}_t)$
Evaluate based on $\psi(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n))$

Figure 1. The online protocol.

Given a sequence of labels[1] $\boldsymbol{y}^n = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)$ and a sequence of algorithm's predictions $\widehat{\boldsymbol{y}}^n = (\widehat{\boldsymbol{y}}_1, \ldots, \widehat{\boldsymbol{y}}_n)$, we let $\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)$ denote the (empirical) *confusion matrix*. In multi-class classification, it is an $m \times m$ matrix $\boldsymbol{C}$ with entries defined as $C_{j\ell} = \frac{1}{n}\sum_{t=1}^n y_{tj}\widehat{y}_{t\ell}$, where y_{tj} is the j-th entry of vector $\boldsymbol{y}_t$. Thus, $C_{j\ell}$ contains the fraction of times an instance from class j was predicted as being in class ℓ. Note that for binary classification ($m = 2$), we get the usual true-positive (C_{11}), false-negative (C_{10}), false-positive (C_{01}), true-negative (C_{00}) entries, where we index the entries from 0 to stick to a more convenient notation. In turn, in multi-label classification, the confusion matrix is an $m \times 2 \times 2$ tensor $\boldsymbol{C}$, where $C_{juv} = \frac{1}{n}\sum_{t=1}^n \mathbf{1}\{y_{tj} = u\}\mathbf{1}\{\widehat{y}_{tj} = v\}$,[2] where $j \in \{1, \ldots, m\}$ and $u, v \in \{0,1\}$. In other words, $\boldsymbol{C}$ is a sequence of m binary confusion matrices, separately for each label. In a given problem, we let $\mathcal{C}$ denote the set of all achievable confusion matrices, that is the set of confusion matrices that can be formed from allowed label and prediction vectors $\boldsymbol{y}^t$, $\widehat{\boldsymbol{y}}^t$ for any t. Note that, independent of the considered problem, the confusion matrix can always be written as $\boldsymbol{C}(\boldsymbol{y}^t, \widehat{\boldsymbol{y}}^t) = \frac{1}{t}\sum_{i=1}^t \boldsymbol{C}(\boldsymbol{y}_i, \widehat{\boldsymbol{y}}_i)$ (with $\boldsymbol{C}(\boldsymbol{y}_i, \widehat{\boldsymbol{y}}_i)$ being confusion matrices computed out of a single label $\boldsymbol{y}_i$ and prediction $\widehat{\boldsymbol{y}}_i$), which gives a simple online update:

$$\boldsymbol{C}(\boldsymbol{y}^t, \widehat{\boldsymbol{y}}^t) = \frac{t-1}{t}\boldsymbol{C}(\boldsymbol{y}^{t-1}, \widehat{\boldsymbol{y}}^{t-1}) + \frac{1}{t}\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)\,. \quad (1)$$

In this work, we focus on online maximization of performance metrics that do not decompose into a sum over instances but are general functions of the confusion matrix of the algorithm. Specifically, after observing the entire sequence of n instances, the algorithm is evaluated by means of a *utility metric* $\psi = \psi(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n))$. For binary classification, examples of such measures include the F-measure, geometric and harmonic mean, area under the ROC curve, recall, precision, etc. We present their definitions in Table 1. Most of these metrics have

[1] Generally, by $\boldsymbol{v}^t$ we denote a sequence $(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_t)$; $f(\boldsymbol{v}^t)$ denotes $(f(\boldsymbol{v}_1), \ldots, f(\boldsymbol{v}_t))$.

[2] $\mathbf{1}(S)$ is the indicator function equal one when S holds.

Table 1. Examples of binary and multiclass confusion matrix measures. Binary measures are expressed in terms of true-positives (tp $= C_{11}$), false-negatives (fn $= C_{10}$), false-positives (fp $= C_{01}$), and true-negatives (tn $= C_{00}$).

Metric	$\psi(\boldsymbol{C}^{2\times 2})$	$\psi(\boldsymbol{C}^{m\times m})$	Metric	$\psi(\boldsymbol{C}^{2\times 2})$	$\psi(\boldsymbol{C}^{m\times m})$
Accuracy	$\mathrm{tp}+\mathrm{tn}$	$\sum_{i=1}^m C_{ii}$	G-mean	$\sqrt{\frac{\mathrm{tp}\cdot\mathrm{tn}}{(\mathrm{tp}+\mathrm{fn})(\mathrm{tn}+\mathrm{fp})}}$	$\left(\prod_{j=1}^m \frac{C_{jj}}{\sum_{i=1}^m C_{ji}}\right)^{1/m}$
Balanced Acc.	$\frac{\mathrm{tp}}{2(\mathrm{tp}+\mathrm{fn})}+\frac{\mathrm{tn}}{2(\mathrm{tn}+\mathrm{fp})}$	$\sum_{i=1}^m \frac{C_{ii}}{m\sum_{j=1}^m C_{ij}}$	H-mean	$2\left(\frac{\mathrm{tp}+\mathrm{fn}}{\mathrm{tp}}+\frac{\mathrm{tn}+\mathrm{fp}}{\mathrm{tn}}\right)^{-1}$	$m\left(\sum_{j=1}^m \frac{\sum_{i=1}^m C_{ji}}{C_{jj}}\right)^{-1}$
Recall	$\frac{\mathrm{tp}}{\mathrm{tp}+\mathrm{fn}}$	micro- or macro-avg.	Q-mean	$1-\sqrt{\frac{1}{2}\left(\left(\frac{\mathrm{fn}}{\mathrm{tp}+\mathrm{fn}}\right)^2+\left(\frac{\mathrm{fp}}{\mathrm{tn}+\mathrm{fp}}\right)^2\right)}$	$1-\sqrt{\frac{1}{m}\sum_{j=1}^m\left(1-\frac{C_{jj}}{\sum_{i=1}^m C_{ji}}\right)^2}$
Precision	$\frac{\mathrm{tp}}{\mathrm{tp}+\mathrm{fp}}$	micro- or macro-avg.	Jaccard	$\frac{\mathrm{tp}}{\mathrm{tp}+\mathrm{fp}+\mathrm{fn}}$	micro- or macro-avg.
F_β-measure	$\frac{(1+\beta^2)\mathrm{tp}}{(1+\beta^2)\mathrm{tp}+\beta^2\mathrm{fn}+\mathrm{fp}}$	micro- or macro-avg.	Matthews coeff.	$\frac{\mathrm{tp}\cdot\mathrm{tn}-\mathrm{fp}\cdot\mathrm{fn}}{\sqrt{(\mathrm{tp}+\mathrm{fp})(\mathrm{tp}+\mathrm{fn})(\mathrm{tn}+\mathrm{fp})(\mathrm{tn}+\mathrm{fn})}}$	micro- or macro-avg.

multi-class (some also presented in Table 1) as well as multi-label extensions in the micro- and macro-averaged variants. By rewriting the multi-class $\boldsymbol{C}^{m\times m}$ confusion matrix into a multi-label confusion tensor $\boldsymbol{C}^{m\times 2\times 2}$,

$$\boldsymbol{C}^{m\times 2\times 2}(\boldsymbol{C}^{m\times m}) := \left[\begin{pmatrix} C_{jj} & \sum_{i\neq j} C_{ij} \\ \sum_{i\neq j} C_{ji} & \sum_{i\neq j,\ell\neq j} C_{i\ell}\end{pmatrix}\right]_{j=1}^m ,$$

we can define the micro- and macro-averaged metrics for both multi-class and multi-label classification as:

$$\text{Micro-}\psi(\boldsymbol{C}^{m\times 2\times 2}) := \psi\left(\frac{1}{m}\sum_{j=1}^m \boldsymbol{C}_j\right),$$

$$\text{Macro-}\psi(\boldsymbol{C}^{m\times 2\times 2}) := \frac{1}{m}\sum_{j=1}^m \psi\left(\boldsymbol{C}_j\right) .$$

All the metrics presented above can be considered under the standard as well as budgeted at k variant (Schultheis et al., 2023; 2024), where the classifier is required to predict exactly k classes per instance ($\widehat{\boldsymbol{y}} \in \widehat{\mathcal{Y}}_k = \{\widehat{\boldsymbol{y}} \in \{0,1\}^m : \sum_j \widehat{y}_j = k\}$). The main difficulty in maximizing ψ comes from the fact that the algorithm is only evaluated at the end of the sequence by a (possibly complex) function of all the labels and predictions, while each prediction $\widehat{\boldsymbol{y}}_t$ must be made immediately upon observing $\boldsymbol{x}_t$, and cannot be changed in the future.

Regret. Since without imposing assumptions on the data distribution $\mathbb{P}$ one cannot meaningfully bound the ψ-accuracy of the algorithm in the absolute terms, the goal is to compare the algorithm's performance *relative* to that of the optimal predictor. Note, however, that defining the optimal predictions $\boldsymbol{y}^{\star n}$ simply as those maximizing the value of the utility, $\boldsymbol{y}^{\star n} = \operatorname{argmax}_{\widehat{\boldsymbol{y}}^n} \psi(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n))$, leads (for essentially all reasonable utilities) to a trivial solution $\boldsymbol{y}^{\star n} = \boldsymbol{y}^n$, which gives the maximum possible value ψ, and is thus not achievable by any algorithm, no matter how large n is. Thus, we proceed differently, defining a *classifier* $\boldsymbol{h}\colon \mathcal{X} \to \mathcal{Y}$ to be a function from the inputs to the outputs, and the optimal predictions $\boldsymbol{y}^{\star n} = \boldsymbol{h}^\star(\boldsymbol{x}^n)$ to be those generated by a classifier $\boldsymbol{h}^\star$ which maximizes *the expected (with respect to the data sequence) value of the utility*:[3]

$$\boldsymbol{h}^\star := \operatorname*{argmax}_{\boldsymbol{h}} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})^n}\left[\psi(\boldsymbol{C}(\boldsymbol{y}^n, \boldsymbol{h}(\boldsymbol{x}^n)))\right], \qquad (2)$$

where the maximization is with respect to all (measurable) classifiers. We define the (expected) *regret* of the algorithm as the difference between its expected performance in terms of ψ, and the performance of the optimal classifier $\boldsymbol{h}^\star$:

$$R_n := \mathbb{E}\left[\psi(\boldsymbol{C}(\boldsymbol{y}^n, \boldsymbol{h}^\star(\boldsymbol{x}^n)))\right] - \mathbb{E}\left[\psi(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n))\right].$$

Our goal is to design no-regret algorithms, that is, algorithms that guarantee $R_n \to 0$ as $n \to \infty$. In the next section, we propose a computationally- and memory-efficient algorithm, which guarantees (under certain assumptions on the utility) $R_n = \mathcal{O}\left(\frac{\ln n}{n}\right)$ up to the estimation error of CPE.

Note that when $n \to \infty$, $\boldsymbol{h}^\star$ coincides with the ψ-optimal population-level classifier. Approximating $\boldsymbol{h}^\star$ in the batch setting is a task which was tackled in a series of past works spanning binary classification (Ye et al., 2012; Menon et al., 2013; Koyejo et al., 2014; Narasimhan et al., 2014), multi-class classification (Narasimhan et al., 2015b; 2022), and multi-label classification (Waegeman et al., 2014; Koyejo et al., 2015; Kotłowski & Dembczyński, 2017; Schultheis et al., 2024), and boils down to, under certain mild assumptions on the data distribution, finding the optimal threshold on the conditional probability $\boldsymbol{\eta}(\boldsymbol{x})$ (for binary classification), or finding the optimal ensemble of cost-sensitive classifiers with an iterative convex optimization algorithm, such as the Frank-Wolfe method or gradient descent (for multi-class and multi-label classification).[4] In this light, the results of our paper can be interpreted as proposing a simple and

[3] Recall that $\boldsymbol{h}(\boldsymbol{x}^n)$ stands for $(\boldsymbol{h}(\boldsymbol{x}_1), \ldots, \boldsymbol{h}(\boldsymbol{x}_n))$.

[4] In the past papers, the optimal classifier is allowed to be

Algorithm 1 Online Measure Maximization Algorithm

Initialization: confusion matrix $\boldsymbol{C}_0$
for $t = 1, \ldots, n$ **do**
  Receive input $\boldsymbol{x}_t$ and probability estimate $\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)$
  Predict $\widehat{\boldsymbol{y}}_t = \operatorname{argmax}_{\widehat{\boldsymbol{y}}} \nabla\psi(\boldsymbol{C}_{t-1}) \cdot \left(\mathbb{E}_{\boldsymbol{y}_t \sim \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)}[\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}})]\right)$
  Receive label $\boldsymbol{y}_t$ and update $\boldsymbol{C}_t = \frac{t-1}{t}\boldsymbol{C}_{t-1} + \frac{1}{t}\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)$

efficient iterative algorithm, which does only a single pass over the data while achieving performance that is very close to that of the aforementioned batch optimization methods.

Remark 2.1 (Adversarial sequence of inputs). One might consider a setup in which the input instances $\boldsymbol{x}^n$ are not drawn i.i.d. from a fixed distribution, but rather form an arbitrary (possibly even adversarial) sequence, and the goal is to analyze the worst case with respect to $\boldsymbol{x}^n$ (note that the labels remain stochastic, being drawn from $\boldsymbol{\eta}(\boldsymbol{x})$). Unfortunately, this setting turns out to be too difficult in the worst case: in Appendix C, we show that there exists a conditional probability function $\boldsymbol{\eta}$ (which we even make known to the algorithm), and a sequence of inputs $\boldsymbol{x}^n$, on which no algorithm can have a vanishing regret. Thus, a stochastic data generation mechanism seems crucial for the existence of a no-regret learner.

3. The algorithm

In this section, we introduce our algorithm, called *Online Metric Maximization Algorithm (OMMA)*. To define the algorithm, we need to make the following assumption:

Assumption 3.1. The utility $\psi(\boldsymbol{C})$ is differentiable in $\boldsymbol{C}$.

This property is shared by all performance metrics commonly used in practice.[5] Let $\nabla\psi(\boldsymbol{C})$ be the matrix of derivatives of ψ with respect to $\boldsymbol{C}$, which is of the same shape as $\boldsymbol{C}$ ($m \times m$ for multi-class, and $m \times 2 \times 2$ for multi-label case). To simplify notation, we let $\boldsymbol{C}_t := \boldsymbol{C}(\boldsymbol{y}^t, \widehat{\boldsymbol{y}}^t)$ denote the confusion matrix of the algorithm after t trials. At trial t, given past labels $\boldsymbol{y}^{t-1}$, past predictions $\widehat{\boldsymbol{y}}^{t-1}$, new input $\boldsymbol{x}_t$ and the conditional probability estimate $\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)$, the OMMA algorithm predicts $\widehat{\boldsymbol{y}}_t$ which maximizes the expected (with respect to $\widehat{\boldsymbol{\eta}}_t$) *linearized* (at $\boldsymbol{C}_{t-1}$) version of the utility:[6]

$$\widehat{\boldsymbol{y}}_t = \operatorname*{argmax}_{\widehat{\boldsymbol{y}}} \nabla\psi(\boldsymbol{C}_{t-1}) \cdot \left(\mathbb{E}_{\boldsymbol{y}_t \sim \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)}\left[\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}})\right]\right) , \quad (3)$$

where the maximization is over the allowed predictions $\widehat{\boldsymbol{y}}$ (e.g., one-hot vectors in multi-class classification), and the operator "$\cdot$" denotes the matrix/tensor dot product (which is a standard vector dot product between vectorized matrix/tensor arguments). Note that the last element in (3) is a confusion matrix computed from *only a single observation* $\boldsymbol{y}_t$ *and a single prediction* $\widehat{\boldsymbol{y}}$. The algorithm does not need to store any past data or predictions beyond the entries of the confusion matrix, as the update of matrix can be done online using (1). Due to the linearity of (3), OMMA is effectively a *cost-sensitive classification* rule with costs determined by the gradient of the utility at the current confusion matrix. In what follows, we will show that formula (3) leads to a very simple rule, boiling down to thresholding or sorting linear functions of $\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)$.

Motivation. OMMA can be derived from considering a greedy method, which, given past labels $\boldsymbol{y}^{t-1}$ and predictions $\widehat{\boldsymbol{y}}^{t-1}$, chooses its prediction $\widehat{\boldsymbol{y}}_t$ in order to maximize the expected utility *including* iteration t (where the expectation is with respect to label $\boldsymbol{y}_t$, which is unknown at the moment of making prediction), that is, to maximize $F(\widehat{\boldsymbol{y}}_t) = \mathbb{E}_{\boldsymbol{y}_t \sim \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)}\left[\psi(\boldsymbol{C}(\boldsymbol{y}^t, \widehat{\boldsymbol{y}}^t))\right]$. By using a Taylor-expansion of $F(\widehat{\boldsymbol{y}}_t)$ with respect to $\boldsymbol{y}_t$, it turns out that maximizing $F(\widehat{\boldsymbol{y}}_t)$ and maximizing (3) are equivalent up to $\mathcal{O}(1/t^2)$ (see Appendix A for details). At the same time, our algorithmic update (3) is based on a linear objective and, therefore, more straightforward to calculate, as well as easier to analyze. In the experiment section, we show that both updates indeed behave very similarly to each other, being essentially indistinguishable for larger values of n.

Example: binary classification. To simplify the presentation, we switch from one-hot vector notation to scalars and denote by $y_t \in \{0, 1\}$ a label at time t ($y_t = 0$ corresponds to $\boldsymbol{y}_t = (1, 0)$, and $y_t = 1$ to $\boldsymbol{y}_t = (0, 1)$); similarly $\widehat{y}_t \in \{0, 1\}$ denotes the prediction and $\eta(\boldsymbol{x}_t) = P(y_t = 1|\boldsymbol{x}_t)$. The confusion matrix $\boldsymbol{C}_{t-1} = \boldsymbol{C}(\boldsymbol{y}^{t-1}, \widehat{\boldsymbol{y}}^{t-1})$ consists of four entries

$$C_{t-1,j\ell} = \sum_{i \le t-1} \mathbf{1}\{y_i = j\}\mathbf{1}\{\widehat{y}_i = \ell\}, \quad j, \ell \in \{0, 1\} . \quad (4)$$

Let us abbreviate $\frac{\partial}{\partial C_{j\ell}}\psi(\boldsymbol{C}_{t-1})$ as $\nabla_{j\ell}$, and $\widehat{\eta}_t(\boldsymbol{x}_t)$ as $\widehat{\eta}_t$. Since

$$\mathbb{E}_{y_t \sim \widehat{\eta}_t}[\boldsymbol{C}(y_t, \widehat{y})] = \begin{bmatrix} (1-\widehat{\eta}_t)(1-\widehat{y}) & (1-\widehat{\eta}_t)\widehat{y} \\ \widehat{\eta}_t(1-\widehat{y}) & \widehat{\eta}_t\widehat{y} \end{bmatrix} ,$$

equation (3) boils down to maximizing a cost-sensitive classification accuracy:

$$\nabla_{00}(1-\widehat{\eta}_t)(1-\widehat{y}) + \nabla_{01}(1-\widehat{\eta}_t)\widehat{y} + \nabla_{10}\widehat{\eta}_t(1-\widehat{y}) + \nabla_{11}\widehat{\eta}_t\widehat{y}, \quad (5)$$

which can also be rewritten, up to terms independent of $\widehat{y}$, as $\widehat{y}(\alpha\widehat{\eta}_t - \beta)$, with

$$\alpha = \nabla_{11} + \nabla_{00} - \nabla_{01} - \nabla_{10}, \quad \beta = \nabla_{00} - \nabla_{01},$$

randomized, that is to take values in the convex hull of $\mathcal{Y}$. All of our analysis effortlessly extends to such a setup, but we do not consider it for the clarity of the presentation. Moreover, our algorithm never needs to randomize its predictions.

[5] In case of non-differentiable and concave metrics, one can still use a supergradient in place of gradient.

[6] The algorithm is well-defined, as $\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}})$ is *linear* in labels and thus its expectation only depends on marginals $\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)$.

a cost-sensitive prediction rule (Elkan, 2001; Natarajan et al., 2018). Since all utilities used in practice are non-decreasing with true positives and true negatives ($\nabla_{11}, \nabla_{00} \geq 0$), and non-increasing with false negatives and false positives ($\nabla_{01}, \nabla_{10} \leq 0$), we get $a \geq 0$, and thus maximizing $\widehat{y}(\alpha\widehat{\eta}_t - \beta)$ boils down to choosing $\widehat{y}_t = 1$ whenever the conditional probability $\widehat{\eta}_t(\boldsymbol{x}_t)$ exceeds a threshold β/α. It is well-known that, under mild assumptions on the input distribution $\mathbb{P}_{\mathcal{X}}$ and the metric ψ, thresholding $\eta(\boldsymbol{x})$ is the optimal classification rule for maximizing the population-level version of ψ (Koyejo et al., 2014; Narasimhan et al., 2014). Thus, OMMA mimics the optimal classification rule, with a threshold computed based on the empirical data.

Example: multi-label classification (with a budget). Here $\boldsymbol{y}_t$ and $\widehat{\boldsymbol{y}}_t$ are label and prediction vectors of length m, and $\boldsymbol{\eta}(\boldsymbol{x}_t)$ is the vector of marginal label probabilities. $\boldsymbol{C}_{t-1}$ is a sequence of binary confusion matrices $\boldsymbol{C}^j_{t-1}$, $j = 1, \ldots, m$. Abbreviating $\frac{\partial}{\partial C^j_{uv}}\psi(\boldsymbol{C}_{t-1})$ as ∇^j_{uv}, and $\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)$ as $\widehat{\boldsymbol{\eta}}_t$ allows us to write down the objective (3) as a direct extension of the one from binary classification (5), summed over m labels:

$$\sum_{j=1}^{m} \widehat{y}_j(\alpha_j\widehat{\eta}_{tj} - \beta_j) \tag{6}$$

with $\alpha_j = \nabla^j_{11} + \nabla^j_{00} - \nabla^j_{01} - \nabla^j_{10}$, $\beta_j = \nabla^j_{00} - \nabla^j_{01}$. Maximizing (6) amounts to setting $\widehat{y}_{tj} = 1$ whenever $\alpha_j\widehat{\eta}_{tj} - \beta_j \geq 0$, or $\widehat{\eta}_{tj} \geq \beta_j/\alpha_j$. In the budgeted-at-$k$ variant, where the algorithm must predict *exactly* k labels each time, the prediction amounts to sorting the labels by means of $\alpha_j\widehat{\eta}_{tj} - \beta_j$ in descending order, and setting the top k of them to 1. Interestingly, this is closely related to the optimal population-level prediction rule for the budgeted setting (Schultheis et al., 2024).

