Ultraconservative Online Algorithms for Multiclass Problems

Koby Crammer[1] and Yoram Singer[1]

School of Computer Science & Engineering
The Hebrew University, Jerusalem 91904, Israel
{kobics,singer}@cs.huji.ac.il

Abstract. In this paper we study online classification algorithms for multiclass problems in the mistake bound model. The hypotheses we use maintain one prototype vector per class. Given an input instance, a multiclass hypothesis computes a similarity-score between each prototype and the input instance and then sets the predicted label to be the index of the prototype achieving the highest similarity. To design and analyze the learning algorithms in this paper we introduce the notion of *ultraconservativeness.* Ultraconservative algorithms are algorithms that update only the prototypes attaining similarity-scores which are higher than the score of the correct label's prototype. We start by describing a family of additive ultraconservative algorithms where each algorithm in the family updates its prototypes by finding a feasible solution for a set of linear constraints that depend on the instantaneous similarity-scores. We then discuss a specific online algorithm that seeks a set of prototypes which have a small norm. The resulting algorithm, which we term MIRA (for Margin Infused Relaxed Algorithm) is ultraconservative as well. We derive mistake bounds for all the algorithms and provide further analysis of MIRA using a generalized notion of the margin for multiclass problems.

1 Introduction

In this paper we present a general approach for deriving algorithms for multiclass prediction problems. In multiclass problems the goal is to assign one of k labels to each input instance. Many machine learning problems can be phrased as a multiclass categorization problem. Examples to such problems include optical character recognition (OCR), text classification, and medical analysis. There are numerous specialized solutions for multiclass problems for specific models such as decision trees [3, 16] and neural networks. Another general approach is based on reducing a multiclass problem to multiple binary problems using output coding [6, 1]. An example of a reduction that falls into the above framework is the "one-against-rest" approach. In one-against-rest a set of binary classifiers is trained, one classifier for each class. The ith classifier is trained to distinguish between the ith class and the rest of the classes. New instances are classified by setting the predicted label to be the index of the classifier attaining the highest score in its prediction. We present a unified approach that operates directly on the multiclass problem by imposing constraints on the updates for the various classes. Thus, our approach is inherently different from methods based on output coding.

Our framework for analyzing the algorithms is the mistake bound model. The algorithms we study work in rounds. On each round the proposed algorithms get a new instance and output a prediction for the instance. They then receive the correct label and update their predication rule in case they made a prediction error. The goal of the algorithms is to minimize the number of mistakes they made compared to the minimal number of errors that an hypothesis, built offline, can achieve.

The algorithms we consider in this paper maintain one prototype vector for each class. Given a new instance we compare each prototype to the instance by computing the similarity-score between the instance and each of the prototypes for the different classes. We then predict the class which achieves the highest similarity-score. In binary problems, this scheme reduces (under mild conditions) to a linear discriminator. After the algorithm makes a prediction it receives the correct label of the input instance and updates the set of prototypes. For a given input instance, the set of labels that attain similarity-scores higher than the score of correct label is called the *error set*. The algorithms we describe share a common feature: they all update only the prototypes from the error sets and the prototype of the correct label. We call such algorithms *ultraconservative*.

We start in Sec. 3 in which we provide a motivation for our framework. We do that by revisiting the well known perceptron algorithm and give a new account of the algorithm using two prototype vectors, one for each class. We then extend the algorithm to a multiclass setting using the notion of ultraconservativeness. In Sec. 4 we further generalize the multiclass version of the extended perceptron algorithm and describe a new family of ultraconservative algorithms that we obtain by replacing the perceptron's update with a set of linear equations. We give a few illustrative examples of specific updates from this family of algorithms. Going back to the perceptron algorithm, we show that in the binary case all the different updates reduce to the perceptron algorithm. We finish Sec. 4 by deriving a mistake bound that is common to all the additive algorithms in the family. We analyze both the separable and the non-separable case.