4. Theoretical analysis

In order to prove a bound on the regret of OMMA algorithm, we need to make additional assumptions on the utility ψ.

Assumption 4.1. $\psi(\boldsymbol{C})$ is differentiable, concave, L-Lipschitz, and M-smooth in $\boldsymbol{C}$, that is for any $\boldsymbol{C}_1, \boldsymbol{C}_2 \in \mathcal{C}$, $\psi(\boldsymbol{C}_1) \leq \psi(\boldsymbol{C}_2) + \nabla\psi(\boldsymbol{C}_2)^\top(\boldsymbol{C}_1 - \boldsymbol{C}_2)$, $|\psi(\boldsymbol{C}_1) - \psi(\boldsymbol{C}_2)\| \leq L\|\boldsymbol{C}_1 - \boldsymbol{C}_2\|$, and $\|\nabla\psi(\boldsymbol{C}_1) - \nabla\psi(\boldsymbol{C}_2)\| \leq M\|\boldsymbol{C}_1 - \boldsymbol{C}_2\|$, where $\|\cdot\|$ denotes the entrywise L_2-norm.

Remark 4.2. Our algorithm generally requires smoothness of the objective to converge. The assumption can be waived by running it on a *smoothed* version of the metric. This, however, comes at the price of a slower convergence rate $\mathcal{O}(\sqrt{\ln n/n})$. As non-smooth metrics are not commonly employed in machine learning, we relegate the discussion to Appendix B.2

Theorem 4.3. *Let Assumption 4.1 hold. Then the OMMA algorithm has its regret bounded by:*

$$R_n \leq \frac{Ma(1+\ln n)}{n} + \frac{2bL}{n}\sum_{t=1}^{n}\mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right]$$

where $a = 1, b = 1$ for multi-class, and $a = m, b = \sqrt{2}$ for multi-label classification.

The proof is given in Appendix B.1. The bound in Theorem 4.3 consists of two parts: (1) the first term of order $\mathcal{O}(\frac{\ln n}{n})$ can be interpreted as the regret of OMMA had it been equipped with the true conditional probability $\boldsymbol{\eta}$ (that is, the estimator is exact); (2) the estimation error of $\widehat{\boldsymbol{\eta}}_t$ averaged over trials $t = 1, \ldots, n$. If the estimation error converges to zero with $n \to \infty$, OMMA becomes a no-regret learning algorithm.

Remark 4.4. In the proof of Theorem 4.3, the Lipschitzness and the smoothness properties are invoked along the parameter path of the algorithm, $\{\boldsymbol{C}_t\}_{t=1}^n$, in order to control the progress in optimizing the utility. These properties might not necessarily hold globally (for every confusion matrix) for utilities given in Table 1, for instance when the observed labels lead to severe class imbalance. However, adding a small constant to the denominator in the definition of a utility stabilizes its values and ensures its global Lipschitzness and smoothness. This is also the approach we take in the experiments to keep our algorithm stable over the initial part of the data sequence.

An alternative approach is to use the fact that most utilities in Table 1 are Lipschitz and smooth when label frequencies in the confusion matrix are bounded away from zero. Taking into account that these properties are applied along the path of the algorithm, and using concentration inequalities on the label frequencies, we show in Appendix E that as long as the probabilities of labels $\mathbb{P}(y_j = 1)$ are bounded away from zero, the regret of OMMA converges at a rate of $\mathcal{O}(\sqrt{\ln n/n})$ with high probability. Motivated by this fact, we also propose to use a regularization technique that, additionally to using a small constant in the denominator of some metrics, also adds the small value λ to the initial entries of the confusion matrix. With the updates of the confusion matrix, this value diminishes, being $\frac{\lambda}{t}$ at iteration t of the algorithm. This simple technique turns out to be helpful for some metrics.

5. Alternative variants of the algorithm

Internal semi-empirical confusion matrix. In the experiment, we also use an alternative version of the OMMA algorithm called OMMA($\widehat{\eta}$), outlined in Algorithm 2. Note that in the algorithm's description we use $\boldsymbol{C}(\widehat{\boldsymbol{\eta}}_t, \widehat{\boldsymbol{y}}_t) = \mathbb{E}_{\boldsymbol{y}_t \sim \widehat{\boldsymbol{\eta}}_t}\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)$ which follows from the fact that the confusion matrix is linear in labels and that $\mathbb{E}_{\boldsymbol{y}_t \sim \widehat{\boldsymbol{\eta}}_t}[\boldsymbol{y}_t] = \widehat{\boldsymbol{\eta}}_t$.

Algorithm 2 OMMA($\widehat{\eta}$)

Initialization: confusion matrix $\boldsymbol{C}_0$
for $t = 1, \ldots, n$ **do**
  Receive input $\boldsymbol{x}_t$ and probability estimate $\widehat{\boldsymbol{\eta}}_t := \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)$
  Predict $\widehat{\boldsymbol{y}}_t = \operatorname{argmax}_{\widehat{\boldsymbol{y}}} \nabla\psi(\boldsymbol{C}_{t-1}) \cdot \boldsymbol{C}(\widehat{\boldsymbol{\eta}}_t, \widehat{\boldsymbol{y}})$
  Update $\boldsymbol{C}_t = \frac{t-1}{t}\boldsymbol{C}_{t-1} + \frac{1}{t}\boldsymbol{C}(\widehat{\boldsymbol{\eta}}_t, \widehat{\boldsymbol{y}}_t)$

The only difference between OMMA and its modification is that OMMA($\widehat{\eta}$) updates its running confusion matrix $\boldsymbol{C}_t$ by means of $\widehat{\boldsymbol{\eta}}_t$ instead of the true label $\boldsymbol{y}_t$. In fact, OMMA($\widehat{\eta}$) does not use the true labels at all, fully trusting its CPE. We remark that $\boldsymbol{C}_t$ does *not* correspond to any empirical confusion matrix (as it is not based on the labels), so it should rather be treated as an internal parameter of OMMA($\widehat{\eta}$). In Appendix D we show that if the algorithm's probability estimator is *exact*, that is $\widehat{\boldsymbol{\eta}}_t \equiv \boldsymbol{\eta}$ for all t, OMMA($\widehat{\eta}$) achieves (under the same assumptions as before) a regret bound $R_n \leq \frac{Ma(2+\ln n)}{n}$, which is very similar to that of the original OMMA algorithm in Theorem 4.3. It is, however, unclear whether the modified algorithm converges for a non-exact CPE. Still, it turns out that OMMA($\widehat{\eta}$) performs surprisingly well in the experiments.

Sparse variant for a large number of labels. The introduced OMMA algorithm requires calculating the derivative with respect to every entry of the confusion matrix at each step, resulting in the total complexity of $\mathcal{O}(nm)$ over the entire sequence of data (assuming constant time for the gradient calculation of a single entry). Even this can be fairly expensive in case of a large number of labels (e.g., in recommendation or extreme classification); in these multi-label problems, however, the number of positive labels per sample is much smaller than the total number of label, $\|\boldsymbol{y}\|_1 \ll m$, and often most of the conditional probabilities in $\boldsymbol{\eta}(\boldsymbol{x})$ are 0 or very close to 0. Many recommenders and extreme classifiers are naturally designed to predict only top-k' entries with the highest values of $\eta(\boldsymbol{x})$. We can then leverage the sparsity of top-k' assuming $\eta(\boldsymbol{x}) = 0$ for the rest of the labels and calculate the gradient with respect to entries of the confusion matrix that correspond to non-zero values of $\boldsymbol{\eta}(\boldsymbol{x})$, resulting in a total complexity of $\mathcal{O}(nk')$ on the entire data sequence. With reasonably selected k', according to Theorem 4.3, we should only slightly increase the regret.

6. Empirical study

To demonstrate the practicality and generality of the introduced OMMA algorithm, we test it on a wide range of multi-label and multi-class benchmark datasets that differ substantially in the number of labels, ranging from tens to a few thousands, and in the imbalance of the label distribution [7]. For multi-class experiments, we use News20 (Lang, 1995), Ledgar-LexGlue (Chalkidis et al., 2022) with tf-idf features, Caltech-256 (Griffin et al., 2007) with features obtained using VGG16 (Simonyan & Zisserman, 2014) trained on ImageNet, and for multi-label experiments, we use YouTube, Flickr (Tang & Liu, 2009) with DeepWalk features (Perozzi et al., 2014), Eurlex-LexGlue (Chalkidis et al., 2021), Mediamill (Snoek et al., 2006), RCV1X (Lewis et al., 2004), and AmazonCat (McAuley & Leskovec, 2013; Bhatia et al., 2016) with tf-idf features. We conduct two types of experiments:

1. With fixed conditional probability estimator (CPE) – we train the CPE using L_2-regularized logistic loss on the provided training sets and use the obtained CPE to predict all $\widehat{\boldsymbol{\eta}}$ for the test set, which are then used in the online algorithms on the test set using the protocol from Figure 1. For benchmarks without default train and test sets, we split them randomly in proportion 70/30.

2. With online CPE – instead of training CPE on a separate training set, it incrementally updated on observed instances in the sequence. In this case, we concatenate train and test sets to create one long sequence used in the online protocol. We report the results of this experiment in Appendix H.

Each experiment is repeated five times, each time randomly shuffling the sequence. We report the mean results over all runs. In Appendix F, we include more details regarding the experimental setup.

We compare OMMA with the following algorithms:

- Top-$k(\widehat{\boldsymbol{\eta}}(\boldsymbol{x}))$ – a classifier that selects k labels with the highest values in $\widehat{\boldsymbol{\eta}}(\boldsymbol{x})$. It is used as a baseline in multi-class and multi-label problems with the budget k constrain. For multi-class problems we use $k = 1$ if the budget is not specified.

- $\widehat{\eta}(\boldsymbol{x}) > 0.5$ – a classifier with a constant threshold that predicts a label as positive if its conditional probability is greater than 0.5. It is used as a baseline in multi-label problems without the budget k constrain.

- OFO – a consistent online algorithm for the (micro- and macro-averaged) F-measure in the multi-label setting (Busa-Fekete et al., 2015; Jasinska et al., 2016).

- Greedy – an algorithm that given past labels and predictions chooses its next prediction in order to maximize the expected utility. It might be seen as a close approximation of the OMMA algorithm. It has been motivated

[7] Code to reproduce the experiments: https://github.com/mwydmuch/xCOLUMNs

Table 2. Results of the different online algorithms on *multi-label* problems, averaged over 5 runs, reported as %. In this table we report the final performance obtained on the sequence of n samples. The best result on each metric is in **bold**, the second best is in *italic*. We additionally report basic statistics of the benchmarks: number of labels m and instances in the test sequence n. $\times$ – means that the algorithm does not support the optimization of that metric.

Method	Micro F1	Macro F1	F1@3	Rec.@3	Pr.@3	G-mean	H-mean	Micro F1	Macro F1	F1@3	Rec.@3	Pr.@3	G-mean	H-mean
	YouTube ($m = 46, n = 7926$)							**Eurlex-LexGlue** ($m = 100, n = 5000$)						
Top-k / $\hat{\eta} > 0.5$	31.20	22.74	30.99	42.13	26.39	32.82	24.46	70.99	52.43	46.35	36.67	74.70	62.33	55.95
OFO	43.71	36.15	$\times$	$\times$	$\times$	$\times$	$\times$	73.23	58.93	$\times$	$\times$	$\times$	$\times$	$\times$
Greedy	$\times$	36.32	34.72	45.84	*67.18*	**77.98**	**77.93**	$\times$	59.83	54.19	52.67	*88.21*	89.74	89.73
Online-FW	43.67	36.00	34.38	45.83	38.92	*77.96*	*77.91*	**73.68**	59.53	54.18	52.68	58.76	89.69	89.75
Online-FW($\hat{\eta}$)	43.69	**36.47**	**35.43**	**45.89**	50.12	77.92	**77.93**	73.22	59.78	*54.34*	**53.87**	51.37	*89.91*	*89.84*
OMMA	**43.73**	*36.34*	34.81	*45.85*	**67.74**	**77.98**	**77.93**	*73.29*	**59.85**	54.15	52.67	**88.41**	89.74	89.73
OMMA($\widehat{\eta}$)	*43.72*	**36.47**	*35.38*	**45.89**	65.49	**77.98**	**77.93**	73.22	*59.84*	**54.37**	*53.85*	84.99	**89.92**	**89.85**
	Mediamill ($m = 101, n = 12914$)							**Flickr** ($m = 195, n = 24154$)						
Top-k / $\hat{\eta} > 0.5$	52.45	4.06	4.43	4.35	7.42	4.62	3.65	29.46	18.27	26.39	38.96	21.38	27.08	20.02
OFO	*56.99*	12.36	$\times$	$\times$	$\times$	$\times$	$\times$	**41.05**	30.46	$\times$	$\times$	$\times$	$\times$	$\times$
Greedy	$\times$	12.43	10.29	*9.41*	*22.54*	*65.68*	**66.48**	$\times$	30.90	*30.42*	**46.41**	*57.55*	*83.39*	**83.37**
Online-FW	56.98	12.22	10.18	9.16	12.70	64.73	*65.68*	**41.05**	30.60	29.58	*46.38*	28.66	83.37	*83.28*
Online-FW($\hat{\eta}$)	*56.99*	*14.33*	**11.97**	**16.43**	17.35	65.37	65.56	*41.02*	**31.17**	29.65	46.28	25.73	83.37	83.20
OMMA	**57.00**	12.39	10.25	9.39	**22.85**	65.67	**66.48**	41.01	30.90	30.39	**46.41**	**58.35**	*83.39*	**83.37**
OMMA($\widehat{\eta}$)	*56.99*	**14.34**	*11.87*	**16.43**	18.31	**65.93**	65.35	*41.02*	*31.15*	**30.55**	46.33	55.56	**83.41**	83.23
	RCV1X ($m = 2456, n = 155962$)							**AmazonCat** ($m = 13330, n = 306784$)						
Top-k / $\hat{\eta} > 0.5$	68.57	11.29	5.34	4.59	13.24	16.01	*12.36*	67.77	28.76	14.98	11.18	30.98	*33.93*	*30.03*
OFO	**69.83**	20.26	$\times$	$\times$	$\times$	$\times$	$\times$	*70.38*	39.60	$\times$	$\times$	$\times$	$\times$	$\times$
Greedy	$\times$	**20.80**	*16.01*	21.20	**30.91**	**69.07**	**67.04**	$\times$	44.20	43.64	57.81	**54.00**	**80.86**	**80.04**
Online-FW	**69.83**	19.82	15.33	21.09	19.88	**69.07**	**67.04**	**70.61**	42.42	40.20	57.74	40.10	**80.86**	**80.04**
Online-FW($\hat{\eta}$)	*69.79*	20.40	15.55	*22.21*	22.17	*69.06*	**67.04**	70.34	*47.32*	*44.63*	*58.87*	49.68	**80.86**	**80.04**
OMMA	69.77	20.57	15.87	21.08	*30.39*	**69.07**	**67.04**	69.90	43.04	42.14	57.80	*52.04*	**80.86**	**80.04**
OMMA($\widehat{\eta}$)	69.71	*20.71*	**16.07**	**22.23**	30.38	*69.06*	**67.04**	70.02	**47.67**	**45.06**	**58.89**	51.52	**80.86**	**80.04**

by a similar algorithm recently introduced for batch multi-label classification with predictions budgeted at k (Schultheis et al., 2023). An efficient implementation is possible for metrics that linearly decompose over labels (e.g., macro-averaged metrics).

- Online-FW – an adaptation of the Frank-Wolfe algorithm, used earlier for batch multi-class (Narasimhan et al., 2015b; 2022) and (budgeted at k) multi-label problems (Schultheis et al., 2024), to the online setting. It re-runs the batch algorithm on all instances observed so far (which need to be stored in memory) in exponentially growing intervals of 10×1.1^i instances with $i \in \{0, 1, 2, 3, \dots\}$, which we found to provide frequent enough updates for achieving good predictive performance. Since this algorithm, similar to OMMA, can be run for all metrics considered in the study, we treat it as the main baseline for our algorithm. For a fair comparison, we also introduce an alternative variant that uses $\widehat{\boldsymbol{\eta}}$ instead of true labels $\boldsymbol{y}$ to estimate its internal confusion matrix. We denote this algorithm as Online-FW($\widehat{\eta}$). Note that despite calling this method "online," it requires storing all previously seen instances in memory and thus does not fully adhere to the online paradigm.

- Offline-FW – the batch variant of the Frank-Wolfe algorithm used earlier in (Narasimhan et al., 2015b; Schultheis et al., 2024), trained using the same training set that was used to obtain CPE. It then uses the same probability estimates as online methods for inference on the test sequence without further updates to the classifier. The comparison with this algorithm is reported in Appendix G.

As discussed in Remark 4.4, we add both small constant $\epsilon =$ 1e-9 to the denominators in all metrics, as well as the regularization value $\lambda \in \{0, \text{1e-6}, \text{1e-3}, 0.1, 1\}$ to the entries of the confusion matrix used in all online algorithms (Greedy, OFO, Online-FW, and OMMA).

We use the following popular metrics for evaluating the algorithms: Micro and Macro-averaged F1, which are widely used to evaluate classifiers in many domains such as information retrieval; budgeted at $k = 3$ Macro F1, Macro Recall,

Table 3. Results of the online algorithms on *multi-class* problems, averaged over 5 runs, reported as %. In this table we report the final performance obtained on the sequence of n samples.

Method	Macro F1	Macro F1@3	Macro Rec.@3	Macro Pr.@3	Multi-class means G-	Multi-class means H-	Multi-class means Q-
	NEWS20 ($m = 20, n = 7532$)						
Top-k	**83.37**	49.23	94.78	33.95	82.51	81.66	80.12
Greedy	82.44	72.69	*94.89*	**83.89**	×	×	×
On.-FW	82.88	54.10	94.88	19.90	82.68	*82.35*	80.95
On.-FW($\hat{\eta}$)	82.55	56.06	**95.01**	15.58	**82.78**	**82.57**	*80.97*
OMMA	82.11	*72.84*	*94.89*	*83.88*	82.72	82.08	80.95
OMMA($\widehat{\eta}$)	*83.07*	**73.41**	**95.01**	83.30	*82.77*	82.18	**81.00**
	LEDGAR-LEXGLUE ($m = 100, n = 10000$)						
Top-k	79.06	51.80	92.04	38.88	0.00	0.00	69.21
Greedy	*79.30*	78.08	93.26	*91.17*	×	×	×
On.-FW	79.22	70.94	93.30	54.52	62.31	75.81	**74.59**
On.-FW($\hat{\eta}$)	79.22	73.85	*93.38*	49.63	*78.02*	**77.58**	74.50
OMMA	79.28	*78.10*	93.26	**91.41**	77.48	74.62	**74.59**
OMMA($\widehat{\eta}$)	**79.34**	**78.22**	**93.39**	90.10	**78.03**	*76.08*	*74.53*
	CALTECH-256 ($m = 256, n = 14890$)						
Top-k	79.45	46.82	89.85	32.58	77.32	75.69	74.53
Greedy	*79.59*	*79.02*	90.20	96.61	×	×	×
On.-FW	79.15	70.29	90.20	63.22	78.31	*77.99*	76.00
On.-FW($\hat{\eta}$)	79.29	72.97	*90.34*	65.42	**78.41**	**78.08**	*76.07*
OMMA	79.54	78.96	90.20	*96.77*	78.33	77.12	**76.15**
OMMA($\widehat{\eta}$)	**79.66**	**79.11**	**90.35**	**96.96**	*78.36*	77.01	75.99

and Macro Precision being well-suited to recommendation systems and extreme classification; and Macro G-mean and H-mean for multi-label problems and Multi-class G-mean, H-mean and Q-mean for multi-class problems, which are frequently used in class imbalance problems. Note that OMMA and Online-FW can target these metrics directly, while the scope of other algorithms is limited.

We present the results of the experiment with fixed CPE in Tables 2 and 3. We report the mean performance on the entire sequence of n instances for the best value of λ. Additionally, we present incremental performance on Figure 2 and the effect of using different λ values on Figure 3 for the Flickr dataset. The introduced OMMA algorithm matches the performance of the online Frank-Wolfe algorithm for most of the measures, performing much better on Macro-Precision@3, which is not Lipschitz. Concurrently, OMMA uses less time and memory as it does not require storing all previously seen instances. We also observe that OMMA, as an approximation of Greedy, matches its performance on metrics supported by this method. This is additionally confirmed by the plots where we can observe that the performance of these algorithms is very close to each other at each iteration t. Surprisingly, OMMA($\widehat{\eta}$) often performs better on macro-averaged metrics, especially on benchmarks with a large number of labels, where many of them have only a small number of positive samples. In these cases, the

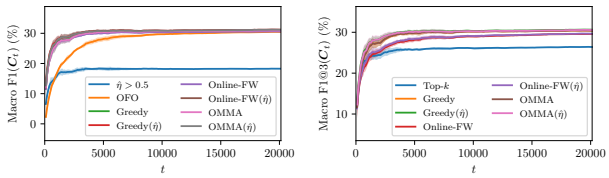

Figure 2. Comparison of the incremental performance of the online algorithms on the Flickr dataset. Averaged over 5 runs, the opaque fill indicates the standard deviation at given iteration t.

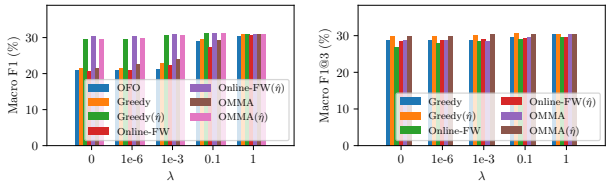

Figure 3. Impact of λ on the results of the online algorithms on the Flickr dataset. Averaged over 5 runs.

variance reduction effect of using $\widehat{\boldsymbol{\eta}}$ seems to compensate the estimation error of $\boldsymbol{\eta}$. It is also much less sensitive to the selection of λ, as small probability values that are added to many entries of the confusion matrix give the same effect as the regularization value λ. Extended results with additional algorithms, standard deviations, running times, and plots for the rest of the datasets can be found in Appendix G.

7. Discussion

In our framework, the algorithm aims to maximize the empirical value of the utility of the observed data sequence, $\psi(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n))$. An interesting direction of research is an alternative online framework, in which the goal is to sequentially learn a *classifier* $\widehat{\boldsymbol{h}}_n \colon \mathcal{X} \to \mathcal{Y}$, which is then evaluated on the entire population by means of $\psi(\boldsymbol{C}(\widehat{\boldsymbol{h}}_n))$, where $\boldsymbol{C}(\widehat{\boldsymbol{h}}_n) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}[\boldsymbol{C}(\boldsymbol{y}, \widehat{\boldsymbol{h}}_n(\boldsymbol{x}))]$ is the population confusion matrix of $\widehat{\boldsymbol{h}}_n$.

In the analysis of the algorithm, we assumed the concavity of the metric. On one hand, *some* assumption of this type seems essential for no-regret learning; for instance, in multi-class macro-averaged F-measure learning, the utility belongs to function class, optimizing which is, in general, NP-hard (Narasimhan et al., 2016). On the other hand, our concavity assumption excludes an important class of linear-fractional functions (such as binary or micro-averaged F-measure), for which a specialized version of our algorithm is known to converge, although without providing a convergence rate (Busa-Fekete et al., 2015). The analysis of our method for linear-fractional utilities is an interesting direction for future research, especially given that our algorithm works very well on these metrics in the experiments.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

Acknowledgments

This research has been supported by Poznan University of Technology via grant no. 0311/SBAD/0758 and by Academy of Finland via grants no. 347707 and 348215.

References

Babbar, R. and Schölkopf, B. Dismec: Distributed sparse machines for extreme multi-label classification. In *Proceedings of the tenth ACM international conference on web search and data mining*, pp. 721–729, 2017.

Baldi, P., Brunak, S., Chauvin, Y., Andersen, C. A., and Nielsen, H. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 16 (5):412–424, 2000.

Beck, A. *First-order methods in optimization*. SIAM, 2017.

Beck, A. and Teboulle, M. Smoothing and first order methods: A unified framework, 2012.

Bhatia, K., Dahiya, K., Jain, H., Mittal, A., Prabhu, Y., and Varma, M. The extreme classification repository: Multi-label datasets and code, 2016. URL http://manikvarma.org/downloads/XC/XMLRepository.html.

Bubeck, S. Convex optimization: Algorithms and complexity. 2015.

Busa-Fekete, R., Szörényi, B., Dembczynski, K., and Hüllermeier, E. Online F-measure optimization. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 595–603, 2015.

Cao, K., Wei, C., Gaidon, A., Arechiga, N., and Ma, T. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems*, 32, 2019.

Cesa-Bianchi, N. and Lugosi, G. *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, USA, 2006.

Chalkidis, I., Jana, A., Hartung, D., Bommarito, M., Androutsopoulos, I., Katz, D. M., and Aletras, N. Lexglue: A benchmark dataset for legal language understanding in english. *arXiv preprint arXiv:2110.00976*, 2021.

Chalkidis, I., Jana, A., Hartung, D., Bommarito, M., Androutsopoulos, I., Katz, D., and Aletras, N. LexGLUE: A benchmark dataset for legal language understanding in English. In Muresan, S., Nakov, P., and Villavicencio, A. (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4310–4330, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.297.

Dembczynski, K., Kotlowski, W., Koyejo, O., and Natarajan, N. Consistency analysis for binary classification revisited. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 961–969. PMLR, 2017.

Dembczyński, K., Kotłowski, W., Koyejo, O., and Natarajan, N. Consistency analysis for binary classification revisited. In *Proceedings of the 34th International Conference on Machine Learning (ICML 2017)*, volume 70 of *Proceedings of Machine Learning Research*, pp. 961–969. PMLR, 2017.

Drummond, C. and Holte, R. C. Severe class imbalance: Why better algorithms aren't the answer. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, pp. 539–546. Springer, 2005.

Elkan, C. The foundations of cost-sensitive learning. JCAI'01, pp. 973–978, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

Fan, R., Chang, K., Hsieh, C., Wang, X., and Lin, C. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

Griffin, G., Holub, A., and Perona, P. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

Hazan, E. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

Hazan, E. and Kale, S. Projection-free online learning. International Conference on Machine Learning (ICML), pp. 521–528, 2012.

Hoeffding, W. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

Jansche, M. A maximum expected utility framework for binary sequence labeling. In Zaenen, A. and van den Bosch, A. (eds.), *Proceedings of the 45th Annual Meeting*

of the Association of Computational Linguistics, pp. 736–743, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

Jasinska, K., Dembczynski, K., Busa-Fekete, R., Pfannschmidt, K., Klerx, T., and Hüllermeier, E. Extreme F-measure maximization using sparse probability estimates. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 1435–1444. JMLR.org, 2016.

Joachims, T. A support vector method for multivariate performance measures. In *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005), August 7-11, 2005, Bonn, Germany*, 2005.

Kar, P., Narasimhan, H., and Jain, P. Online and stochastic gradient methods for non-decomposable loss functions. *Advances in Neural Information Processing Systems*, 27, 2014.

Kennedy, K., Mac Namee, B., and Delany, S. J. Learning without default: A study of one-class classification and the low-default portfolio problem. In *Irish conference on artificial intelligence and cognitive science*, pp. 174–187. Springer, 2009.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y. (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

Kotłowski, W. and Dembczyński, K. Surrogate regret bounds for generalized classification performance metrics. *Machine Learning*, 10:549–572, 2017.

Koyejo, O., Natarajan, N., Ravikumar, P., and Dhillon, I. S. Consistent binary classification with generalized performance metrics. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 2744–2752, 2014.

Koyejo, O., Natarajan, N., Ravikumar, P., and Dhillon, I. S. Consistent multilabel classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 3321–3329, 2015.

Lang, K. Newsweeder: Learning to filter netnews. In *Machine learning proceedings 1995*, pp. 331–339. Elsevier, 1995.

Lewis, D. D. Evaluating and optimizing autonomous text classification systems. In *Proceedings of the 18th Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pp. 246–254. ACM, 1995.

Lewis, D. D., Yang, Y., Russell-Rose, T., and Li, F. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397, 2004.

Liu, M., Zhang, X., Zhou, X., and Yang†, T. Faster online learning of optimal threshold for consistent F-measure optimization. 2018.

McAuley, J. and Leskovec, J. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pp. 165–172, 2013.

Menon, A. K., Narasimhan, H., Agarwal, S., and Chawla, S. On the statistical consistency of algorithms for binary classification under class imbalance. In *International Conference on Machine Learning (ICML)*, 2013.

Narasimhan, H., Vaish, R., and Agarwal, S. On the statistical consistency of plug-in classifiers for non-decomposable performance measures. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 1493–1501, 2014.

Narasimhan, H., Kar, P., and Jain, P. Optimizing non-decomposable performance measures: A tale of two classes. In *ICML*, volume 37, pp. 199–208, 2015a.

Narasimhan, H., Ramaswamy, H., Saha, A., and Agarwal, S. Consistent multiclass algorithms for complex performance measures. In Bach, F. and Blei, D. (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2398–2407, Lille, France, 07 2015b. PMLR.

Narasimhan, H., Pan, W., Kar, P., Protopapas, P., and Ramaswamy, H. G. Optimizing the multiclass f-measure via biconcave programming. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 1101–1106, 2016.

Narasimhan, H., Ramaswamy, H. G., Tavker, S. K., Khurana, D., Netrapalli, P., and Agarwal, S. Consistent multiclass algorithms for complex metrics and constraints, 2022.

Natarajan, N., Dhillon, I. S., Ravikumar, P., and Tewari, A. Cost-sensitive learning with noisy labels. *Journal of Machine Learning Research*, 18(155):1–33, 2018.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

Perozzi, B., Al-Rfou, R., and Skiena, S. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pp. 701–710. ACM, 2014.

Reddi, S. J., Sra, S., Póczos, B., and Smola, A. Stochastic frank-wolfe methods for nonconvex optimization. *Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 1244–1251, 2016.

Schultheis, E., Wydmuch, M., Kotłowski, W., Babbar, R., and Dembczyński, K. Generalized test utilities for long-tail performance in extreme multi-label classification. In *Neural Information Processing Systems (NeurIPS)*, 2023.

Schultheis, E., Kotłowski, W., Wydmuch, M., Babbar, R., Borman, S., and Dembczyński, K. Consistent algorithms for multi-label classification with macro-at-k metrics. In *International Conference on Learning Representations (ICLR)*, 2024.

Shalev-Shwartz, S. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2012.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Singh, S. and Khim, J. T. Optimal binary classification beyond accuracy. *Advances in Neural Information Processing Systems*, 35:18226–18240, 2022.

Snoek, C. G. M., Worring, M., van Gemert, J. C., Geusebroek, J.-M., and Smeulders, A. W. M. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *ACM International Conference on Multimedia*, pp. 421–430. Association for Computing Machinery, 2006.

Tang, L. and Liu, H. Relational learning via latent social dimensions. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pp. 817–826. Association for Computing Machinery, 2009.

Waegeman, W., Dembczynski, K., Jachnik, A., Cheng, W., and Hüllermeier, E. On the bayes-optimality of F-measure maximizers. *Journal of Machine Learning Research*, 15(1):3333–3388, 2014.

Wang, S. and Yao, X. Multiclass imbalance problems: Analysis and potential solutions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 42(4): 1119–1130, 2012.

Yan, Y., Yang, T., Yang, Y., and Chen, J. A framework of online learning with imbalanced streaming data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31 (1), 2017.

Ye, N., Chai, K. M. A., Lee, W. S., and Chieu, H. L. Optimizing F-measure: A tale of two approaches. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress, 2012.

A. Motivation for the OMMA algorithm

In this section, we motivate the OMMA update (3) as a $O(1/t^2)$ approximation of a greedy optimization method defined below. As this section is only meant to be a motivation form the algorithm, we keep the derivation somewhat informal. Fix a trial t, and let $\boldsymbol{y}^{t-1}$ and $\widehat{\boldsymbol{y}}^{t-1}$ be the past predictions and labels, respectively. Given $\boldsymbol{x}_t$, let $\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)$ be the conditional probability estimate, which we concisely denote as $\widehat{\boldsymbol{\eta}}_t$. The greedy method is defined as returning prediction which *maximizes the $\widehat{\boldsymbol{\eta}}_t$-expected utility up to (and including) trial t* (the expectation is with respect to label $\boldsymbol{y}_t$, which is unknown at the moment of making prediction). That is, the greedy algorithm maximizes:

$$F(\widehat{\boldsymbol{y}}_t) = \mathbb{E}_{\boldsymbol{y}_t \sim \boldsymbol{\eta}_t}\left[\psi(\boldsymbol{C}(\boldsymbol{y}^t, \widehat{\boldsymbol{y}}^t))\right]$$

with respect to $\widehat{\boldsymbol{y}}_t$. Denote $\boldsymbol{C}_t = \boldsymbol{C}(\boldsymbol{y}^t, \widehat{\boldsymbol{y}}^t)$, $\boldsymbol{C}_{t-1} = \boldsymbol{C}(\boldsymbol{y}^{t-1}, \widehat{\boldsymbol{y}}^{t-1})$, and $\boldsymbol{\Delta}_t = \boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)$. Note that by (1) we have

$$\boldsymbol{C}_t = \frac{t-1}{t}\boldsymbol{C}_{t-1} + \frac{1}{t}\boldsymbol{\Delta}_t = \boldsymbol{C}_{t-1} + \frac{1}{t}(\boldsymbol{\Delta}_t - \boldsymbol{C}_{t-1}).$$

Taylor-expanding $\psi(\boldsymbol{C}_t)$ around $\boldsymbol{C}_{t-1}$, and leaving only the first-order terms gives

$$\begin{aligned}\psi(\boldsymbol{C}_t) &= \psi(\boldsymbol{C}_{t-1}) + \frac{1}{t}\nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{\Delta}_t - \boldsymbol{C}_{t-1}) + O\left(\frac{1}{t^2}\right) \\ &= \mathrm{const}(\boldsymbol{\Delta}_t) + \frac{1}{t}\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \boldsymbol{\Delta}_t + O\left(\frac{1}{t^2}\right).\end{aligned}$$

Taking expectation with respect to $\boldsymbol{y}_t$ on both sides gives:

$$F(\widehat{\boldsymbol{y}}_t) = \mathrm{const}(\widehat{\boldsymbol{y}}_t) + \mathbb{E}_{\boldsymbol{y}_t \sim \boldsymbol{\eta}_t}\left[\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)\right] + O\left(\frac{1}{t^2}\right),$$

maximizing of which is equivalent (up to $O(1/t^2)$ to OMMA update (3).

B. Regret bounds

B.1. Proof of Theorem 4.3

Let $\boldsymbol{h}^\star$ be the optimal classifier. Let

$$\boldsymbol{C}^\star = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}\left[\boldsymbol{C}(\boldsymbol{y}, \boldsymbol{h}^\star(\boldsymbol{x}))\right], \tag{7}$$

denote its *population confusion matrix*, which is the expected value of confusion matrix entries on an instance $(\boldsymbol{x}, \boldsymbol{y})$ randomly drawn from $\mathbb{P}$. Due to concavity of ψ, we have:

$$\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})^n}\left[\psi(\boldsymbol{C}(\boldsymbol{y}^n, \boldsymbol{h}^\star(\boldsymbol{x}^n)))\right] \leq \psi\left(\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})^n}\left[\boldsymbol{C}(\boldsymbol{y}^n, \boldsymbol{h}^\star(\boldsymbol{x}^n))\right]\right) = \psi\left(\boldsymbol{C}^\star\right),$$

thus the regret can be upper-bounded by:

$$R_n \leq \psi(\boldsymbol{C}^\star) - \mathbb{E}\left[\psi(\boldsymbol{C}_n)\right]. \tag{8}$$

where we used $\boldsymbol{C}_n = \boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)$. Comparing the algorithm with the population version of the optimal confusion matrix, rather than the expected empirical one, will turn out to be useful in making an important argument later in the proof.

We now switch to the analysis of the OMMA update (3). One problematic issue there is that the algorithms makes use of $\widehat{\boldsymbol{\eta}}_t$, which might not coincide with the true conditional $\boldsymbol{\eta}$, potentially leading to different predictions in both cases. We will, however, show now that the discrepancy in algorithm's prediction due to $\widehat{\boldsymbol{\eta}}_t$ can be bounded by its estimation error $\|\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t) - \boldsymbol{\eta}(\boldsymbol{x}_t)\|$, and is thus well controlled. To this end, we write down the OMMA update (3) as:

$$\widehat{\boldsymbol{y}}_t = \operatorname*{argmax}_{\widehat{\boldsymbol{y}}} \widehat{F}(\widehat{\boldsymbol{y}}), \qquad \text{where } \widehat{F}(\widehat{\boldsymbol{y}}) = \nabla\psi(\boldsymbol{C}_{t-1}) \cdot \left(\mathbb{E}_{\boldsymbol{y}_t \sim \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)}\left[\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}})\right]\right). \tag{9}$$

Let $\widetilde{\boldsymbol{y}}_t$ be a hypothetical prediction made as in (9), but with $\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)$ replaced by the true conditional probability $\boldsymbol{\eta}(\boldsymbol{x}_t)$, that is

$$\widetilde{\boldsymbol{y}}_t = \operatorname*{argmax}_{\widehat{\boldsymbol{y}}} \widetilde{F}(\widehat{\boldsymbol{y}}) \qquad \text{where } \widetilde{F}(\widehat{\boldsymbol{y}}) = \nabla\psi(\boldsymbol{C}_{t-1}) \cdot \left(\mathbb{E}_{\boldsymbol{y}_t \sim \boldsymbol{\eta}(\boldsymbol{x}_t)}\left[\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}})\right]\right).$$

We will bound:

$$\begin{aligned}\widetilde{F}(\widetilde{\boldsymbol{y}}_t) - \widetilde{F}(\widehat{\boldsymbol{y}}_t) &= \widetilde{F}(\widetilde{\boldsymbol{y}}_t) - \widehat{F}(\widetilde{\boldsymbol{y}}_t) + \overbrace{\widehat{F}(\widetilde{\boldsymbol{y}}_t) - \widehat{F}(\widehat{\boldsymbol{y}}_t)}^{\leq 0 \text{ from (9)}} + \widehat{F}(\widehat{\boldsymbol{y}}_t) - \widetilde{F}(\widehat{\boldsymbol{y}}_t) \\ &\leq 2 \max_{\widehat{\boldsymbol{y}}} |\widetilde{F}(\widehat{\boldsymbol{y}}) - \widehat{F}(\widehat{\boldsymbol{y}})| \\ &\leq 2\|\nabla\psi(\boldsymbol{C}_{t-1})\| \max_{\widehat{\boldsymbol{y}}} \Big\| \underbrace{\mathbb{E}_{\boldsymbol{y}_t\sim\boldsymbol{\eta}(\boldsymbol{x}_t)}\left[\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}})\right] - \mathbb{E}_{\boldsymbol{y}_t\sim\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)}\left[\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}})\right]}_{\boldsymbol{\Delta}_t} \Big\|,\end{aligned}$$

where the last inequality is from Cauchy-Schwarz inequality. Due to Lipschitzness of ψ, $\|\nabla\psi(\boldsymbol{C}_{t-1})\| \leq L$. For multi-class classification, $\boldsymbol{\Delta}_t$ is a matrix with entries equal to $\boldsymbol{\Delta}_{t,j\ell} = (\eta_j(\boldsymbol{x}_t) - \widehat{\eta}_{tj}(\boldsymbol{x}_t))\widehat{y}_\ell$. Due to one-hot constraint $\sum_\ell \widehat{y}_\ell = 1$, we get that for any $\widehat{\boldsymbol{y}}$, $\|\boldsymbol{\Delta}_t\|^2 = \sum_{j=1}^m |\eta_j(\boldsymbol{x}_t) - \widehat{\eta}_{tj}(\boldsymbol{x}_t)|^2 = \|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|^2$. For multi-label classification $\boldsymbol{\Delta}_t$ is a sequence of m 2×2 matrices, with j-th matrix having entries

$$\begin{pmatrix} (\widehat{\eta}_{tj}(\boldsymbol{x}_t) - \eta_j(\boldsymbol{x}_t))(1-\widehat{y}_j) & (\widehat{\eta}_{tj}(\boldsymbol{x}_t) - \eta_j(\boldsymbol{x}_t))\widehat{y}_j \\ (\eta_j(\boldsymbol{x}_t) - \widehat{\eta}_{tj}(\boldsymbol{x}_t))(1-\widehat{y}_j) & (\eta_j(\boldsymbol{x}_t) - \widehat{\eta}_{tj}(\boldsymbol{x}_t))\widehat{y}_j \end{pmatrix},$$

so that $\|\boldsymbol{\Delta}_t\|^2 = 2\sum_{j=1}^m |\eta_j(\boldsymbol{x}_t) - \widehat{\eta}_{tj}(\boldsymbol{x}_t)|^2 = 2\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|^2$. Using $b = 1$ for multi-class, and $b = \sqrt{2}$ for multi-label setting, we finally get:

$$\widetilde{F}(\widetilde{\boldsymbol{y}}_t) - \widetilde{F}(\widehat{\boldsymbol{y}}_t) \leq 2bL\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|,$$

or using the definition of $\widetilde{F}$ and rearranging,

$$\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \left(\mathbb{E}_{\boldsymbol{y}_t\sim\boldsymbol{\eta}_t(\boldsymbol{x}_t)}\left[\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)\right]\right) \geq \max_{\widehat{\boldsymbol{y}}} \left\{\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \left(\mathbb{E}_{\boldsymbol{y}_t\sim\boldsymbol{\eta}_t(\boldsymbol{x}_t)}\left[\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}})\right]\right)\right\} - 2bL\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|. \quad (10)$$

Thus, even thought the algorithm uses the estimator $\widehat{\boldsymbol{\eta}}_t$ in place of $\boldsymbol{\eta}$, the prediction it picks (left-hand side) is suboptimal with respect to the choice it should have made had it known true $\boldsymbol{\eta}$ (first term on the right-hand side) by at most the estimation error of $\widehat{\boldsymbol{\eta}}_t$ (second term on the right-hand side). We will use this result later in the proof.