The fact that all algorithms from Sec. 4 achieve the same mistake bound implies that there are some undetermined degrees of freedom. We present in Sec. 5 a new online algorithm that gives a unique update and is based on a relaxation of the set of linear constraints employed by the family of algorithms from Sec. 4. The algorithm is derived by adding an objective function that incorporates the norm of the new matrix of prototypes and minimizing it subject to a subset of the linear constraints. Following recent trend, we call the new algorithm MIRA for Margin Infused Relaxed Algorithm. We analyze MIRA and give a mistake bound related to the instantaneous margin of individual examples. This analysis leads to modification of MIRA which incorporates the margin into the update rule. Both MIRA and of the additive algorithms from Sec. 4 can be combined with kernels techniques and voting methods.

The algorithms presented in this paper underscore a general framework for deriving ultraconservative multiclass algorithms. This framework can be used in combination with other online techniques. To conclude, we outline some of our current research directions. Due to the lack of space many of the proofs and some of the results have omitted. They will appear in a forthcoming long version of this paper.

Related Work Multiclass extensions to binary approaches by maintaining multiple prototypes are by no means new. The widely read and cited book by Duda and Hart [7] describes a multiclass extension to the perceptron that employs multiple vectors. However, direct methods for online learning of multiclass problems in the mistake bound model have received relatively little attention.

A question that is common to numerous online algorithms is how to compromise the following two demands. On one hand, we want to update the classifier we learn so that it will better predict the current input instance, in particular if an error occurs when using the current classifier. On the other hand, we do not want to change the current classifier too radically, especially if it classifies well most of the previously observed instances. The good old perceptron algorithm suggested by Rosenblatt [17] copes with these two requirements by replacing the classifier with a linear combination of the current hyperplane and the current instance vector. Although the algorithm uses a simple update rule, it performs well on many synthetic and real-world problems. The perceptron algorithm spurred voluminous work which clearly cannot be covered here. For an overview of numerous additive and multiplicative online algorithms see the paper by Kivinen and Warmuth [12]. We outline below some of the research that is more relevant to the work presented in this paper.

Kivinen and Warmuth [12] presented numerous online algorithms for regression. Their algorithms are based on minimization of an objective function which is a sum of two terms. The first term is equal to the distance between the new classifier and the current classifier while the second term is the loss on the current example. The resulting update rule can be viewed as a gradient-descent method. Although multiclass classification problems are a special case of regression problems, the algorithms for regression put emphasis on smooth loss functions which might not be suitable for classification problems.

The idea of seeking a hyperplane of a small norm is a primary goal in support vector machines (SVM) [4, 18]. Algorithms for constructing support vector machines solve optimization problems with a quadratic objective function and linear constraints. The work in [2, 9] suggests to minimize the objective function in a gradient-decent method, which can be performed by going over the sample sequentially. Algorithms with a similar approach include the Sequential Minimization Optimization (SMO) algorithm introduced by Platt [15]. SMO works on rounds, on each round it chooses two examples of the sample and minimizes the objective function by modifying variables relevant only to these two examples. While these algorithms share some similarities with the algorithmic approaches described in this paper, they were all designed for batch problems and were not analyzed in the mistake bound model.

Another approach to the problem of designing an update rule which results in a linear classifier of a small norm was suggested by Li and Long [13]. The algorithm Li and Long proposed, called ROMMA, tackles the problem by finding a hyperplane with a minimal norm under two linear constraints. The first constraint is presented so that the new classifier will classify well previous examples, while the second rule demands that the hyperplane will classify correctly the current new instance. Solving this minimization problem leads to an additive update rule with adaptive coefficients.

Grove, Littlestone and Schuurmans [11] introduced a general framework of quasi-additive binary algorithms, which contain the perceptron and Winnow as special cases. In [10] Gentile proposed an extension to a subset of the quasi-additive algorithms, which uses an additive conservative update rule with decreasing learning rates.