We now proceed with the main part of the proof. From the concavity of the utility,

$$\psi(\boldsymbol{C}^\star) \leq \psi(\boldsymbol{C}_{t-1}) + \nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}^\star - \boldsymbol{C}_{t-1}),$$

or, after rearranging,

$$\nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}_{t-1} - \boldsymbol{C}^\star) \leq \psi(\boldsymbol{C}_{t-1}) - \psi(\boldsymbol{C}^\star). \quad (11)$$

It follows from the concavity of ψ and its smoothness that (see, e.g., Bubeck (2015)) for any $\boldsymbol{C}_1, \boldsymbol{C}_2 \in \mathcal{C}$,

$$\psi(\boldsymbol{C}_1) \geq \psi(\boldsymbol{C}_2) + \nabla\psi(\boldsymbol{C}_2) \cdot (\boldsymbol{C}_1 - \boldsymbol{C}_2) - \frac{M}{2}\|\boldsymbol{C}_1 - \boldsymbol{C}_2\|^2.$$

Substituting $\boldsymbol{C}_1 = \boldsymbol{C}_t$ and $\boldsymbol{C}_2 = \boldsymbol{C}_{t-1}$, we get:

$$\begin{aligned}\psi(\boldsymbol{C}_t) &\geq \psi(\boldsymbol{C}_{t-1}) + \nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}_t - \boldsymbol{C}_{t-1}) - \frac{M}{2}\|\boldsymbol{C}_t - \boldsymbol{C}_{t-1}\|_1^2 \\ &= \psi(\boldsymbol{C}_{t-1}) + \frac{1}{t}\nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t) - \boldsymbol{C}_{t-1}) - \frac{M}{2t^2}\|\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t) - \boldsymbol{C}_{t-1}\|^2\end{aligned}$$

where we used (1) to get:

$$\boldsymbol{C}_t = \frac{t-1}{t}\boldsymbol{C}_{t-1} + \frac{1}{t}\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t) = \boldsymbol{C}_{t-1} + \frac{1}{t}(\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t) - \boldsymbol{C}_{t-1}).$$

We upper-bound

$$\|\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t) - \boldsymbol{C}_{t-1}\|^2 \leq \max_{\boldsymbol{C}_1, \boldsymbol{C}_2 \in \mathcal{C}} \|\boldsymbol{C}_1 - \boldsymbol{C}_2\|^2.$$

We have

$$\|\boldsymbol{C}_1 - \boldsymbol{C}_2\|^2 = \|\boldsymbol{C}_1\|^2 - 2\boldsymbol{C}_1 \cdot \boldsymbol{C}_2 + \|\boldsymbol{C}_2\|^2 \leq \|\boldsymbol{C}_1\|^2 + \|\boldsymbol{C}_2\|^2 \leq \|\boldsymbol{C}_1\|_1 + \|\boldsymbol{C}_2\|_1,$$

where the first inequality follows from the fact that confusion matrix has nonnegative entries, while the second inequality follows from the fact that all entries are at most 1 (so that the sum of squares of the entries is not larger than the sum of their absolute values). For multi-class classification $\|\boldsymbol{C}\|_1 = 1$ for any $\boldsymbol{C} \in \mathcal{C}$, while for multi-label classification, $\|\boldsymbol{C}\|_1 = m$ for any $\boldsymbol{C} \in \mathcal{C}$. Thus, using $a = 1$ for multi-class, and $a = m$ for multi-label classification, we have:

$$\psi(\boldsymbol{C}_t) \geq \psi(\boldsymbol{C}_{t-1}) + \frac{1}{t}\nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t) - \boldsymbol{C}_{t-1}) - \frac{Ma}{t^2}.$$

Multiplying both sides of the inequality by (-1) followed by adding $\psi(\boldsymbol{C}^\star)$ gives:

$$\begin{aligned}
\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_t) &\leq \psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_{t-1}) + \frac{1}{t}\nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}_{t-1} - \boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)) + \frac{Ma}{t^2} \\
&= \psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_{t-1}) + \frac{1}{t}\nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}_{t-1} - \boldsymbol{C}^\star) + \frac{1}{t}\nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}^\star - \boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)) + \frac{Ma}{t^2} \\
&\leq \frac{t-1}{t}\left(\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_{t-1})\right) + \frac{1}{t}\nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}^\star - \boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)) + \frac{Ma}{t^2}
\end{aligned}$$

where in the last inequality we used (11). Multiplying both sides by t and denoting $A_t = t(\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_t))$, we get

$$A_t \leq A_{t-1} + \nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}^\star - \boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)) + \frac{Ma}{t}.$$

Now, let $\mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{y}_t)}[\cdot]$ denote the expectation with respect to $(\boldsymbol{x}_t, \boldsymbol{y}_t)$ (formally: conditioned on all the past $(\boldsymbol{x}^{t-1}, \boldsymbol{y}^{t-1})$). Since $\boldsymbol{C}_{t-1}$ does not depend on $(\boldsymbol{x}_t, \boldsymbol{y}_t)$, $\mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{y}_t)}[\boldsymbol{C}_{t-1}] = \boldsymbol{C}_{t-1}$ (and similarly for A_{t-1}). Applying this expectation to both sides of the inequality above gives:

$$\mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{y}_t)}[A_t] \leq A_{t-1} + \mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{y}_t)}\left[\nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}^\star - \boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t))\right] + \frac{Ma}{t}. \tag{12}$$

We proceed with further bounding the middle term on the right-hand side of (12). Using the rule of conditional expectation, $\mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{y}_t)}[\cdot] = \mathbb{E}_{\boldsymbol{x}_t}[\mathbb{E}_{\boldsymbol{y}_t \sim \boldsymbol{\eta}(\boldsymbol{x}_t)}[\cdot]]$,

$$\begin{aligned}
\mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{y}_t)}\left[\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)\right] &= \mathbb{E}_{\boldsymbol{x}_t}\left[\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \mathbb{E}_{\boldsymbol{y}_t \sim \boldsymbol{\eta}(\boldsymbol{x}_t)}\left[\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)\right]\right] \\
\text{(from (10))} \quad &\geq \mathbb{E}_{\boldsymbol{x}_t}\left[\max_{\widehat{\boldsymbol{y}}}\left\{\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \mathbb{E}_{\boldsymbol{y}_t \sim \boldsymbol{\eta}(\boldsymbol{x}_t)}\left[\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}})\right]\right\}\right] - 2bL\,\mathbb{E}_{\boldsymbol{x}_t}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right],
\end{aligned}$$

or, after rearranging,

$$\mathbb{E}_{\boldsymbol{x}_t}\left[\max_{\widehat{\boldsymbol{y}}}\left\{\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \mathbb{E}_{\boldsymbol{y}_t \sim \boldsymbol{\eta}(\boldsymbol{x}_t)}\left[\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}})\right]\right\}\right] \leq \mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{y}_t)}\left[\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)\right] + 2bL\,\mathbb{E}_{\boldsymbol{x}_t}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right] \tag{13}$$

Using the fact that neither $\boldsymbol{C}_{t-1}$ nor $\boldsymbol{C}^\star$ depend on $(\boldsymbol{x}_t, \boldsymbol{y}_t)$ (here we employ the fact that $\boldsymbol{C}^\star$ is the population optimal matrix, which does not depend on any data), we rewrite:

$$\begin{aligned}
\mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{y}_t)}\left[\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \boldsymbol{C}^\star\right] &= \nabla\psi(\boldsymbol{C}_{t-1}) \cdot \boldsymbol{C}^\star \\
\text{(from (7))} \quad &= \nabla\psi(\boldsymbol{C}_{t-1}) \cdot \mathbb{E}_{(\boldsymbol{x}, \boldsymbol{y})}[\boldsymbol{C}(\boldsymbol{y}, \boldsymbol{h}^\star(\boldsymbol{x})] \\
((\boldsymbol{x}, \boldsymbol{y}) \text{ has the same distr. as } (\boldsymbol{x}_t, \boldsymbol{y}_t)) \quad &= \nabla\psi(\boldsymbol{C}_{t-1}) \cdot \mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{y}_t)}[\boldsymbol{C}(\boldsymbol{y}_t, \boldsymbol{h}^\star(\boldsymbol{x}_t)] \\
(\boldsymbol{C}_{t-1} \text{ does not depend on } (\boldsymbol{x}_t, \boldsymbol{y}_t)) \quad &= \mathbb{E}_{\boldsymbol{x}_t}\left[\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \mathbb{E}_{\boldsymbol{y}_t \sim \boldsymbol{\eta}(\boldsymbol{x}_t)}\left[\boldsymbol{C}(\boldsymbol{y}_t, \boldsymbol{h}^\star(\boldsymbol{x}_t)\right]\right] \\
&\leq \mathbb{E}_{\boldsymbol{x}_t}\left[\max_{\widehat{\boldsymbol{y}}}\left\{\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \mathbb{E}_{\boldsymbol{y}_t \sim \boldsymbol{\eta}(\boldsymbol{x}_t)}\left[\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}})\right]\right\}\right] \\
\text{(from (13))} \quad &\leq \mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{y}_t)}\left[\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)\right] + 2bL\,\mathbb{E}_{\boldsymbol{x}_t}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right].
\end{aligned}$$

Thus, the middle term on the right-hand side of (12) can be bounded by

$$\mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{y}_t)}\left[\nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}^\star - \boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t))\right] \leq 2bL\,\mathbb{E}_{\boldsymbol{x}_t}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right],$$

which, after substituting to (12) and taking the expectation on both sides with respect to all the past $(\boldsymbol{x}^{t-1}, \boldsymbol{y}^{t-1})$,

$$\mathbb{E}[A_t] \leq \mathbb{E}[A_{t-1}] + 2bL\,\mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right] + \frac{Ma}{t}.$$

Applying the above to $t = n$, and recursively expanding,

$$\begin{aligned}
\mathbb{E}[A_n] &\leq \mathbb{E}[A_{n-1}] + 2bL\,\mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_n) - \widehat{\boldsymbol{\eta}}_n(\boldsymbol{x}_n)\|\right] + \frac{Ma}{n} \\
&\leq \mathbb{E}[A_{n-2}] + 2bL\,\mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_{n-1}) - \widehat{\boldsymbol{\eta}}_{n-1}(\boldsymbol{x}_{n-1})\|\right] + 2bL\,\mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_n) - \widehat{\boldsymbol{\eta}}_n(\boldsymbol{x}_n)\|\right] + \frac{Ma}{n} + \frac{Ma}{n-1} \\
&\leq \ldots \leq \mathbb{E}[A_0] + 2bL\sum_{t=1}^{n}\mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right] + Ma\underbrace{\sum_{t=1}^{n}\frac{1}{t}}_{H_n} \\
&\leq \mathbb{E}[A_0] + 2bL\sum_{t=1}^{n}\mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right] + Ma(1+\ln n), \qquad (14)
\end{aligned}$$

where we bounded the harmonic number $H_n = \sum_{t=1}^{n} t^{-1}$ by $1 + \ln n$. Recall that $A_t = t(\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_t))$, which gives $A_0 = 0$ (our recursion (12) was also valid for $t = 1$). After dividing both sides of (14) by n and using (8) we get

$$R_n \leq \frac{Ma(1+\ln n)}{n} + \frac{2bL}{n}\sum_{t=1}^{n}\mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right],$$

which was to be shown. □

B.2. Non-smooth utilities

While all commonly used utilities are smooth functions of confusion matrix, it is possible to modify the OMMA algorithm so that it achieves vanishing regret even for a non-smooth utility ψ, but at a price of a slower convergence rate $O(\sqrt{(\ln n)/n})$. For the sake of clarity, here we give a regret bound for the case when the time horizon n is known ahead of time.

The modification of the algorithm is based on optimizing a *smoothed* version of ψ.

A collection of convex functions $\{\psi_\gamma \colon \gamma > 0\}$ is called a γ-smoothing of a convex and closed function ψ with parameters (α, β) if $0 \leq \psi(\boldsymbol{C}) - \psi_\gamma(\boldsymbol{C}) \leq \alpha\gamma$ for all $\boldsymbol{C} \in \mathcal{C}$, and ψ_γ is $\frac{\beta}{\gamma}$-smooth (Beck & Teboulle, 2012). That is, γ dictates a trade-off between the smoothness of ψ_γ and its deviation from the original function ψ.

Running the OMMA algorithm with ψ replaced by its $\frac{\beta}{\gamma}$-smoothed (concave) ψ_γ results in the regret bound from Theorem 4.3:

$$R_n^\gamma \leq \frac{\beta a(1+\ln n)}{\gamma n} + \frac{2bL_\gamma}{n}\sum_{t=1}^{n}\mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right]$$

where $R_n^\gamma = \mathbb{E}\left[\psi_\gamma(\boldsymbol{C}(\boldsymbol{y}^n, \boldsymbol{h}^\star(\boldsymbol{x}^n)))\right] - \mathbb{E}\left[\psi_\gamma(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n))\right]$ is the regret defined in terms of ψ_γ rather than the original function ψ, while L_γ is the Lipschitz constant of ψ_γ. Using the property of γ-smoothing $0 \leq \psi(\boldsymbol{C}) - \psi_\gamma(\boldsymbol{C}) \leq \alpha\gamma$ twice to $\boldsymbol{C} = \boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)$ and $\boldsymbol{C} = \boldsymbol{C}(\boldsymbol{y}^n, \boldsymbol{h}^\star(\boldsymbol{x}^n))$ we relate the regret with respect to the original utility with that of its smoothed version:

$$R_n = \mathbb{E}\left[\psi(\boldsymbol{C}(\boldsymbol{y}^n, \boldsymbol{h}^\star(\boldsymbol{x}^n)))\right] - \mathbb{E}\left[\psi(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n))\right] \leq \mathbb{E}\left[\psi_\gamma(\boldsymbol{C}(\boldsymbol{y}^n, \boldsymbol{h}^\star(\boldsymbol{x}^n)))\right] - \mathbb{E}\left[\psi_\gamma(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n))\right] + \alpha\gamma = R_n^\gamma + \alpha\gamma.$$

Thus the resulting bound on the ψ-regret of the algorithm is:

$$R_n \leq \frac{\beta a(1+\ln n)}{\gamma n} + \frac{2bL_\gamma}{n}\sum_{t=1}^{n}\mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right] + \alpha\gamma.$$

The optimal choice of γ (i.e., the one minimizing the right-hand side of the bound) is $\gamma = \sqrt{\frac{a\beta(1+\ln n)}{\alpha n}}$, which results in:

$$R_n \leq 2\sqrt{\frac{a\alpha\beta(1+\ln n)}{n}} + \frac{2bL_\gamma}{n}\sum_{t=2}^{n}\mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right]. \qquad (15)$$

An example of γ-smoothing is the *infimal convolution smoothing* with Moreau-Yoshida regularization (Beck, 2017):

$$\psi_\gamma(\boldsymbol{C}) = \max_{\boldsymbol{C}'} \left\{ \psi(\boldsymbol{C}') - \frac{1}{2\gamma}\|\boldsymbol{C} - \boldsymbol{C}'\|^2 \right\},$$

which has parameters $\alpha = \frac{L^2}{2}$, $\beta = 1$, and $L_\gamma = L$, where L is the Lipschitz constant of ψ. Using it in the algorithm, the bound (15) becomes:

$$R_n \leq L\sqrt{\frac{2a(1+\ln n)}{n}} + \frac{2bL}{n}\sum_{t=2}^{n} \mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right].$$

C. The adversarial setting is too difficult

Consider the following variation of the problem. At trials $t = 1, \ldots, n$:

- The algorithm receives input instance $\boldsymbol{x}_t$ together with its conditional label probability vector $\boldsymbol{\eta}_t \equiv \boldsymbol{\eta}(\boldsymbol{x}_t)$
- The algorithm issues prediction $\widehat{\boldsymbol{y}}_t$
- The label $\boldsymbol{y}_t$ is drawn from $\boldsymbol{\eta}_t$ and revealed to the algorithm

Note that we allow the algorithm to know *exact* conditional probabilities (rather than only their estimate $\widehat{\boldsymbol{\eta}}_t$), but we do not make any assumptions about the sequence of inputs $\boldsymbol{x}^n$ (in particular, we do not assume they are generated i.i.d.). At the end of the sequence, the algorithm evaluated by the expected utility, $\mathbb{E}_{\boldsymbol{y}^n \sim \boldsymbol{\eta}^n}\, \psi(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)$, and is compared with the set of predictions maximizing the expected utility:

$$\boldsymbol{y}^{\star n} = \operatorname*{argmax}_{\widehat{\boldsymbol{y}}^n} \mathbb{E}_{\boldsymbol{y}^n \sim \boldsymbol{\eta}^n}\left[\psi(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)\right]. \tag{16}$$

The regret is given by:

$$R_n = \mathbb{E}\left[\psi(\boldsymbol{y}^n, \boldsymbol{y}^{\star n})\right] - \mathbb{E}\left[\psi(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)\right],$$

where we dropped the subscript in expectation as clear from the context. We will show that in this setting, any algorithm has a regret which is at least $\frac{1}{36} - o(1)$. To this end, take a binary classification setting ($m = 2$). As in the example in Section 3, to simplify the presentation we turn the one-hot vector notation into scalars, and repeat the translation here for the sake of convenience. Let $y_t \in \{0, 1\}$ denote a label at time t (where $y_t = 0$ corresponds to $\boldsymbol{y}_t = (1, 0)$ in the one-hot notation, while $y_t = 1$ to $\boldsymbol{y}_t = (0, 1)$). Similarly, let $\widehat{y}_t \in \{0, 1\}$ denote the prediction. The conditional probability is $\eta_t = P(y_t = 1|\boldsymbol{x}_t)$ (which corresponds to $\eta_1(\boldsymbol{x}_t)$ in the vector notation). The confusion matrix $\boldsymbol{C}_t := \boldsymbol{C}(\boldsymbol{y}^t, \widehat{\boldsymbol{y}}^t)$ consists of four entries $C_{t-1,j\ell} = \sum_{i \leq t} \mathbf{1}\{y_i = j\}\mathbf{1}\{\widehat{y}_i = \ell\}$, $j, \ell \in \{0, 1\}$ (true negatives, false positives, false negatives, true positives).

We will consider the following utility:

$$\psi(\boldsymbol{C}) = \min\{C_{00}, C_{11}\},$$

which is concave and 1-Lipschitz. While ψ is non-smooth, we can show a similar bound with a smoothed version of it, however, it would require much more involved analysis. Given predictions $\widehat{y}^n$, the expected value (with respect to y^n) of $nC_{n,11}$ can be calculated as:

$$n\,\mathbb{E}[C_{n,11}] = (n-1)\,\mathbb{E}[C_{n-1,11}] + \mathbb{E}[\widehat{y}_t\,\mathbb{E}_{y_t}[y_t]] = (n-1)\,\mathbb{E}[C_{n-1,11}] + \mathbb{E}[\widehat{y}_t\eta_t] = \ldots = \mathbb{E}\left[\sum_t \widehat{y}_t\eta_t\right],$$

where we used the fact that $\widehat{y}_t$ does not depend on y_t. A similar method can be applied to $nC_{n,00}$. Thus, we get:

$$\mathbb{E}[C_{n,11}] = \mathbb{E}\left[\frac{1}{n}\sum_{t=1}^{n}\widehat{y}_t\eta_t\right], \qquad \mathbb{E}[C_{n,00}] = \mathbb{E}\left[\frac{1}{n}\sum_{t=1}^{n}(1-\widehat{y}_t)(1-\eta_t)\right]. \tag{17}$$

Now, as ψ is concave, we have

$$\mathbb{E}\left[\psi(\boldsymbol{C}_n)\right] \leq \psi\left(\mathbb{E}\left[\boldsymbol{C}_n\right]\right), \tag{18}$$

and the regret can be lower-bounded by:

$$R_n \geq \mathbb{E}\left[\psi(y^n, y^{\star n})\right] - \psi\left(\mathbb{E}\left[\boldsymbol{C}_n\right]\right).$$

We now turn into defining a sequence of outcomes. Since the only property of inputs $\boldsymbol{x}_t$ used by the algorithm (and appearing in the regret) is the conditional probability η_t, we will directly operate on sequences of η_t's. Let $n = 2m$ be even, and suppose the algorithm observes for the first half of the game $\eta_1 = \eta_2 = \ldots = \eta_m = \frac{2}{3}$. In the second half of the game, the algorithm will observe one of the two sequences:

1. either $\eta_{m+1} = \eta_{m+2} = \ldots = \eta_n = \frac{2}{3}$ (sequence **1**),
2. or $\eta_{m+1} = \eta_{m+2} = \ldots = \eta_n = \frac{1}{3}$ (sequence **2**).

We will exploit the fact that the algorithm up to iteration m does not know which sequence it should prepare for, while the optimal predictions for both sequences are substantially different. Let $S_1 = \mathbb{E}\left[m^{-1}\sum_{t=1}^{m}\widehat{y}_t\right]$ and $S_2 = \mathbb{E}\left[m^{-1}\sum_{t=m+1}^{n}\widehat{y}_t\right]$ be the expected average values of the predictions of the algorithm in the first half and second half of the sequence, respectively. Using (17), and the fact the first m elements of the sequence are all equal to η_1, while the last m elements are all equal to η_n, we have

$$\mathbb{E}[C_{n,11}] = \frac{1}{n}mS_1\eta_1 + mS_2\eta_n = \frac{S_1\eta_1 + S_2\eta_n}{2},$$

and similarly $\mathbb{E}[C_{n,10}] = \frac{(1-S_1)(1-\eta_1)+(1-S_2)(1-\eta_n)}{2}$. Using (18), the performance of the algorithm for sequence **1** ($\eta_1 = \eta_n = \frac{2}{3}$) is at most:

$$\text{(sequence } \mathbf{1}\text{)} \quad \mathbb{E}\left[\psi(\boldsymbol{C}_n)\right] \leq \min\left\{\frac{1}{3}(S_1 + S_2), \frac{1}{6}(2 - S_1 - S_2)\right\}, \tag{19}$$

while for sequence **2** ($\eta_1 = \frac{2}{3}, \eta_n = \frac{1}{3}$) it is at most:

$$\text{(sequence } \mathbf{2}\text{)} \quad \mathbb{E}\left[\psi(\boldsymbol{C}_n)\right] \leq \min\left\{\frac{1}{6}(2S_1 + S_2), \frac{1}{6}(3 - S_1 - 2S_2)\right\} \tag{20}$$

Note that we must have $0 \leq S_1, S_2 \leq 1$ as these corresponds to expected averages of predictions of the algorithm. Furthermore S_2 can be optimally chosen by the algorithm *separately* for case **1** and **2**, as it corresponds to the second half of predictions, when the algorithm already knows whether the sequence is of type **1** or **2**. In both cases, the optimal choice (maximizing the right-hand sides of (19)-(20)) is to make both terms under min equal. For sequence **2** it is always possible:

$$\frac{1}{6}(2S_1 + S_2) = \frac{1}{6}(3 - S_1 - 2S_2) \quad \Rightarrow \quad S_2 = 1 - S_1,$$

which leads to a bound:

$$\text{(sequence } \mathbf{2}\text{)} \quad \mathbb{E}\left[\psi(\boldsymbol{C}_n)\right] \leq \frac{1}{6}(1 + S_1) \tag{21}$$

For sequence **1** we get:

$$\frac{1}{3}(S_1 + S_2) = \frac{1}{6}(2 - S_1 - S_2) \quad \Rightarrow \quad S_2 = \frac{2}{3} - S_1,$$

which is only possible when $S_1 \leq \frac{2}{3}$. Thus, we set $S_2 = \max\left\{0, \frac{2}{3} - S_1\right\}$, which gives

$$\text{(sequence } \mathbf{1}\text{)} \quad \mathbb{E}\left[\psi(\boldsymbol{C}_n)\right] \leq \min\left\{\frac{2}{9}, \frac{2 - S_1}{6}\right\}, \tag{22}$$

We will lower bound the value of the optimal utility by simply picking a particular set of predictions $y^{\star n}$ rather than using (16). Assume n is divisible by 3. For sequence **1**, set $y_t^\star = 1$ in the first one-third of the sequence, while $y_t^\star = 0$ in the last two-thirds of the sequence. Let $Y_1 = \sum_{t=1}^{n/3} y_i$ and $Y_2 = \sum_{t=n/3}^{n} y_i$. We get $C_{11}^\star := C_{11}(y^n, y^{\star n}) = Y_1/n$ and $C_{00}^\star = Y_2/n$. Note that $Y_1 \sim \text{Binomial}(n/3, 2/3)$, $Y_2 \sim \text{Binomial}(2n/3, 1/3)$, both having the same expectation $\mathbb{E}[Y_1] = \mathbb{E}[Y_2] = \frac{2n}{9}$. Using the fact that $Y_i - \mathbb{E}[Y_i]$, $i = 1, 2$, are subgaussian (as being binomial), we have for any $s \in \mathbb{R}$:

$$\mathbb{E}\left[e^{s(Y_1 - \mathbb{E}[Y_1])}\right] \leq e^{\frac{ns^2}{24}}, \qquad \mathbb{E}\left[e^{s(Y_2 - \mathbb{E}[Y_2])}\right] \leq e^{\frac{ns^2}{12}},$$

which we can plug into a standard derivation of a bound for the minimum of subgaussian variables. First note that:

$$\mathbb{E}\left[\min\{Y_1, Y_2\}\right] = \frac{2n}{9} - \mathbb{E}\left[\max\{\mathbb{E}[Y_1] - Y_1, \mathbb{E}[Y_1] - Y_2\}\right].$$

To bound the maximum, we have for any $s > 0$:

$$\begin{aligned}
\mathbb{E}\left[\max\{\mathbb{E}[Y_1] - Y_1, \mathbb{E}[Y_1] - Y_2\}\right] &= \frac{1}{s}\,\mathbb{E}\left[\ln e^{s\max\{\mathbb{E}[Y_1]-Y_1, \mathbb{E}[Y_1]-Y_2\}}\right] \\
\text{(Jensen's ineq.)} \quad &\le \frac{1}{s}\ln\mathbb{E}\left[e^{s\max\{\mathbb{E}[Y_1]-Y_1, \mathbb{E}[Y_1]-Y_2\}}\right] \le \frac{1}{s}\ln\mathbb{E}\left[e^{s(\mathbb{E}[Y_1]-Y_1)} + e^{s(\mathbb{E}[Y_1]-Y_1)}\right] \\
&\le \frac{1}{s}\ln\left(e^{\frac{ns^2}{24}} + e^{\frac{ns^2}{12}}\right) \le \frac{1}{s}\ln\left(2e^{\frac{ns^2}{12}}\right) = \frac{\ln 2}{s} + \frac{ns}{12} \\
\left(s = \sqrt{\frac{12\ln 2}{n}}\right) \quad &= 2\sqrt{\frac{\ln 2}{12n}} \le \frac{\sqrt{n}}{2}.
\end{aligned}$$

Thus, $\mathbb{E}\left[\min\{Y_1, Y_2\}\right] \ge \frac{2n}{9} - \frac{\sqrt{n}}{2}$, and $\mathbb{E}\left[\psi(\boldsymbol{C}^\star)\right] \ge \frac{2}{9} - \frac{1}{2\sqrt{n}}$.

For sequence **2** we set $y_t^\star = 1$ in the first half of the sequence and $y_t^\star = 0$ in the second half of the sequence. We get $C_{11}^\star := Y_1/n$ and $C_{00}^\star = Y_2/n$, where $Y_1 \sim \text{Binomial}(n/2, 2/3)$, $Y_2 \sim \text{Binomial}(n/2, 2/3)$ with $\mathbb{E}[Y_1] = \mathbb{E}[Y_2] = \frac{n}{3}$. Proceeding similarly as in the previous case, we get $\mathbb{E}\left[\psi(\boldsymbol{C}^\star)\right] \ge \frac{1}{3} - \frac{1}{2\sqrt{n}}$.

Using (22) and (21), the regret of the algorithm can be lower-bounded for the sequence **1** and **2**, respectively, as:

$$\begin{aligned}
R_{\mathbf{1}} &\ge \frac{2}{9} - \min\left\{\frac{2}{9}, \frac{2 - S_1}{6}\right\} - \frac{1}{2\sqrt{n}} = \max\left\{0, \frac{S_1}{6} - \frac{1}{9}\right\} - \frac{1}{2\sqrt{n}}, \\
R_{\mathbf{2}} &\ge \frac{1}{3} - \frac{1}{6}(1 + S_1) - \frac{1}{2\sqrt{n}} = \frac{1 - S_1}{6} - \frac{1}{2\sqrt{n}}.
\end{aligned}$$

Now, the crucial point is that the algorithm *cannot* tune S_1 separately for both cases, as it does not know which of the two sequences will occur. Since the adversary will select a sequence which incurs more regret to the algorithm, the best choice of S_1 is to make the regret (bound) equal in both cases. Clearly, the algorithm will choose $S_1 \ge \frac{2}{3}$, as any smaller value does not change the regret bound for sequence **1** (it is already zero), while it only increases the regret bound for sequence **2**. For $S_1 \ge \frac{2}{3}$, we can equalize the bounds in both cases by setting:

$$\frac{S_1}{6} - \frac{1}{9} = \frac{1 - S_1}{6} \quad \Rightarrow \quad S_1 = \frac{5}{6},$$

which gives the worse-case regret at least $\frac{1}{36} - \frac{1}{2\sqrt{n}}$. Thus, no matter how large n is, the regret does not decrease to zero.

D. OMMA($\widehat{\eta}$) algorithm

In this section, we prove a regret bound for OMMA($\widehat{\eta}$) algorithm defined as Algorithm 2, in the case when the CPE is exact, that is $\widehat{\boldsymbol{\eta}}_t \equiv \boldsymbol{\eta}$ for all t. The bound is of the same order as that of OMMA in Theorem 4.3 under the same assumptions. We assume familiarity of the reader with the proof of Theorem 4.3 in Section B.1.

The algorithm updates its parameter matrix according to

$$\boldsymbol{C}_t = \frac{t-1}{t}\boldsymbol{C}_{t-1} + \frac{1}{t}\boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t),$$

with prediction give by

$$\widehat{\boldsymbol{y}}_t = \operatorname*{argmax}_{\widehat{\boldsymbol{y}}} \nabla\psi(\boldsymbol{C}_{t-1}) \cdot \boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}), \tag{23}$$

where $\boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t) = \mathbb{E}_{\boldsymbol{y}_t \sim \boldsymbol{\eta}_t}\left[\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)\right]$, and we abbreviate $\boldsymbol{\eta}_t = \boldsymbol{\eta}_t(\boldsymbol{x}_t)$. Note that neither algorithm's predictions $\widehat{\boldsymbol{y}}_t$ nor its parameters $\boldsymbol{C}_t$ depend on the labels. We remind that $\boldsymbol{C}_n = \boldsymbol{C}(\boldsymbol{\eta}^n, \widehat{\boldsymbol{y}}^n)$ *does not correspond* to the confusion matrix of the

algorithm $\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)$, as it does not contain labels. We can however interpret $\boldsymbol{C}_n$ as the *expected algorithm's confusion matrix* $\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)$ with labels $\boldsymbol{y}_t$ drawn from $\mathbb{P}(\cdot|\boldsymbol{x}_t)$, as:

$$\boldsymbol{C}_n = \sum_{t=1}^n \boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t) = \sum_{t=1}^n \mathbb{E}_{\boldsymbol{y}_t|\boldsymbol{x}_t}[\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)] = \mathbb{E}_{\boldsymbol{y}^n|\boldsymbol{x}^n}\left[\sum_{t=1}^n \boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)\right] = \mathbb{E}_{\boldsymbol{y}^n|\boldsymbol{x}^n}\left[\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)\right].$$

Using the smoothness of ψ, for any $\boldsymbol{y}^n$, we have (Bubeck, 2015)

$$\psi(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)) \geq \psi(\boldsymbol{C}_n) + \nabla\psi(\boldsymbol{C}_n)\cdot(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n) - \boldsymbol{C}_n) - \frac{M}{2}\|\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n) - \psi(\boldsymbol{C}_n)\|^2,$$

By taking expectation over $\boldsymbol{y}^n|\boldsymbol{x}^n$ on both sides and using the fact proven above that $\boldsymbol{C}_n = \mathbb{E}_{\boldsymbol{y}^n|\boldsymbol{x}^n}\left[\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)\right]$, the gradient term on the right-hand side disappears, while the quadratic term becomes the total *variance* of $\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)$:

$$\mathbb{E}_{\boldsymbol{y}^n|\boldsymbol{x}^n}[\psi(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n))] \geq \psi(\boldsymbol{C}_n) - \frac{L}{2}\mathrm{Var}_{\boldsymbol{y}^n|\boldsymbol{x}^n}\left(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)\right), \tag{24}$$

where we define $\mathrm{Var}(\boldsymbol{C}) = \mathbb{E}[\|\boldsymbol{C} - \mathbb{E}[\boldsymbol{C}]\|^2]$ to be the total variance of all entries of $\boldsymbol{C}$. Since $\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n) = \frac{1}{n}\sum_{t=1}^n \boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)$, with each $\boldsymbol{y}_t$ independent of each other we have, from the scaling property of variance and its additivity for independent random variables,

$$\mathrm{Var}_{\boldsymbol{y}^n|\boldsymbol{x}^n}\left(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)\right) = \frac{1}{n^2}\sum_{t=1}^n \mathrm{Var}_{\boldsymbol{y}^t|\boldsymbol{x}^t}\left(\boldsymbol{C}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)\right) \leq \frac{2a}{n},$$

where we bounded each term in the sum by $\max_{\boldsymbol{C}_1, \boldsymbol{C}_2 \in \mathcal{C}} \|\boldsymbol{C}_1 - \boldsymbol{C}_2\|^2 \leq 2a$, where $a = 1$ for multi-class classification and $a = m$ for multi-label classification (see the proof of Theorem 4.3 in Appendix B.1). Using the bound on the variance in (24) and taking expectation with respect to $\boldsymbol{x}^n$ on both sides gives

$$\mathbb{E}[\psi(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n))] \geq \mathbb{E}_{\boldsymbol{x}^n}[\psi(\boldsymbol{C}_n)] - \frac{Ma}{n} \tag{25}$$

Let $\boldsymbol{h}^\star$ be the optimal classifier and let

$$\boldsymbol{C}^\star = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}\left[\boldsymbol{C}(\boldsymbol{y}, \boldsymbol{h}^\star(\boldsymbol{x}))\right] = \mathbb{E}_{\boldsymbol{x}}\left[\boldsymbol{C}(\boldsymbol{\eta}(\boldsymbol{x}), \boldsymbol{h}^\star(\boldsymbol{x}))\right] \tag{26}$$

be its population confusion matrix. From the concavity of ψ, $\mathbb{E}\left[\psi(\boldsymbol{C}(\boldsymbol{y}^n, \boldsymbol{h}^\star(\boldsymbol{x}^n)))\right] \leq \psi\left(\mathbb{E}\left[\boldsymbol{C}(\boldsymbol{y}^n, \boldsymbol{h}^\star(\boldsymbol{x}^n))\right]\right) = \psi\left(\boldsymbol{C}^\star\right)$, which, together with (25) gives an upper bound on the regret:

$$R_n = \mathbb{E}\left[\psi(\boldsymbol{C}(\boldsymbol{y}^n, \boldsymbol{h}^\star(\boldsymbol{x}^n)))\right] - \mathbb{E}[\psi(\boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n))] \leq \underbrace{\psi(\boldsymbol{C}^\star) - \mathbb{E}_{\boldsymbol{x}^n}[\psi(\boldsymbol{C}_n)]}_{=:Q_n} + \frac{Ma}{n}. \tag{27}$$

The rest of the proof derives a bound on Q_n. From the concavity of the utility,

$$\psi(\boldsymbol{C}^\star) \leq \psi(\boldsymbol{C}_{t-1}) + \nabla\psi(\boldsymbol{C}_{t-1})\cdot(\boldsymbol{C}^\star - \boldsymbol{C}_{t-1}),$$

or, after rearranging,

$$\nabla\psi(\boldsymbol{C}_{t-1})\cdot(\boldsymbol{C}_{t-1} - \boldsymbol{C}^\star) \leq \psi(\boldsymbol{C}_{t-1}) - \psi(\boldsymbol{C}^\star). \tag{28}$$

Using the smoothness of ψ,

$$\begin{aligned}
\psi(\boldsymbol{C}_t) &\geq \psi(\boldsymbol{C}_{t-1}) + \nabla\psi(\boldsymbol{C}_{t-1})\cdot(\boldsymbol{C}_t - \boldsymbol{C}_{t-1}) - \frac{M}{2}\|\boldsymbol{C}_t - \boldsymbol{C}_{t-1}\|_1^2 \\
&= \psi(\boldsymbol{C}_{t-1}) + \frac{1}{t}\nabla\psi(\boldsymbol{C}_{t-1})\cdot(\boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t) - \boldsymbol{C}_{t-1}) - \frac{M}{2t^2}\|\boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t) - \boldsymbol{C}_{t-1}\|^2 \\
&\geq \psi(\boldsymbol{C}_{t-1}) + \frac{1}{t}\nabla\psi(\boldsymbol{C}_{t-1})\cdot(\boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t) - \boldsymbol{C}_{t-1}) - \frac{Ma}{2t^2},
\end{aligned} \tag{29}$$

where in the first equality we used

$$\boldsymbol{C}_t = \frac{t-1}{t}\boldsymbol{C}_{t-1} + \frac{1}{t}\boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t) = \boldsymbol{C}_{t-1} + \frac{1}{t}(\boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t) - \boldsymbol{C}_{t-1}),$$

while in the second inequality we bounded $\|\boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t) - \boldsymbol{C}_{t-1}\|^2 \leq \max_{\boldsymbol{C}_1, \boldsymbol{C}_2 \in \mathcal{C}} \|\boldsymbol{C}_1 - \boldsymbol{C}_2\|^2 \leq 2a$. Multiplying both sides of (29) by (-1) and adding $\psi(\boldsymbol{C}^\star)$ gives:

$$\begin{aligned}
\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_t) &\leq \psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_{t-1}) + \frac{1}{t}\nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}_{t-1} - \boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t)) + \frac{Ma}{t^2} \\
&= \psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_{t-1}) + \frac{1}{t}\nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}_{t-1} - \boldsymbol{C}^\star) + \frac{1}{t}\nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}^\star - \boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t)) + \frac{Ma}{t^2} \\
&\leq \frac{t-1}{t}\left(\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_{t-1})\right) + \frac{1}{t}\nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}^\star - \boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t)) + \frac{Ma}{t^2}
\end{aligned}$$

where in the last inequality we used (28). Multiplying both sides by t and denoting $A_t = t(\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_t))$, we get

$$A_t \leq A_{t-1} + \nabla\psi(\boldsymbol{C}_{t-1}) \cdot (\boldsymbol{C}^\star - \boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t)) + \frac{Ma}{t}.$$

Taking expectation with respect to $\boldsymbol{x}_t$ (conditioned on $\boldsymbol{x}^{t-1}$) on both sides and using the fact than $\boldsymbol{C}_{t-1}, \boldsymbol{C}^\star, A_{t-1}$ do not depend on $\boldsymbol{x}_t$,

$$\mathbb{E}_{\boldsymbol{x}_t}[A_t] \leq A_{t-1} + \underbrace{\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \boldsymbol{C}^\star - \nabla\psi(\boldsymbol{C}_{t-1}) \cdot \mathbb{E}_{\boldsymbol{x}_t}\left[\boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t)\right]}_{B_t} + \frac{Ma}{t}. \tag{30}$$

We show that B_t is nonpositive. Using (26),

$$\begin{aligned}
\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \boldsymbol{C}^\star &= \nabla\psi(\boldsymbol{C}_{t-1}) \cdot \mathbb{E}_{\boldsymbol{x}}\left[\boldsymbol{C}(\boldsymbol{\eta}(\boldsymbol{x}), \boldsymbol{h}^\star(\boldsymbol{x}))\right] \\
(\boldsymbol{x} \text{ has the same distr. as } \boldsymbol{x}_t) \quad &= \nabla\psi(\boldsymbol{C}_{t-1}) \cdot \mathbb{E}_{\boldsymbol{x}_t}\left[\boldsymbol{C}(\boldsymbol{\eta}_t, \boldsymbol{h}^\star(\boldsymbol{x}_t))\right] \\
&= \mathbb{E}_{\boldsymbol{x}_t}\left[\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \boldsymbol{C}(\boldsymbol{\eta}_t, \boldsymbol{h}^\star(\boldsymbol{x}_t))\right] \\
(\text{from (23)}) \quad &\leq \mathbb{E}_{\boldsymbol{x}_t}\left[\nabla\psi(\boldsymbol{C}_{t-1}) \cdot \boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t)\right] \\
&= \nabla\psi(\boldsymbol{C}_{t-1}) \cdot \mathbb{E}_{\boldsymbol{x}_t}\left[\boldsymbol{C}(\boldsymbol{\eta}_t, \widehat{\boldsymbol{y}}_t)\right],
\end{aligned}$$

which implies $B_t \leq 0$. Using this fact and taking the expectation on both sides of (30) with respect to all the past $\boldsymbol{x}^{t-1}$ gives:

$$\mathbb{E}[A_t] \leq \mathbb{E}[A_{t-1}] + \frac{Ma}{t}.$$

Taking $t = n$, expanding the recursion down to $A_0 = 0$, and bounding $\sum_{t=1}^n \frac{1}{t} \leq 1 + \ln n$ as in the Appendix B.1 gives $\mathbb{E}[A_n] \leq Ma(1 + \ln n)$. Since $\mathbb{E}[A_n] = nQ_n$ as defined in (27), we have $Q_n \leq \frac{Ma(1+\ln n)}{n}$ and (27) gives:

$$R_n \leq \frac{Ma(2 + \ln n)}{n}.$$

E. An alternative regret bound for OMMA with $\boldsymbol{p}$-dependent Lipschitzness and smoothness constants

As mentioned in Section B, some of the utilities in Table 1, or their micro- and macro-extensions in multi-label classification, are not globally Lipschitz and smooth with fixed constants L and M over all admissible confusion matrices. One can, however, make them smooth and Lipschitz in a slightly narrower sense, if one allows these constants to depend on the *label frequencies* in the confusion matrix, that is the averaged/expected rate of positive labels per class, which leads to a so called $\boldsymbol{p}$-Lipschitnzess property (Dembczyński et al., 2017), as well as $\boldsymbol{p}$-smoothness.

To make it formal, define a label frequency vector $\boldsymbol{p}(\boldsymbol{C}) \in [0,1]^m$ for *multi-class* confusion matrix $\boldsymbol{C} \in [0,1]^{m \times m}$ as $p_k(\boldsymbol{C}) = \frac{1}{m}\sum_{\ell=1}^m C_{k\ell}$, while for *multi-label* confusion matrix $\boldsymbol{C} \in [0,1]^{m \times 2 \times 2}$ as $p_k(\boldsymbol{C}) = C_{k,11} + C_{k,10}$. Note if $\boldsymbol{C}$ is computed over some data distribution $\mathbb{P}$, $p_k(\boldsymbol{C}) = \mathbb{P}(y_k = 1)$ is a probability of class k in the multi-class case, or a marginal probability of label k in the multi-label case. For empirical confusion matrix $\boldsymbol{C} = \boldsymbol{C}(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}^n)$, $\boldsymbol{p}(\boldsymbol{C})$ is a vector of average counts, $\boldsymbol{p}(\boldsymbol{C}) = \overline{\boldsymbol{y}}_n := \frac{1}{n}\sum_{i=1}^n \boldsymbol{y}^i$. Note that the label frequency vector is independent on the predictions and only depends on the distribution/sequence of labels.