The algorithms presented in this paper are reminiscent of some of the widely used methods for constructing classifiers in multiclass problems. As mentioned above, a popular approach for solving classification problems with many classes is to learn a set of binary classifiers where each classifier is designed to separate one class from the rest of classes. If we use the perceptron algorithm to learn the binary classifiers, we need to maintain and update one vector for each possible class. This approach shares the same form of hypothesis as the algorithms presented in this paper, which maintain one prototype per class. Nonetheless, there is one major difference between the ultraconservative algorithms we present and the one-against-rest approach. In one-against-rest we update and change each of the classifiers *independently* of the others. In fact we can construct them one after the other by re-running over the data. In contrast, ultraconservative algorithms update all the prototypes in tandem thus updating one prototype has a global effect on the other prototypes. There are situations in which there is an error due to some classes, but not all the respective prototypes should be updated. Put another way, we might perform milder changes to the set of classifiers by changing them together with the prototypes so as to achieve the same goal. As a result we get better mistake bounds and empirically better algorithms.

2 Preliminaries

The focus of this paper is online algorithms for multiclass prediction problems. We observe a sequence $(\bar{x}^1, y^1), \ldots, (\bar{x}^t, y^t), \ldots$ of instance-label pairs. Each instance $\bar{x}^t$ is in $\mathbb{R}^n$ and each label belongs to a finite set $\mathcal{Y}$ of size k. We assume without loss of generality that $\mathcal{Y} = \{1, 2, \ldots, k\}$. A *multiclass classifier* is a function $H(\bar{x})$ that maps instances from $\mathbb{R}^n$ into one of the possible labels in $\mathcal{Y}$. In this paper we focus on classifiers of the form $H(\bar{x}) = \arg\max_{r=1}^{k}\{\bar{M}_r \cdot \bar{x}\}$, where $\mathbf{M}$ is a $k \times n$ matrix over the reals and $\bar{M}_r \in \mathbb{R}^n$ denotes the rth row of $\mathbf{M}$. We call the inner product of $\bar{M}_r$ with the instance $\bar{x}$, the *similarity-score* for class r. Thus, the classifiers we consider in this paper set the label of an instance to be the index of the row of $\mathbf{M}$ which achieves the highest similarity-score. The margin of H on $\bar{x}$ is the difference between the similarity-score of the correct label y and the maximum among the similarity-scores of the rest of the rows of $\mathbf{M}$. Formally, the margin that $\mathbf{M}$ achieves on $(\bar{x}, y)$ is,

$$\bar{M}_y \cdot \bar{x} - \max_{r \neq y}\{\bar{M}_r \cdot \bar{x}\} \quad .$$

The l_p norm of a vector $\bar{u} = (u_1, \ldots, u_l)$ in $\mathbb{R}^l$ is $\|\bar{u}\|_p = \left(\sum_{i=1}^{l} |u_i|^p\right)^{\frac{1}{p}}$. We define the l_p vector-norm of a matrix $\mathbf{M}$ to be the l_p norm of the vector we get by concatenating the rows of $\mathbf{M}$, that is,

$$\|\mathbf{M}\|_p = \|(\bar{M}_1, \ldots, \bar{M}_k)\|_p \quad .$$

The framework that we use in this paper is the mistake bound model for online learning. The algorithms we consider work in rounds. On round t an online learning algorithm gets an instance $\bar{x}^t$. Given $\bar{x}^t$, the learning algorithm outputs a prediction, $\hat{y}^t = \arg\max_r\{\bar{M}_r \cdot \bar{x}^t\}$. It then receives the correct label y^t and updates its classification rule by modifying the matrix $\mathbf{M}$. We say that the algorithm made a (multiclass) prediction error if $\hat{y}^t \neq y^t$. Our goal is to make as few prediction errors as possible. When the algorithm makes a prediction error there might be more than one row of $\mathbf{M}$ achieving a score higher than the score of the row corresponding to the correct label. We define the *error-set* for $(\bar{x}, y)$ using a matrix $\mathbf{M}$ to be the index of all the rows in $\mathbf{M}$ which achieve such high scores. Formally, the error-set for a matrix $\mathbf{M}$ on an instance-label pair $(\bar{x}, y)$ is,

$$E = \{r \neq y : \bar{M}_r \cdot \bar{x} \geq \bar{M}_y \cdot \bar{x}\} \ .$$

Many online algorithms update their prediction rule only on rounds on which they made a prediction error. Such algorithms are called *conservative*. We give a definition that extends the notion of conservativeness to multiclass settings.

Definition 1 (Ultraconservative). *An online multiclass algorithm of the form $H(\bar{x}) = \arg\max_r\{\bar{M}_r \cdot \bar{x}\}$ is ultraconservative if it modifies $\mathbf{M}$ only when the error-set E for $(\bar{x}, y)$ is not empty and the indices of the rows that are modified are from $E \cup \{y\}$.*

Note that our definition implies that an ultraconservative algorithm is also conservative. For binary problems the two definitions coincide.

3 From binary to multiclass

Roseneblatt's perceptron algorithm [17] is a well known online algorithm for binary classification problems. The algorithm maintains a weight vector $\bar{w} \in \mathbb{R}^n$ that is used for prediction. To motivate our multiclass algorithms let us now describe the perceptron algorithm using the notation employed in this paper. In our setting the label of each instance belongs to the set $\{1, 2\}$. Given an input instance $\bar{x}$ the perceptron algorithm predicts that its label is $\hat{y} = 1$ iff $\bar{w} \cdot \bar{x} \geq 0$ and otherwise it predicts $\hat{y} = 2$. The algorithm modifies $\bar{w}$ only on rounds with

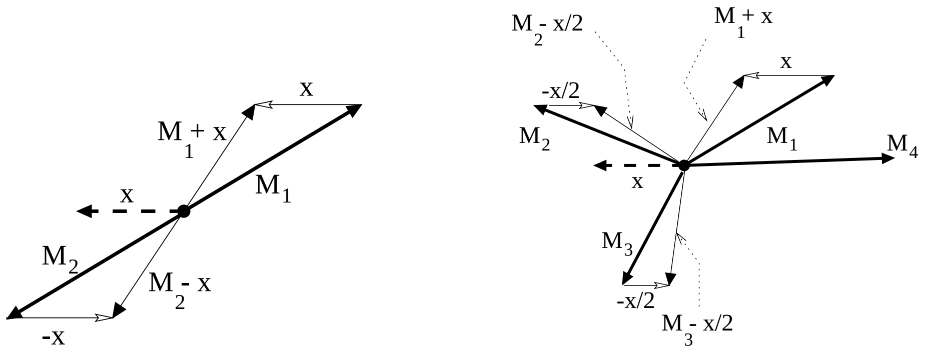

Fig. 1. A geometrical illustration of the update for a binary problem (left) and a four-class problem (right) using the extended perceptron algorithm.

prediction errors and is thus conservative. On such rounds $\bar{w}$ is changed to $\bar{w} + \bar{x}$ if the correct label is $y = 1$ and to $\bar{w} - \bar{x}$ if $y = 2$.

To implement the perceptron algorithm using a prototype matrix $\mathbf{M}$ with one row (prototype) per class, we set the first row $\bar{M}_1$ to $\bar{w}$ and the second row $\bar{M}_2$ to $-\bar{w}$. We now modify $\mathbf{M}$ every time the algorithm misclassifies $\bar{x}$ as follows. If the correct label is 1 we replace $\bar{M}_1$ with $\bar{M}_1 + \bar{x}$ and $\bar{M}_2$ with $\bar{M}_2 - \bar{x}$. Similarly, we replace $\bar{M}_1$ with $\bar{M}_1 - \bar{x}$ and $\bar{M}_2$ with $\bar{M}_2 + \bar{x}$ when the correct label is 2 and $\bar{x}$ is misclassified. Thus, the row $\bar{M}_y$ is moved toward the misclassified instance $\bar{x}$ while the other row is moved away from $\bar{x}$. Note that this update implies that the total change to the two prototypes is zero. An illustration of this geometrical interpretation is given on the left-hand side of Fig. 1. It is straightforward to verify that the algorithm is equivalent to the perceptron algorithm.