We say utility ψ is $\boldsymbol{p}$-Lipschitz if for any $\boldsymbol{C}, \boldsymbol{C}'$, $|\psi(\boldsymbol{C}) - \psi(\boldsymbol{C}')| \leq L(\boldsymbol{p}(\boldsymbol{C}))\|\boldsymbol{C} - \boldsymbol{C}'\|$, that is, the Lipschitz constant may depend on $\boldsymbol{p}(\boldsymbol{C})$. Similarly, ψ is $\boldsymbol{p}$-smooth if $\|\nabla\psi(\boldsymbol{C}) - \nabla\psi(\boldsymbol{C}')| \leq M(\boldsymbol{p}(\boldsymbol{C}))\|\boldsymbol{C} - \boldsymbol{C}'\|$. Generally these functions

$L(\boldsymbol{p})$ and $M(\boldsymbol{p})$ will degrade (increase) when any of coordinates of $\boldsymbol{p}$ gets close to zero. Therefore, we define:

$$L_\gamma = \max_{\boldsymbol{p}:\ \forall k, p_k \geq \gamma} L(\boldsymbol{p}),$$

as the largest value of $L(\boldsymbol{p})$ when all elements of $\boldsymbol{p}$ are separated from 0 by at least γ; M_γ is defined analogously. In what follows, we assume that L_γ and M_γ are well-controlled when γ is separated from 0.

Given a data distribution $\mathbb{P}$, let $\boldsymbol{p}^\star = \mathbb{E}[\boldsymbol{y}]$ denote a vector of label (marginal) probabilities, and we let $\gamma^\star = \min_k p_k^\star$.

Theorem E.1. *Let ψ be concave, $\boldsymbol{p}$-Lipschitz and $\boldsymbol{p}$-smooth. Assume $\psi(\boldsymbol{C}) \in [0,1]$ for all $\boldsymbol{C} \in \mathcal{C}$. The regret of OMMA algorithm is bounded by:*

$$R_n \leq O\left(\sqrt{\frac{\log n}{n}}\right) + 2bL_{\frac{\gamma^\star}{4}} e_n$$

where $e_n = \frac{1}{n}\sum_{t=1}^n \mathbb{E}[\|\boldsymbol{\eta}(\boldsymbol{x}_t) - \widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|]$ is the average estimation error of the CPE, while the constants under $O(\cdot)$ depend on m, $L_{\frac{\gamma^\star}{4}}$ and $M_{\frac{\gamma^\star}{4}}$.

Proof. We assume the reader is familiar with the proof of Theorem 4.3 in Appendix B.1, to which we will refer at multiple occasions. We start with recalling that the regret can be upper-bounded by $R_n \leq \psi(\boldsymbol{C}^\star) - \mathbb{E}[\psi(\boldsymbol{C}_n)]$, where $\boldsymbol{C}^\star = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}[\boldsymbol{C}(\boldsymbol{y}, \boldsymbol{h}^\star(\boldsymbol{x}))]$ is the population confusion matrix of the optimal classifier $\boldsymbol{h}^\star$.

We say that a sequence of labels $\boldsymbol{y}^n$ is (n_0, β)-*nice* if for all $t \geq n_0$, the running averages are all at least β, that is $\overline{y}_{tk} \geq \beta$ for all $t \geq n_0$ and all $k = 1, \ldots, m$.

Lemma E.2. *Let $\boldsymbol{y}^n$ be drawn i.i.d. from $\mathbb{P}$. Then $\boldsymbol{y}^n$ is $(n_0, \frac{\gamma^\star}{4})$-nice with probability at least $1 - mn_0 e^{-\gamma^{\star 2} n_0/2}$.*

Proof. (of Lemma E.2) Split the sequence into J intervals of length at least $n_J = \lfloor \frac{n}{J} \rfloor$ each (for the sake of readability, assume n is divisible by J). Let

$$\boldsymbol{s}_j = \frac{1}{n_J} \sum_{t=1+(j-1)n_J}^{jn_J} \boldsymbol{y}_t$$

be the vector of average label counts within the j-th interval, $j \in \{1, \ldots, J\}$. Note that $\mathbb{E}[\boldsymbol{s}_j] = \boldsymbol{p}^\star$. Using Hoeffding's inequality (Hoeffding, 1963; Cesa-Bianchi & Lugosi, 2006) and a union bound over coordinates/labels $k = 1, \ldots, m$, we have $\min_k\{s_{j,k} - p_k^\star\} \geq -\beta$ with probability at least $1 - me^{-2\beta^2 n_J}$. Next, taking union bound over the intervals, $\min_{j,k}\{s_{j,k} - p_k^\star\} \geq -\beta$ with probability at least $1 - \delta$ where $\delta = n_J m e^{-2\beta^2 n_J}$. Thus, with probability at least $1 - \delta$, for every interval j and every label k simultaneously,

$$\sum_{i=1+(j-1)n_J}^{jn_J} y_{ik} \geq n_J(p_k^\star - \beta)\,. \tag{31}$$

Take any iteration t in the j-th interval with $j \geq 2$, that is $t = (j-1)n_J + r$ for $1 \leq r \leq n_J$. We have, with probability at least $1 - \delta$:

$$\overline{y}_{tk} = \frac{1}{t}\sum_{i=1}^t y_{ik} \geq \frac{1}{t}\sum_{i=1}^{(j-1)n_J} y_{ik} \geq \frac{(j-1)n_J}{t}(p_k^\star - \beta)\,.$$

Plugging in $t = (j-1)n_J + r$,

$$\overline{y}_{tk} \geq \frac{(j-1)n_J}{(j-1)n_J + r}(p_k^\star - \beta) \geq \frac{(j-1)n_J}{jn_J}(p_k^\star - \beta) = \frac{(j-1)}{j}(p_k^\star - \beta) \geq \frac{1}{2}(p_k^\star - \beta),$$

where we used the fact that $j \geq 2$ for the last inequality. Thus, for every t outside the first interval, the running average for each label is at least $\frac{1}{2}(p_k^\star - \beta)$. Taking $\beta = \frac{\gamma^\star}{2}$, using the fact that $p_k^\star \geq \gamma^\star$, we see that $\overline{y}_{tk} \geq \frac{\gamma^\star}{4}$ for all $t > n_J =: n_0$ and all k with probability at least $1 - mn_0 e^{-\gamma^{\star 2} n_0/2}$. □

We use Lemma E.2 as follows: with probability at least $1-\delta_1$ with $\delta_1 := mn_0 e^{-\gamma^{\star 2} n_0/2}$, starting from $t \geq n_0+1$, $\overline{y}_{tk} \geq \frac{\gamma^\star}{4}$ for all t and k, and thus we have $\boldsymbol{p}$-Lipschitzness and $\boldsymbol{p}$-smoothness well-controlled for ψ evaluated at $\boldsymbol{C}_{t-1}$, that is for $t \geq n_0+1$, all constants used in the proof of Theorem 4.3 are at most $L_{\frac{\gamma^\star}{4}}$ and $M_{\frac{\gamma^\star}{4}}$, which we denote simply by L and M.

We will need one more probabilistic bound. In Appendix B.1 we have shown that

$$\mathbb{E}_{(\boldsymbol{x}_t,\boldsymbol{y}_t)}\left[\nabla\psi(\boldsymbol{C}_{t-1})\cdot(\boldsymbol{C}^\star - \boldsymbol{C}(\boldsymbol{y}_t,\widehat{\boldsymbol{y}}_t))\right] \leq 2bL\,\mathbb{E}_{\boldsymbol{x}_t}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t)-\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right].$$

Define $V_t = \nabla\psi(\boldsymbol{C}_{t-1})\cdot(\boldsymbol{C}^\star - \boldsymbol{C}(\boldsymbol{y}_t,\widehat{\boldsymbol{y}}_t)) - 2bL\,\mathbb{E}_{\boldsymbol{x}_t}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t)-\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right]$ and note that the above inequality implies $\mathbb{E}_{(\boldsymbol{x}_t,\boldsymbol{y}_t)}[V_t] \leq 0$. We also have

$$|V_t - \mathbb{E}_{(\boldsymbol{x}_t,\boldsymbol{y}_t)}[V_t]| \leq |V_t| + |\,\mathbb{E}_{(\boldsymbol{x}_t,\boldsymbol{y}_t)}[V_t]| \leq 2\|\nabla\psi(\boldsymbol{C}_{t-1})\| \max_{\boldsymbol{C},\boldsymbol{C}'}\|\boldsymbol{C}-\boldsymbol{C}'\| + 4bL\max_{\boldsymbol{\eta},\boldsymbol{\eta}'}\|\boldsymbol{\eta}-\boldsymbol{\eta}'\|,$$

where we used Cauchy-Schwarz inequality. We note that $\max_{\boldsymbol{C},\boldsymbol{C}'}\|\boldsymbol{C}-\boldsymbol{C}'\| \leq 2a$ and $\max_{\boldsymbol{\eta},\boldsymbol{\eta}'}\|\boldsymbol{\eta}-\boldsymbol{\eta}'\| \leq c$ where $c=2$ for multi-class classification and $c=m$ for multi-label classification. Since $b=1$ for multi-class and $b=\sqrt{2}$ for multi-label classification, denoting $G_{t-1} = \|\nabla\psi(\boldsymbol{C}_{t-1})\|$ we conclude that $|V_t - \mathbb{E}_{(\boldsymbol{x}_t,\boldsymbol{y}_t)}[V_t]| \leq 4aG_{t-1} + 8aL$ (note that $a=1$ for multi-class and $a=m$ for multi-label case). We define $U_t = V_t - \mathbb{E}_{(\boldsymbol{x}_t,\boldsymbol{y}_t)}[V_t]$ if $G_{t-1} \leq L$ and otherwise $U_t = 0$. It follows that $|U_t| \leq 4aL + 8aL \leq 12aL$, and U_t forms a martingale difference sequence (see Cesa-Bianchi & Lugosi (2006), Appendix A.1.3). Thus, by Hoeffding-Azuma inequality, $\frac{1}{n-m_0}\sum_{t=m_0+1}^n U_t \leq \beta$ with probability at most $1-\delta_2 = 1-\exp\left\{-\frac{2\beta^2(n-m_0)}{144a^2L^2}\right\}$. Using it on a nice sequence (when $G_{t-1} \leq L$ for all $t \geq n_0+1$), we conclude (by the union bound) that with probability $1-\delta_1-\delta_2$,

$$\sum_{t>n_0} V_t \leq \sum_{t>n_0}(V_t - \mathbb{E}_{(\boldsymbol{x}_t,\boldsymbol{y}_t)}[V_t]) = \sum_{t>n_0} U_t \leq \beta(n-n_0).$$

Having control over L and M, and over the term $\sum_{t>n_0} V_t$, we use the entire machinery of the proof of Theorem 4.3 in the Appendix B.1 (with expectation $\mathbb{E}_{(\boldsymbol{x}_t,\boldsymbol{y}_t)}[\cdot]$ replaced by just proven high-probability bound) to conclude that with probability at least $1-\delta_0-\delta_1$,

$$\begin{aligned}
A_n &\leq A_{n_0} + 2bL\sum_{t=n_0+1}^n \mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t)-\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right] + \beta(n-n_0) + \sum_{t=n_0+1}^n \frac{Ma}{t} \\
&\leq A_{n_0} + 2bL\sum_{t=1}^n \mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t)-\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right] + \beta(n-n_0) + Ma(1+\ln n)
\end{aligned}$$

where $A_t = t(\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_t))$. This means that with probability at least $1-\delta_0-\delta_1$,

$$\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_n) \leq \frac{2n_0}{n} + \frac{\beta(n-n_0)}{n} + \frac{Ma(1+\ln n)}{n} + \frac{2bL}{n}\sum_{t=1}^n \mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t)-\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right],$$

where we bounded $|\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_n)| \leq 1$ using assumption $\psi(\boldsymbol{C}) \in [0,1]$ for all $\boldsymbol{C} \in \mathcal{C}$. Let E be the event that the sequence is nice and Hoeffding-Azuma inequality holds, which happens with probability at least $1-\delta_0-\delta_1$. We have

$$\begin{aligned}
\mathbb{E}[\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_n)] &= \mathbb{E}[\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_n)|E]\,\mathbb{P}(E) + \underbrace{\mathbb{E}[\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_n)|\neg E]}_{\leq 1}\,\mathbb{P}(\neg E) \\
&\leq \mathbb{E}[\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_n)|E] + \mathbb{P}(\neg E) \\
&\leq \frac{2n_0}{n} + \frac{\beta(n-n_0)}{n} + \frac{Ma(1+\ln n)}{n} + \frac{2bL}{n}\sum_{t=1}^n \mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t)-\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right] + \delta_0 + \delta_1 \\
&= \frac{2n_0}{n} + \frac{\beta(n-n_0)}{n} + \frac{Ma(1+\ln n)}{n} + \frac{2bL}{n}\sum_{t=1}^n \mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t)-\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right] + mn_0 e^{-\frac{\gamma^{\star 2} n_0}{2}} + e^{-\frac{2\beta^2(n-m_0)}{144a^2L^2}}.
\end{aligned}$$

Finally, we tune the remaining parameters β and n_0. We set $n_0 = \sqrt{n}$, and $\beta = \frac{12aL\sqrt{\ln n}}{\sqrt{2(n-n_0)}}$ (in order to have $e^{-\frac{2\beta^2(n-m_0)}{144a^2L^2}} = e^{-\ln n} = \frac{1}{n}$). Bounding $\sqrt{n-n_0} \leq \sqrt{n}$, and denoting the estimation error by $e_n = \frac{1}{n}\sum_{t=1}^n \mathbb{E}\left[\|\boldsymbol{\eta}(\boldsymbol{x}_t)-\widehat{\boldsymbol{\eta}}_t(\boldsymbol{x}_t)\|\right]$, we

finally get:

$$\mathbb{E}[\psi(\boldsymbol{C}^\star) - \psi(\boldsymbol{C}_n)] \leq \frac{2 + 12aL_{\frac{\gamma^\star}{4}}\sqrt{\ln n}}{\sqrt{n}} + \frac{M_{\frac{\gamma^\star}{4}}a(1+\ln n)}{n} + \frac{2bL_{\frac{\gamma^\star}{4}}}{n} + m\sqrt{n}e^{-\frac{\gamma^{\star 2}\sqrt{n}}{2}} + \frac{1}{n} + 2bL_{\frac{\gamma^\star}{4}}e_n$$
$$= O\left(\sqrt{\frac{\log n}{n}}\right) + 2bL_{\frac{\gamma^\star}{4}}e_n,$$

which was to be shown. □

F. Details of experiments setup

In multi-label experiments with fixed CPE, we use LIBLINEAR model with L_2-regularized logistic loss (Fan et al., 2008). For larger problems (RCV1X and AmazonCat), we made it sparse by truncating all the weights whose absolute value is above the threshold of 0.01 as introduced in (Babbar & Schölkopf, 2017) to significantly reduce memory requirements and inference time, then we made the probability estimates $\hat{\eta}$ sparse for these datasets by only keeping top 100 values for each sample, similarly to (Schultheis et al., 2023). In multi-class experiments with both fixed and online CPE, we use PyTorch (Paszke et al., 2019) linear model with L_2-regularized binary cross entropy or cross entropy loss, optimized using ADAM (Kingma & Ba, 2015). We tuned the learning rate and regularization strength and used the best-obtained model.

We evaluate all the algorithms using 5 random sequences of test sets according to Figure 1. All algorithms are tested on the same 5 sequences for fair comparison. All the methods were implemented in Python. For RCV1X and AmazonCat, we implemented all the algorithms (OFO, Greedy, Frank-Wolfe, and OMMA) to allow them to leverage the sparsity for a more fair computational cost comparison.

All the experiments were conducted on a workstation with 64 GB of RAM.

G. Extended results with the fixed CPE

In this section, we include extended results of our empirical experiments. In Table 4, we report the results presented in the main paper but also results for the Greedy algorithm that only uses $\widehat{\boldsymbol{\eta}}_t$ (Greedy($\hat{\eta}$)) as initially proposed by Schultheis et al. (2023) and Offline Frank-Wolfe (Offline-FW) algorithm. We additionally report standard deviations of the results as well as mean running times. In Figure 4, we present running performance for all datasets and metrics presented in Table 4. In Figure 5, we present the effect of selecting different regularization constants λ for a selected subset of datasets (without RCV1X and AmazonCat) and all metrics presented in Table 4.

In most cases, Greedy($\hat{\eta}$) performs similarly to OMMA($\widehat{\eta}$), as in the case of standard Greedy and OMMA performing similarly. Offline-FW obtains mixed results, sometimes much better and sometimes much worse than online algorithms. Similarly to Online-FW, it is not performing that well on macro-averaged Precision@k.

We note that the running time comparison favors the Online-FW algorithm, as its updates are better-optimized thanks to vectorization and less frequent than OMMA's and other online methods, which update after every step. We expect OMMA to be much faster if implemented more efficiently.

H. Results with the online CPE

In the main paper and the previous section, we presented the results using fixed CPE that was trained on a separate, much larger training set. This approach of being agnostic to the CPE model is commonly used in the literature on learning complex performance metrics (Koyejo et al., 2014; Busa-Fekete et al., 2015; Dembczynski et al., 2017; Liu et al., 2018; Narasimhan et al., 2022; Schultheis et al., 2024). However, it is possible to train CPE along with using online optimization methods.

In this section, we additionally present the results with such online CPE conducted on four datasets (YouTube, Eurlex-LexGlue, Mediamill, and Flickr). In this experiment, the CPE starts with randomly initialized parameters and updates them after each observed instance in the test sequence. In Table 6, we report the final results on the sequence, and in Figure 6, we present running performance for the same set of datasets and metrics. As expected, the average performance over the sequence is lower than in the case of fixed CPE, but the proposed OMMA performance is, in most cases, on par or better than the other algorithms.

Table 4. Predictive performance and running times of the different online algorithms on *multi-label* problems, averaged over 5 runs and reported with standard deviations (±)s. The best result on each metric is in **bold**, the second best is in *italic*. We additionally report basic statistics of the benchmarks: number of classes/labels and instances in the test sequence. × – means that the algorithm does not support the optimization of that metric. The inference time of Offline-FW was not measured.

Method	Micro F1		Macro F1		Macro F1@3		Macro Recall@3		Macro Precision@3		Macro G-mean		Macro H-mean	
	(%)	time (s)	(%)	time (s)	(%)	time (s)	(%)	time (s)	(%)	time (s)	(%)	time (s)	(%)	time (s)
	YOUTUBE ($m = 46, n = 7926$)													
Top-k / $\hat{\eta} > 0.5$	31.20± 0.00	0.13± 0.01	22.74± 0.00	0.13± 0.01	30.99± 0.00	0.08± 0.00	42.13± 0.00	0.09± 0.02	26.39± 0.00	0.08± 0.00	32.82± 0.00	0.13± 0.01	24.46± 0.00	0.13± 0.01
OFO	43.71± 0.05	0.33± 0.01	36.15± 0.17	0.58± 0.10	×	×	×	×	×	×	×	×	×	×
Greedy	×	×	36.32± 0.15	1.35± 0.08	34.72± 0.17	1.27± 0.02	45.84± 0.09	1.22± 0.02	*67.18± 1.56*	1.22± 0.03	*77.98± 0.05*	1.44± 0.04	**77.93± 0.07**	1.57± 0.16
Greedy($\hat{\eta}$)	×	×	**36.47± 0.07**	1.30± 0.04	*35.38± 0.05*	1.27± 0.02	45.84± 0.09	1.23± 0.04	*67.18± 1.56*	1.23± 0.04	**77.99± 0.02**	1.42± 0.05	*77.92± 0.03*	1.51± 0.08
Online-FW	43.67± 0.05	6.80± 0.55	36.00± 0.11	10.92± 0.21	34.38± 0.25	10.76± 0.42	45.83± 0.10	5.34± 0.07	38.92± 1.56	6.54± 0.39	77.96± 0.05	6.60± 0.26	77.91± 0.04	11.46± 0.57
Online-FW($\hat{\eta}$)	43.69± 0.02	10.02± 0.14	**36.47± 0.04**	11.57± 0.19	**35.43± 0.08**	29.99± 0.39	**45.89± 0.05**	6.39± 0.07	50.12± 1.95	11.50± 0.06	77.92± 0.03	2.00± 0.01	**77.93± 0.03**	15.52± 1.48
OMMA	**43.73± 0.04**	2.78± 0.18	*36.34± 0.19*	2.84± 0.09	34.81± 0.11	2.80± 0.05	*45.85± 0.09*	2.41± 0.07	**67.74± 1.61**	2.40± 0.08	*77.98± 0.05*	2.92± 0.13	**77.93± 0.07**	3.15± 0.17
OMMA($\hat{\eta}$)	*43.72± 0.03*	2.79± 0.17	**36.47± 0.08**	2.73± 0.10	*35.38± 0.06*	2.72± 0.05	**45.89± 0.04**	2.38± 0.06	65.49± 2.14	2.38± 0.07	*77.98± 0.02*	2.94± 0.15	**77.93± 0.03**	3.15± 0.22
Offline-FW	43.75± 0.01	–	36.55± 0.05	–	35.68± 0.05	–	44.34± 0.00	–	51.51± 5.55	–	78.03± 0.01	–	77.96± 0.01	–
	EURLEX-LEXGLUE ($m = 100, n = 5000$)													
Top-k / $\hat{\eta} > 0.5$	70.99± 0.00	0.08± 0.00	52.43± 0.00	0.08± 0.00	46.35± 0.00	0.06± 0.01	36.67± 0.00	0.05± 0.00	74.70± 0.00	0.05± 0.00	62.33± 0.00	0.08± 0.01	55.95± 0.00	0.08± 0.00
OFO	73.23± 0.01	0.20± 0.01	58.93± 0.22	0.18± 0.01	×	×	×	×	×	×	×	×	×	×
Greedy	×	×	59.83± 0.11	0.84± 0.03	54.19± 0.07	0.81± 0.03	52.67± 0.13	0.76± 0.03	*88.21± 1.24*	0.76± 0.02	89.74± 0.09	0.90± 0.05	89.73± 0.10	0.96± 0.03
Greedy($\hat{\eta}$)	×	×	**59.88± 0.15**	0.84± 0.04	**54.39± 0.15**	0.83± 0.04	52.67± 0.13	0.75± 0.02	*88.21± 1.24*	0.75± 0.03	**89.92± 0.05**	0.90± 0.04	**89.85± 0.05**	0.94± 0.04
Online-FW	**73.68± 0.04**	3.76± 0.29	59.53± 0.14	7.60± 0.50	54.18± 0.06	8.72± 0.39	52.68± 0.19	3.95± 0.15	58.76± 1.64	8.20± 1.25	89.69± 0.05	3.88± 0.26	89.75± 0.08	7.68± 0.41
Online-FW($\hat{\eta}$)	73.22± 0.01	7.26± 0.11	59.78± 0.11	9.68± 0.13	54.34± 0.15	34.28± 0.40	**53.87± 0.15**	5.40± 0.04	51.37± 0.89	23.53± 0.87	*89.91± 0.02*	1.71± 0.02	*89.84± 0.04*	20.59± 1.95
OMMA	*73.29± 0.02*	1.73± 0.07	*59.85± 0.07*	1.84± 0.11	54.15± 0.11	1.81± 0.07	52.67± 0.13	1.45± 0.05	**88.41± 0.73**	1.44± 0.04	89.74± 0.09	1.88± 0.17	89.73± 0.10	1.97± 0.11
OMMA($\hat{\eta}$)	73.22± 0.01	1.75± 0.09	59.84± 0.12	1.73± 0.08	*54.37± 0.13*	1.73± 0.07	*53.85± 0.15*	1.47± 0.05	84.99± 1.31	1.46± 0.04	**89.92± 0.05**	1.90± 0.18	**89.85± 0.05**	2.08± 0.25
Offline-FW	73.31± 0.01	–	55.82± 0.29	–	50.05± 0.01	–	52.02± 0.00	–	61.11± 3.07	–	89.18± 0.00	–	89.00± 0.01	–
	MEDIAMILL ($m = 101, n = 12914$)													
Top-k / $\hat{\eta} > 0.5$	52.45± 0.00	0.23± 0.02	4.06± 0.00	0.23± 0.01	4.43± 0.00	0.13± 0.00	4.35± 0.00	0.13± 0.00	7.42± 0.00	0.13± 0.00	4.62± 0.00	0.23± 0.02	3.65± 0.00	0.23± 0.01
OFO	*56.99± 0.01*	0.57± 0.01	12.36± 0.03	1.06± 0.03	×	×	×	×	×	×	×	×	×	×
Greedy	×	×	12.43± 0.07	2.85± 0.27	10.29± 0.07	2.18± 0.17	*9.41± 0.32*	1.99± 0.10	*22.54± 1.35*	1.98± 0.11	*65.68± 0.07*	3.08± 0.31	**66.48± 0.41**	3.14± 0.22
Greedy($\hat{\eta}$)	×	×	*14.33± 0.04*	2.32± 0.10	*11.88± 0.14*	2.12± 0.11	*9.41± 0.32*	1.99± 0.11	*22.54± 1.35*	1.96± 0.08	**65.93± 0.17**	3.05± 0.26	65.37± 0.16	3.14± 0.21
Online-FW	56.98± 0.01	8.72± 0.50	12.22± 0.10	11.70± 0.58	10.18± 0.14	9.69± 1.50	9.16± 0.70	3.56± 0.29	12.70± 0.84	9.82± 0.54	64.73± 0.50	6.92± 0.93	*65.68± 0.45*	20.44± 1.45
Online-FW($\hat{\eta}$)	*56.99± 0.00*	11.25± 0.17	*14.33± 0.02*	16.15± 1.44	**11.97± 0.10**	47.57± 0.45	**16.43± 0.08**	8.13± 0.05	17.35± 0.68	27.09± 1.23	65.37± 0.22	2.48± 0.02	65.56± 0.07	49.09± 0.26
OMMA	**57.00± 0.01**	4.56± 0.33	12.39± 0.07	4.51± 0.05	10.25± 0.10	4.56± 0.23	9.39± 0.32	3.83± 0.19	**22.85± 1.11**	3.76± 0.16	65.67± 0.11	4.77± 0.36	**66.48± 0.40**	4.95± 0.23
OMMA($\hat{\eta}$)	*56.99± 0.01*	4.52± 0.25	**14.34± 0.03**	4.75± 0.63	11.87± 0.11	4.45± 0.22	**16.43± 0.07**	3.82± 0.16	18.31± 0.49	3.81± 0.19	**65.93± 0.17**	4.88± 0.37	65.35± 0.16	5.08± 0.35
Offline-FW	56.99± 0.01	–	13.15± 0.14	–	10.39± 0.12	–	16.51± 0.00	–	14.38± 0.99	–	66.08± 0.20	–	65.80± 0.27	–
	FLICKR ($m = 195, n = 24154$)													
Top-k / $\hat{\eta} > 0.5$	29.46± 0.00	0.39± 0.03	18.27± 0.00	0.38± 0.01	26.39± 0.00	0.23± 0.04	38.96± 0.00	0.22± 0.04	21.38± 0.00	0.19± 0.01	27.08± 0.00	0.39± 0.04	20.02± 0.00	0.39± 0.03
OFO	**41.05± 0.02**	0.92± 0.03	30.46± 0.13	0.95± 0.04	×	×	×	×	×	×	×	×	×	×
Greedy	×	×	30.90± 0.10	3.62± 0.22	30.42± 0.03	3.30± 0.16	**46.41± 0.15**	3.03± 0.11	*57.55± 1.29*	3.05± 0.10	83.39± 0.07	5.11± 0.37	**83.37± 0.03**	5.20± 0.31
Greedy($\hat{\eta}$)	×	×	*31.16± 0.06*	3.45± 0.20	**30.63± 0.11**	3.33± 0.13	**46.41± 0.15**	3.06± 0.12	*57.55± 1.29*	3.04± 0.10	**83.42± 0.03**	5.05± 0.37	83.24± 0.03	5.12± 0.29
Online-FW	**41.05± 0.01**	10.82± 1.13	30.60± 0.06	17.19± 0.46	29.58± 0.08	14.36± 0.81	*46.38± 0.16*	7.31± 0.15	28.66± 0.93	10.06± 1.25	83.37± 0.03	12.04± 0.67	*83.28± 0.05*	17.76± 1.18
Online-FW($\hat{\eta}$)	*41.02± 0.02*	14.66± 0.18	**31.17± 0.06**	18.17± 0.24	29.65± 0.21	41.54± 3.51	46.28± 0.08	10.30± 0.09	25.73± 0.82	31.20± 0.83	83.37± 0.02	9.35± 1.03	83.20± 0.03	37.08± 0.84
OMMA	41.01± 0.02	6.94± 0.35	30.90± 0.13	7.14± 0.44	30.39± 0.05	7.06± 0.35	**46.41± 0.15**	5.83± 0.19	**58.35± 0.34**	5.86± 0.17	83.39± 0.07	7.57± 0.67	**83.37± 0.03**	7.91± 0.44
OMMA($\hat{\eta}$)	*41.02± 0.03*	7.05± 0.43	31.15± 0.08	6.97± 0.38	*30.55± 0.09*	7.09± 0.44	46.33± 0.05	5.83± 0.18	55.56± 0.43	5.90± 0.19	*83.41± 0.03*	7.51± 0.58	83.23± 0.04	8.08± 0.61
Offline-FW	40.96± 0.00	–	30.80± 0.04	–	30.26± 0.09	–	46.45± 0.00	–	39.27± 1.14	–	83.44± 0.01	–	83.22± 0.01	–
	RCV1X ($m = 2456, n = 155962$)													
Top-k / $\hat{\eta} > 0.5$	68.57± 0.00	12.72± 2.65	11.29± 0.00	13.48± 4.35	5.34± 0.00	1.44± 0.02	4.59± 0.00	1.45± 0.03	13.24± 0.00	1.44± 0.04	16.01± 0.00	12.01± 2.14	*12.36± 0.00*	15.99± 6.34
OFO	**69.83± 0.00**	27.57± 13.52	20.26± 0.15	76.05± 39.24	×	×	×	×	×	×	×	×	×	×
Greedy	×	×	**20.80± 0.16**	91.71± 34.37	16.01± 0.12	27.04± 0.43	21.20± 0.08	25.58± 0.38	**30.91± 0.37**	25.37± 0.39	**69.07± 0.00**	183.85± 69.54	**67.04± 0.00**	234.79± 106.38
Greedy($\hat{\eta}$)	×	×	20.50± 0.09	57.47± 19.59	**16.09± 0.09**	27.00± 0.42	21.20± 0.08	25.56± 0.42	**30.91± 0.37**	25.20± 0.56	*69.06± 0.00*	192.01± 126.11	**67.04± 0.00**	233.71± 102.62
Online-FW	**69.83± 0.00**	30.37± 1.42	19.82± 0.16	36.40± 0.99	15.33± 0.09	52.14± 1.35	21.09± 0.15	36.73± 0.62	19.88± 0.21	62.42± 6.57	**69.07± 0.00**	30.96± 0.65	**67.04± 0.00**	40.11± 1.61
Online-FW($\hat{\eta}$)	*69.79± 0.01*	46.18± 0.47	20.40± 0.08	63.85± 0.56	15.55± 0.10	220.60± 0.55	*22.21± 0.08*	47.24± 0.49	22.17± 0.34	205.95± 0.77	*69.06± 0.00*	48.66± 0.87	**67.04± 0.00**	68.64± 1.10
OMMA	69.77± 0.00	58.51± 6.52	20.57± 0.16	56.42± 4.68	15.87± 0.11	66.80± 13.58	21.08± 0.10	56.39± 0.94	*30.39± 0.21*	53.22± 8.16	**69.07± 0.00**	63.82± 10.34	**67.04± 0.00**	64.56± 5.64
OMMA($\hat{\eta}$)	69.71± 0.00	59.59± 6.46	*20.71± 0.11*	60.31± 9.50	*16.07± 0.08*	62.89± 15.22	**22.23± 0.09**	56.86± 0.86	30.38± 0.24	49.32± 4.36	*69.06± 0.00*	65.63± 9.65	**67.04± 0.00**	69.29± 8.55
Offline-FW	69.84± 0.00	–	15.08± 0.05	–	11.74± 0.05	–	22.22± 0.00	–	20.98± 0.71	–	69.48± 0.00	–	67.50± 0.00	–
	AMAZONCAT ($m = 13330, n = 306784$)													
Top-k / $\hat{\eta} > 0.5$	67.77± 0.00	52.41± 14.16	28.76± 0.00	60.25± 23.57	14.98± 0.00	3.03± 0.08	11.18± 0.00	2.83± 0.05	30.98± 0.00	2.83± 0.04	*33.93± 0.00*	77.78± 35.26	*30.03± 0.00*	39.14± 6.60
OFO	*70.38± 0.00*	88.22± 33.36	39.60± 0.06	154.54± 62.87	×	×	×	×	×	×	×	×	×	×
Greedy	×	×	44.20± 0.06	175.96± 48.04	43.64± 0.05	52.42± 0.87	57.81± 0.01	49.01± 1.22	**54.00± 0.07**	48.90± 1.10	**80.86± 0.00**	415.92± 182.65	**80.04± 0.00**	449.12± 231.23
Greedy($\hat{\eta}$)	×	×	44.08± 0.04	127.17± 28.98	**46.11± 0.06**	52.50± 0.95	57.81± 0.01	50.08± 0.62	**54.00± 0.07**	48.58± 0.72	**80.86± 0.00**	465.89± 144.03	**80.04± 0.00**	597.36± 209.20
Online-FW	**70.61± 0.01**	44.76± 1.36	42.42± 0.03	60.33± 1.26	40.20± 0.04	69.75± 0.46	57.74± 0.03	73.35± 0.52	40.10± 0.18	79.41± 0.97	**80.86± 0.00**	63.70± 0.94	**80.04± 0.00**	83.86± 1.95
Online-FW($\hat{\eta}$)	70.34± 0.00	88.21± 1.10	*47.32± 0.06*	111.40± 1.59	44.63± 0.05	425.18± 1.53	*58.87± 0.04*	89.94± 0.38	49.68± 0.14	416.26± 1.83	**80.86± 0.00**	97.05± 1.06	**80.04± 0.00**	129.23± 1.45
OMMA	69.90± 0.00	107.61± 5.61	43.04± 0.05	106.79± 5.02	42.14± 0.05	114.83± 11.24	57.80± 0.03	91.40± 1.46	*52.04± 0.15*	92.92± 3.23	**80.86± 0.00**	129.35± 16.42	**80.04± 0.00**	120.60± 5.64
OMMA($\hat{\eta}$)	70.02± 0.00	106.94± 4.27	**47.67± 0.03**	109.30± 6.82	*45.06± 0.05*	111.92± 9.07	**58.89± 0.04**	91.80± 0.94	51.52± 0.13	93.68± 3.58	**80.86± 0.00**	118.54± 6.60	**80.04± 0.00**	121.44± 5.28
Offline-FW	70.43± 0.01	–	38.20± 0.09	–	39.38± 0.06	–	62.39± 0.00	–	42.01± 0.61	–	83.23± 0.00	–	82.58± 0.00	–

Table 5. Predictive performance and running times of the different online algorithms on *multi-class* problems, averaged over 5 runs and reported with standard deviations (±)s. The best result on each metric is in **bold**, the second best is in *italic*. We additionally report basic statistics of the benchmarks: number of classes/labels and instances in the test sequence. × – means that the algorithm does not support the optimization of that metric. The inference time of Offline-FW was not measured.

Method	Macro F1 (%)	Macro F1 time (s)	Macro F1@3 (%)	Macro F1@3 time (s)	Macro Recall@3 (%)	Macro Recall@3 time (s)	Macro Precision@3 (%)	Macro Precision@3 time (s)	Multi-class G-Mean (%)	Multi-class G-Mean time (s)	Multi-class H-Mean (%)	Multi-class H-Mean time (s)	Multi-class Q-Mean (%)	Multi-class Q-Mean time (s)
	NEWS20 ($m = 20, n = 7532$)													
Top-k	**83.37± 0.00**	0.06± 0.00	49.23± 0.00	0.08± 0.00	94.78± 0.00	0.06± 0.00	33.95± 0.00	0.07± 0.01	82.51± 0.00	0.07± 0.02	81.66± 0.00	0.09± 0.01	80.12± 0.00	0.07± 0.02
Greedy	82.44± 0.44	1.30± 0.04	72.69± 0.71	1.27± 0.03	*94.89± 0.06*	1.28± 0.05	**83.89± 0.38**	1.22± 0.07	×	×	×	×	×	×
Greedy($\hat{\eta}$)	*83.08± 0.09*	1.29± 0.03	*73.37± 0.60*	1.27± 0.03	*94.89± 0.06*	1.32± 0.05	**83.89± 0.38**	1.17± 0.01	×	×	×	×	×	×
Online-FW	82.88± 0.05	6.67± 0.31	54.10± 1.09	8.18± 0.24	94.88± 0.05	6.76± 0.24	19.90± 2.95	6.66± 0.46	82.68± 0.15	1.93± 0.42	*82.35± 0.04*	8.93± 1.65	80.95± 0.06	8.17± 0.34
Online-FW($\hat{\eta}$)	82.55± 0.27	19.46± 1.66	56.06± 3.04	13.04± 0.26	**95.01± 0.02**	6.85± 0.18	15.58± 4.48	11.69± 0.74	**82.78± 0.05**	1.92± 0.43	**82.57± 0.10**	21.99± 5.32	*80.97± 0.10*	22.87± 1.17
OMMA	82.11± 0.35	3.51± 0.08	72.84± 0.61	3.53± 0.07	*94.89± 0.05*	2.83± 0.14	*83.88± 0.51*	3.19± 0.58	82.72± 0.15	3.15± 0.66	82.08± 0.14	3.57± 0.60	80.95± 0.08	3.81± 0.40
OMMA($\hat{\eta}$)	83.07± 0.11	3.57± 0.07	**73.41± 1.06**	3.53± 0.03	**95.01± 0.03**	3.11± 0.21	83.30± 0.43	3.16± 0.56	*82.77± 0.06*	3.44± 0.81	82.18± 0.10	3.45± 0.56	**81.00± 0.09**	3.56± 0.07
Offline-FW	83.31± 0.02	–	72.37± 0.28	–	94.93± 0.00	–	80.87± 1.48	–	82.61± 0.11	–	82.07± 0.01	–	78.52± 0.59	–
	LEDGAR-LEXGLUE ($m = 100, n = 10000$)													
Top-k	79.06± 0.00	0.09± 0.00	51.80± 0.00	0.12± 0.00	92.04± 0.00	0.10± 0.00	38.88± 0.00	0.10± 0.01	0.00± 0.00	0.10± 0.02	0.00± 0.00	0.12± 0.03	69.21± 0.00	0.09± 0.00
Greedy	79.30± 0.03	1.71± 0.05	78.08± 0.29	1.81± 0.03	93.26± 0.23	1.69± 0.06	*91.17± 0.58*	1.70± 0.08	×	×	×	×	×	×
Greedy($\hat{\eta}$)	**79.35± 0.24**	1.72± 0.02	*78.21± 0.29*	1.75± 0.03	93.26± 0.23	1.64± 0.05	*91.17± 0.58*	1.69± 0.24	×	×	×	×	×	×
Online-FW	79.22± 0.10	7.21± 1.60	70.94± 0.80	7.32± 0.21	93.30± 0.28	6.31± 0.09	54.52± 0.74	8.04± 1.97	62.31± 31.15	3.06± 0.47	75.81± 0.67	8.73± 1.86	**74.59± 0.31**	5.49± 0.67
Online-FW($\hat{\eta}$)	79.22± 0.09	14.62± 3.47	73.85± 0.28	13.48± 1.04	*93.38± 0.10*	8.56± 0.63	49.63± 2.36	9.97± 0.19	*78.02± 0.33*	3.14± 0.64	**77.58± 0.31**	37.99± 4.17	74.50± 0.32	41.84± 3.15
OMMA	79.28± 0.04	4.50± 0.17	78.10± 0.27	5.05± 0.08	93.26± 0.23	4.12± 0.29	**91.41± 0.75**	3.84± 0.27	77.48± 0.61	3.96± 0.75	74.62± 2.49	5.20± 1.14	**74.59± 0.25**	4.76± 0.10
OMMA($\hat{\eta}$)	*79.34± 0.26*	4.70± 0.09	**78.22± 0.30**	5.11± 0.11	**93.39± 0.09**	4.02± 0.27	90.10± 0.33	4.01± 0.45	**78.03± 0.32**	4.01± 0.69	*76.08± 0.37*	4.54± 0.09	*74.53± 0.24*	4.84± 0.05
Offline-FW	78.76± 0.13	–	77.82± 0.51	–	93.46± 0.00	–	82.78± 1.16	–	77.33± 0.36	–	75.12± 0.07	–	74.25± 0.13	–
	CALTECH-256 ($m = 256, n = 14890$)													
Top-k	79.45± 0.00	0.16± 0.00	46.82± 0.00	0.18± 0.00	89.85± 0.00	0.18± 0.00	32.58± 0.00	0.20± 0.01	77.32± 0.00	0.17± 0.01	75.69± 0.00	0.18± 0.00	74.53± 0.00	0.16± 0.00
Greedy	79.59± 0.07	2.60± 0.04	79.02± 0.10	2.69± 0.04	90.20± 0.07	2.72± 0.07	96.61± 0.32	2.93± 0.29	×	×	×	×	×	×
Greedy($\hat{\eta}$)	**79.70± 0.06**	2.71± 0.05	**79.14± 0.07**	2.74± 0.07	90.20± 0.07	2.79± 0.15	96.61± 0.32	2.88± 0.39	×	×	×	×	×	×
Online-FW	79.15± 0.16	10.07± 0.42	70.29± 0.77	12.46± 1.39	90.20± 0.05	57.84± 2.57	63.22± 0.61	12.08± 1.01	78.31± 0.14	3.38± 0.04	*77.99± 0.12*	16.06± 1.98	76.00± 0.11	15.86± 2.15
Online-FW($\hat{\eta}$)	79.29± 0.08	52.60± 2.07	72.97± 0.85	28.32± 2.72	*90.34± 0.03*	21.59± 1.07	65.42± 1.30	22.67± 1.24	**78.41± 0.12**	3.94± 0.03	**78.08± 0.12**	53.02± 0.23	*76.07± 0.12*	52.89± 0.17
OMMA	79.54± 0.08	7.01± 0.10	78.96± 0.13	7.28± 0.16	90.20± 0.07	5.71± 0.10	*96.77± 0.36*	7.13± 0.46	78.33± 0.13	5.61± 0.12	77.12± 0.17	6.42± 0.20	**76.15± 0.07**	6.99± 0.17
OMMA($\hat{\eta}$)	*79.66± 0.04*	7.11± 0.09	*79.11± 0.06*	7.23± 0.08	**90.35± 0.03**	5.88± 0.18	**96.96± 0.12**	7.16± 0.54	*78.36± 0.11*	5.68± 0.13	77.01± 0.10	6.35± 0.06	75.99± 0.09	7.13± 0.14
Offline-FW	78.40± 0.00	–	77.74± 0.07	–	90.28± 0.00	–	91.34± 1.24	–	77.88± 0.09	–	76.59± 0.00	–	75.72± 0.00	–