We can now use this interpretation and generalize the perceptron algorithm to multiclass problems as follows. For k classes maintain a matrix $\mathbf{M}$ of k rows, one row per class. For each input instance $\bar{x}$, the multiclass generalization of the perceptron calculates the similarity-score between the instance and each of the k prototypes. The predicted label, $\hat{y}$, is the index of the row (prototype) of $\mathbf{M}$ which achieves the highest score, that is, $\hat{y} = \arg\max_r\{\bar{M}_r \cdot \bar{x}\}$. If $\hat{y} \neq y$ the algorithm moves $\bar{M}_y$ toward $\bar{x}$ by replacing $\bar{M}_y$ with $\bar{M}_y + \bar{x}$. In addition, the algorithm moves each row $\bar{M}_r$ $(r \neq y)$ for which $\bar{M}_r \cdot \bar{x} \geq \bar{M}_y \cdot \bar{x}$ away from $\bar{x}$. The indices of these rows constitute the error set E. The algorithms presented in this paper, and in particular the multiclass version of the perceptron algorithm, modify $\mathbf{M}$ such that the following property holds: The total change in units of $\bar{x}$ in the rows of $\mathbf{M}$ that are moved away from $\bar{x}$ is equal to the change of $\bar{M}_y$, (in units of $\bar{x}$). Specifically, for the multiclass perceptron we replace $\bar{M}_y$ with $\bar{M}_y + \bar{x}$ and for each r in E we replace $\bar{M}_r$ with $\bar{M}_r - \bar{x}/|E|$. A geometric illustration of this update is given in the right-hand side of Fig. 1. There are four classes in the example appearing in the figure. The correct label of $\bar{x}$ is $y = 1$ and since $\bar{M}_1$ is not the most similar vector to $\bar{x}$, it is moved toward $\bar{x}$. The rows $\bar{M}_2$ and $\bar{M}_3$ are also modified by subtracting $\bar{x}/2$ from each one. The last row $\bar{M}_4$ is not in the error-set since $\bar{M}_1 \cdot \bar{x} > \bar{M}_4 \cdot \bar{x}$ and therefore it is not modified. We defer the analysis of the algorithm to the next section in which we describe and analyze a family of online multiclass algorithms that also includes this algorithm.

4 A family of additive multiclass algorithms

We describe a family of ultraconservative algorithms by using the algorithm of the previous section as our starting point. The algorithm is ultraconservative and thus updates $\mathbf{M}$ only on rounds with predictions errors. The row $\bar{M}_y$ is changed to $\bar{M}_y + \bar{x}$ while for each $r \in E$ we modify $\bar{M}_r$ to $\bar{M}_r - \bar{x}/|E|$. Let us introduce a vector of weights $\bar{\tau} = (\tau_1, \ldots, \tau_k)$ and rewrite the update of the rth row as $\bar{M}_r + \tau_r \bar{x}$. Thus, for $r = y$ we have $\tau_r = 1$, for $r \in E$ we set $\tau_r = -1/|E|$, and for $r \notin E \cup \{y\}$, τ_r is zero. The weights $\bar{\tau}$ were chosen such that the total change of the rows of $\mathbf{M}$ whose indices are from E are equal to the change in $\bar{M}_y$, that is, $1 = \tau_y = -\sum_{r \in E} \tau_r$. If we do not impose the condition that for $r \in E$ all the τ_r's attain the same value, then the constraints on $\bar{\tau}$ become $\sum_{r \in E \cup \{y\}} \tau_r = 0$. This constraint enables us to move the prototypes from the error-set E away from $\bar{x}$ in different proportions as long as the total change is sum to one. The result is a whole family of multiclass algorithms. A pseudo-code of the family of algorithms is provided in Fig. 2. Note that the constraints on $\bar{\tau}$ are redundant and we could have used less constraints. We make use of this more elaborate set of constraints in the next section.