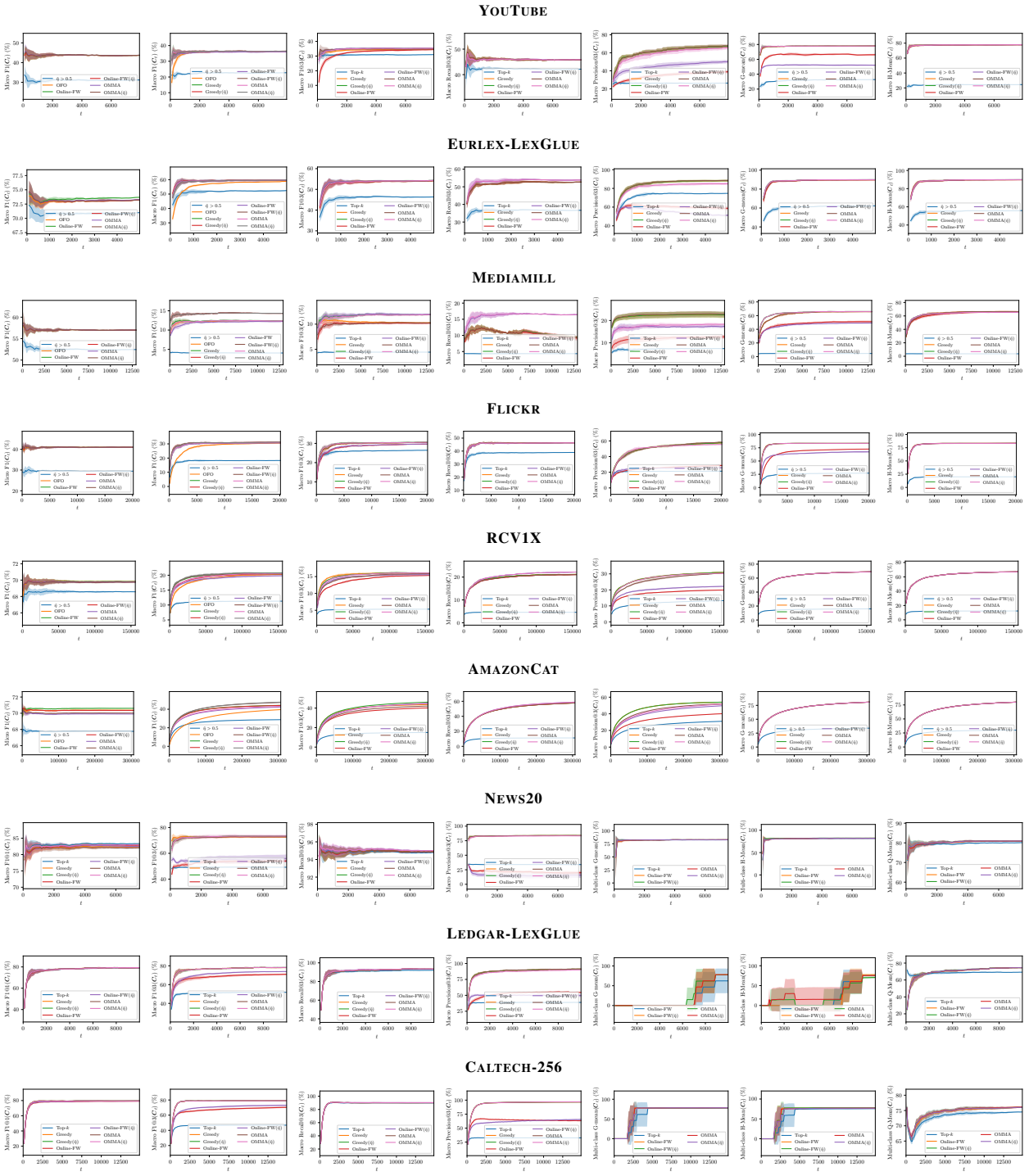

Figure 4. Running comparison of performance for the online algorithms. Averaged over 5 runs, the opaque fill indicate the standard deviation at given iteration t.

Figure 5. Comparison of performance of the online algorithms with different values of λ. Averaged over 5 runs.

Table 6. Predictive performance and running times of the different online algorithms on *multi-label* problems with *incrementally updated CPE*, averaged over 5 runs and reported with standard deviations ($\pm$)s. The best result on each metric is in **bold**, the second best is in *italic*. We additionally report basic statistics of the benchmarks: number of classes/labels and instances in the test sequence. $\times$ – means that the algorithm does not support the optimization of that metric. The inference time of Offline-FW was not measured.

Method	Micro F1		Macro F1		Macro F1@3		Macro Recall@3		Macro Precision@3		Macro G-Mean		Macro H-Mean	
	(%)	time (s)	(%)	time (s)	(%)	time (s)	(%)	time (s)	(%)	time (s)	(%)	time (s)	(%)	time (s)
	YOUTUBE ($m = 46, n = 31703$)													
Top-k / $\hat{\eta} > 0.5$	26.07$\pm$ 0.11	2.84$\pm$ 0.07	18.26$\pm$ 0.06	2.73$\pm$ 0.04	27.73$\pm$ 0.03	0.29$\pm$ 0.00	38.11$\pm$ 0.04	0.28$\pm$ 0.00	23.08$\pm$ 0.12	0.28$\pm$ 0.00	27.83$\pm$ 0.18	2.80$\pm$ 0.02	19.85$\pm$ 0.07	2.81$\pm$ 0.09
OFO	32.52$\pm$ 0.31	4.38$\pm$ 0.03	28.26$\pm$ 0.40	4.15$\pm$ 0.03	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$
Greedy	$\times$	$\times$	25.21$\pm$ 0.17	7.75$\pm$ 0.12	*31.53$\pm$ 0.10*	5.27$\pm$ 0.06	36.76$\pm$ 0.11	5.08$\pm$ 0.06	*46.91$\pm$ 3.36*	4.79$\pm$ 0.02	**76.39$\pm$ 0.01**	7.79$\pm$ 0.06	**76.41$\pm$ 0.03**	7.99$\pm$ 0.03
Greedy($\hat{\eta}$)	$\times$	$\times$	*31.88$\pm$ 0.04*	7.88$\pm$ 0.18	31.17$\pm$ 0.06	5.44$\pm$ 0.03	36.76$\pm$ 0.11	5.06$\pm$ 0.05	*46.91$\pm$ 3.36*	4.79$\pm$ 0.04	**76.39$\pm$ 0.02**	7.82$\pm$ 0.08	*76.31$\pm$ 0.02*	8.00$\pm$ 0.06
Online-FW	33.95$\pm$ 0.08	22.50$\pm$ 1.65	26.33$\pm$ 0.22	21.96$\pm$ 0.71	31.13$\pm$ 0.10	20.70$\pm$ 2.49	36.74$\pm$ 0.10	10.65$\pm$ 0.19	27.97$\pm$ 0.89	6.08$\pm$ 0.19	*76.03$\pm$ 0.08*	14.64$\pm$ 0.60	75.92$\pm$ 0.04	21.97$\pm$ 2.08
Online-FW($\hat{\eta}$)	**39.46$\pm$ 0.02**	19.60$\pm$ 0.24	31.78$\pm$ 0.02	20.52$\pm$ 0.14	30.09$\pm$ 0.09	67.48$\pm$ 0.57	*42.03$\pm$ 0.06*	12.75$\pm$ 0.14	9.36$\pm$ 0.71	50.73$\pm$ 2.62	75.53$\pm$ 0.07	11.52$\pm$ 2.78	75.15$\pm$ 0.10	55.67$\pm$ 10.27
OMMA	32.02$\pm$ 0.14	20.90$\pm$ 0.36	25.21$\pm$ 0.17	18.06$\pm$ 0.20	**31.54$\pm$ 0.12**	14.09$\pm$ 0.13	36.75$\pm$ 0.11	11.86$\pm$ 0.21	**47.72$\pm$ 3.13**	11.73$\pm$ 0.12	**76.39$\pm$ 0.01**	19.08$\pm$ 0.14	**76.41$\pm$ 0.02**	20.31$\pm$ 0.35
OMMA($\widehat{\eta}$)	*39.34$\pm$ 0.02*	21.06$\pm$ 0.34	**31.89$\pm$ 0.05**	17.90$\pm$ 0.34	31.13$\pm$ 0.12	14.06$\pm$ 0.11	**42.05$\pm$ 0.05**	11.75$\pm$ 0.25	43.87$\pm$ 1.10	11.64$\pm$ 0.39	**76.39$\pm$ 0.02**	18.95$\pm$ 0.14	*76.31$\pm$ 0.03*	20.17$\pm$ 0.06
	EURLEX-LEXGLUE ($m = 100, n = 65000$)													
Top-k / $\hat{\eta} > 0.5$	26.50$\pm$ 0.03	10.76$\pm$ 0.06	6.54$\pm$ 0.02	10.83$\pm$ 0.00	12.69$\pm$ 0.03	0.60$\pm$ 0.01	10.90$\pm$ 0.02	0.58$\pm$ 0.01	31.48$\pm$ 1.17	0.60$\pm$ 0.01	9.31$\pm$ 0.02	10.73$\pm$ 0.02	6.62$\pm$ 0.02	10.75$\pm$ 0.09
OFO	47.54$\pm$ 0.01	16.24$\pm$ 0.08	22.67$\pm$ 0.14	15.83$\pm$ 0.17	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$
Greedy	$\times$	$\times$	21.53$\pm$ 0.06	21.14$\pm$ 0.12	22.61$\pm$ 0.07	10.71$\pm$ 0.02	25.02$\pm$ 0.11	10.18$\pm$ 0.35	35.17$\pm$ 0.24	10.09$\pm$ 0.05	*71.71$\pm$ 0.01*	21.27$\pm$ 0.11	*72.49$\pm$ 0.05*	21.60$\pm$ 0.31
Greedy($\hat{\eta}$)	$\times$	$\times$	*28.61$\pm$ 0.03*	20.73$\pm$ 0.15	*29.86$\pm$ 0.04*	10.80$\pm$ 0.07	25.02$\pm$ 0.11	10.07$\pm$ 0.15	35.17$\pm$ 0.24	10.20$\pm$ 0.09	70.56$\pm$ 0.02	21.27$\pm$ 0.05	69.62$\pm$ 0.04	21.44$\pm$ 0.08
Online-FW	49.10$\pm$ 0.12	35.78$\pm$ 0.37	21.10$\pm$ 0.19	36.84$\pm$ 0.90	25.73$\pm$ 0.21	29.24$\pm$ 3.35	24.56$\pm$ 0.09	15.94$\pm$ 0.32	21.78$\pm$ 1.41	12.41$\pm$ 0.29	70.89$\pm$ 0.29	40.18$\pm$ 0.64	71.00$\pm$ 0.13	54.91$\pm$ 2.13
Online-FW($\hat{\eta}$)	**51.78$\pm$ 0.03**	38.66$\pm$ 0.41	26.70$\pm$ 0.13	40.81$\pm$ 0.10	28.70$\pm$ 0.04	113.14$\pm$ 0.51	*26.17$\pm$ 0.04*	21.37$\pm$ 0.17	25.42$\pm$ 1.42	95.89$\pm$ 0.51	70.67$\pm$ 0.08	23.76$\pm$ 0.76	70.06$\pm$ 0.10	121.24$\pm$ 0.35
OMMA	47.09$\pm$ 0.05	49.15$\pm$ 0.37	21.53$\pm$ 0.06	43.19$\pm$ 0.11	22.58$\pm$ 0.03	28.61$\pm$ 0.05	25.01$\pm$ 0.12	22.67$\pm$ 0.16	*63.71$\pm$ 2.42*	23.14$\pm$ 0.17	**71.73$\pm$ 0.01**	46.14$\pm$ 0.56	**72.51$\pm$ 0.05**	48.10$\pm$ 0.68
OMMA($\widehat{\eta}$)	*51.60$\pm$ 0.03*	48.38$\pm$ 0.29	**28.63$\pm$ 0.02**	42.67$\pm$ 0.74	**29.87$\pm$ 0.04**	29.60$\pm$ 0.54	**26.25$\pm$ 0.04**	22.98$\pm$ 0.04	**64.96$\pm$ 1.49**	23.43$\pm$ 0.25	70.58$\pm$ 0.02	46.16$\pm$ 0.12	69.64$\pm$ 0.04	48.40$\pm$ 0.79
	MEDIAMILL ($m = 101, n = 43907$)													
Top-k / $\hat{\eta} > 0.5$	48.38$\pm$ 0.04	5.21$\pm$ 0.06	3.09$\pm$ 0.01	5.21$\pm$ 0.05	3.65$\pm$ 0.02	0.39$\pm$ 0.01	3.72$\pm$ 0.01	0.38$\pm$ 0.02	6.87$\pm$ 0.20	0.39$\pm$ 0.00	3.95$\pm$ 0.06	4.91$\pm$ 0.03	2.45$\pm$ 0.01	5.15$\pm$ 0.05
OFO	52.00$\pm$ 0.02	8.22$\pm$ 0.04	**8.14$\pm$ 0.03**	8.63$\pm$ 0.16	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$
Greedy	$\times$	$\times$	*8.12$\pm$ 0.02*	12.39$\pm$ 0.02	4.71$\pm$ 0.04	7.00$\pm$ 0.09	4.22$\pm$ 0.07	6.68$\pm$ 0.10	**11.60$\pm$ 1.88**	6.56$\pm$ 0.11	*53.55$\pm$ 0.11*	12.11$\pm$ 0.10	53.72$\pm$ 0.11	11.96$\pm$ 0.03
Greedy($\hat{\eta}$)	$\times$	$\times$	7.79$\pm$ 0.01	12.17$\pm$ 0.01	**5.67$\pm$ 0.03**	7.10$\pm$ 0.15	4.22$\pm$ 0.07	6.59$\pm$ 0.06	**11.60$\pm$ 1.88**	6.62$\pm$ 0.15	47.59$\pm$ 0.15	11.85$\pm$ 0.05	43.82$\pm$ 0.21	11.90$\pm$ 0.04
Online-FW	51.81$\pm$ 0.03	27.08$\pm$ 0.91	8.02$\pm$ 0.00	25.08$\pm$ 1.11	4.82$\pm$ 0.08	14.94$\pm$ 2.17	4.08$\pm$ 0.05	10.43$\pm$ 0.41	6.19$\pm$ 0.15	7.47$\pm$ 0.10	**53.63$\pm$ 0.01**	19.08$\pm$ 0.09	**53.77$\pm$ 0.07**	21.75$\pm$ 0.10
Online-FW($\hat{\eta}$)	**52.16$\pm$ 0.01**	27.50$\pm$ 0.56	7.74$\pm$ 0.03	32.25$\pm$ 0.21	**5.67$\pm$ 0.03**	84.47$\pm$ 2.23	*4.27$\pm$ 0.08*	16.26$\pm$ 0.22	6.24$\pm$ 0.36	75.56$\pm$ 0.87	49.23$\pm$ 0.40	12.18$\pm$ 0.29	41.81$\pm$ 0.36	72.49$\pm$ 0.98
OMMA	51.71$\pm$ 0.02	30.72$\pm$ 0.33	*8.12$\pm$ 0.02*	26.51$\pm$ 0.13	4.71$\pm$ 0.04	18.89$\pm$ 0.14	4.22$\pm$ 0.07	15.20$\pm$ 0.27	*10.30$\pm$ 1.06*	15.56$\pm$ 0.13	53.54$\pm$ 0.12	27.21$\pm$ 0.25	*53.73$\pm$ 0.11*	29.98$\pm$ 0.24
OMMA($\widehat{\eta}$)	*52.12$\pm$ 0.01*	30.63$\pm$ 0.48	7.80$\pm$ 0.01	26.04$\pm$ 0.17	*5.66$\pm$ 0.03*	18.61$\pm$ 0.12	**4.30$\pm$ 0.06**	15.20$\pm$ 0.11	8.77$\pm$ 0.54	15.20$\pm$ 0.15	47.57$\pm$ 0.14	28.14$\pm$ 0.38	43.81$\pm$ 0.22	28.18$\pm$ 0.05
	FLICKR ($m = 195, n = 80513$)													
Top-k / $\hat{\eta} > 0.5$	14.42$\pm$ 0.03	16.48$\pm$ 0.12	4.25$\pm$ 0.05	16.50$\pm$ 0.27	14.73$\pm$ 0.14	0.85$\pm$ 0.02	19.71$\pm$ 0.07	0.86$\pm$ 0.01	17.10$\pm$ 0.69	0.88$\pm$ 0.01	7.85$\pm$ 0.15	16.48$\pm$ 0.17	4.52$\pm$ 0.05	16.41$\pm$ 0.11
OFO	**32.28$\pm$ 0.05**	24.31$\pm$ 0.78	14.22$\pm$ 0.12	24.56$\pm$ 0.68	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$
Greedy	$\times$	$\times$	5.74$\pm$ 0.03	30.57$\pm$ 0.38	15.23$\pm$ 0.03	14.14$\pm$ 0.20	21.98$\pm$ 0.16	13.53$\pm$ 0.21	**31.81$\pm$ 1.19**	13.17$\pm$ 0.18	62.53$\pm$ 0.04	30.87$\pm$ 0.65	65.78$\pm$ 0.02	31.65$\pm$ 0.57
Greedy($\hat{\eta}$)	$\times$	$\times$	*15.13$\pm$ 0.16*	30.93$\pm$ 0.33	16.05$\pm$ 0.03	14.09$\pm$ 0.09	21.98$\pm$ 0.16	13.35$\pm$ 0.12	**31.81$\pm$ 1.19**	13.14$\pm$ 0.19	**69.00$\pm$ 0.05**	31.57$\pm$ 1.05	*68.05$\pm$ 0.03*	32.38$\pm$ 0.18
Online-FW	14.05$\pm$ 0.37	58.37$\pm$ 1.02	5.79$\pm$ 0.04	70.80$\pm$ 2.28	14.77$\pm$ 0.12	29.77$\pm$ 3.92	21.69$\pm$ 0.15	20.50$\pm$ 0.18	12.51$\pm$ 0.70	15.81$\pm$ 0.36	66.00$\pm$ 0.17	50.70$\pm$ 0.48	66.77$\pm$ 0.08	45.53$\pm$ 0.75
Online-FW($\hat{\eta}$)	*31.58$\pm$ 0.07*	58.87$\pm$ 0.38	13.97$\pm$ 0.07	61.33$\pm$ 0.28	**16.09$\pm$ 0.04**	166.71$\pm$ 3.33	*27.15$\pm$ 0.02*	34.83$\pm$ 0.55	14.57$\pm$ 0.77	140.62$\pm$ 0.84	*67.71$\pm$ 0.10*	34.11$\pm$ 0.28	**68.48$\pm$ 0.08**	160.12$\pm$ 0.75
OMMA	10.51$\pm$ 0.45	64.39$\pm$ 0.63	5.74$\pm$ 0.03	58.36$\pm$ 1.42	15.24$\pm$ 0.01	36.03$\pm$ 0.40	21.95$\pm$ 0.15	28.58$\pm$ 0.16	28.88$\pm$ 1.35	29.26$\pm$ 0.20	62.53$\pm$ 0.04	63.16$\pm$ 1.74	65.77$\pm$ 0.03	67.12$\pm$ 0.66
OMMA($\widehat{\eta}$)	31.33$\pm$ 0.08	66.84$\pm$ 0.36	**15.15$\pm$ 0.17**	59.32$\pm$ 1.07	*16.06$\pm$ 0.03*	36.00$\pm$ 0.32	**27.18$\pm$ 0.02**	29.11$\pm$ 0.18	*29.52$\pm$ 0.24*	28.93$\pm$ 0.54	**69.00$\pm$ 0.05**	61.45$\pm$ 1.41	*68.05$\pm$ 0.03*	65.96$\pm$ 0.39

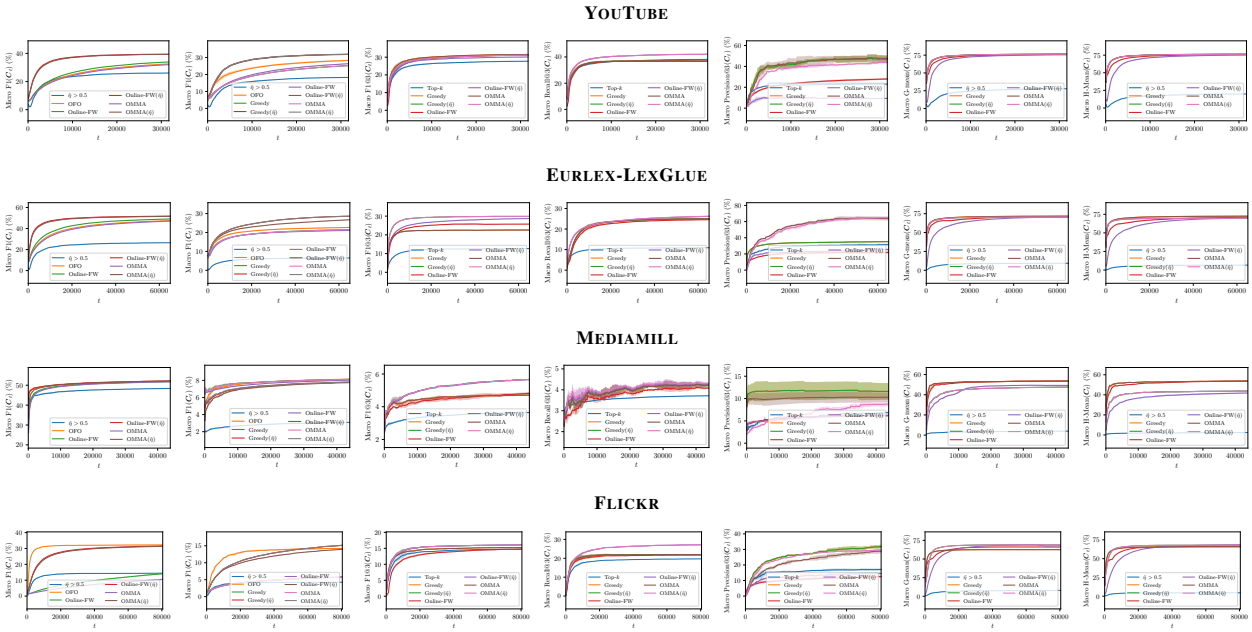

Figure 6. Running comparison of performance for the online algorithms with *incrementally updated CPE*. Averaged over 5 runs, the opaque fill indicate the standard deviation at given iteration t.