Before analyzing the family of algorithms we have just introduced, we give a few examples of specific schemes to set $\bar{\tau}$. We have already described one update above which sets $\bar{\tau}$ to,

$$\tau_r = \begin{cases} -\frac{1}{|E|} & r \in E \\ 1 & r = y \\ 0 & \text{otherwise} \end{cases} .$$

Since all the τ's for rows in the error-set are equal, we call this the *uniform* multiclass update. We can also be further conservative and modify in addition to $\bar{M}_y$ only one other row in $\mathbf{M}$. A reasonable choice is to modify the row that achieves the highest similarity-score. That is, we set $\bar{\tau}$ to,

$$\tau_r = \begin{cases} -1 & r = \arg\max_s \{\bar{M}_s \cdot \bar{x}\} \\ 1 & r = y \\ 0 & \text{otherwise} \end{cases} .$$

We call this form of updating $\bar{\tau}$ the *worst-margin* multiclass update. The two examples above set τ_r for $r \in E$ to a fixed value, ignoring the actual values of similarity-scores each row achieves. We can also set $\bar{\tau}$ in promotion to the excess in the similarity-score of each row in the error set (with respect to $\bar{M}_y$). For instance, we can set $\bar{\tau}$ to be,

$$\tau_r = \begin{cases} -\frac{[\bar{M}_r \cdot \bar{x} - \bar{M}_y \cdot \bar{x}]_+}{\sum_{r=1}^{k} [\bar{M}_r \cdot \bar{x} - \bar{M}_y \cdot \bar{x}]_+} & r \neq y \\ 1 & r = y \end{cases} ,$$

```
**Initialize:** Set $\mathbf{M} = 0$ $(\mathbf{M} \in \mathbb{R}^{k\times n})$ .
**Loop:** For $t = 1, 2, \ldots, T$
  – Get a new instance $\bar{x}^t \in \mathbb{R}^n$.
  – Predict $\hat{y}^t = \arg\max_{r=1}^{k}\{\bar{M}_r \cdot \bar{x}^t\}$.
  – Get a new label $y^t$.
  – Set $E = \{r \neq y^t \ : \bar{M}_r \cdot \bar{x}^t \geq \bar{M}_{y^t} \cdot \bar{x}^t\}$.
  – If $E \neq \emptyset$ update $\mathbf{M}$ by choosing any $\tau_1^t, \ldots, \tau_k^t$ that satisfy:
      1. $\tau_r^t \leq \delta_{r,y^t}$ for $r = 1, \ldots, k$.
      2. $\sum_{r=1}^k \tau_r^t = 0$.
      3. $\tau_r^t = 0$ for $r \notin E \cup \{y^t\}$.
      4. $\tau_{y^t}^t = 1$.
  – For $r = 1, 2, \ldots, k$ update: $\bar{M}_r \leftarrow \bar{M}_r + \tau_r^t \, \bar{x}^t$ .
**Output :** $H(\bar{x}) = \arg\max_r\{\bar{M}_r \cdot \bar{x}\}$.
```

Fig. 2. A family of additive multiclass algorithms.

where $[x]_+$ is equal to x if $x \geq 0$ and zero otherwise. Note that the above update implies that $\tau_r = 0$ for $r \notin E \cup \{y\}$. We now proceed to analyze the algorithms.

4.1 Analysis

In the analysis of the algorithms of Fig. 2 we use the following auxiliary lemma.

Lemma 1. *For any set $\{\tau_1, \ldots, \tau_k\}$ such that, $\sum_{r=1}^k \tau_r = 0$ and $\tau_r \leq \delta_{r,y}$ for $r = 1, \ldots, k$, then $\sum_r \tau_r^2 \leq 2\tau_y \leq 2$.*

We now give the main theorem of this section.

Theorem 1. *Let $(\bar{x}^1, y^1), \ldots, (\bar{x}^T, y^T)$ be an input sequence for any multiclass algorithm from the family described in Fig. 2 where $\bar{x}^t \in \mathbb{R}^n$ and $y^t \in \{1, 2, \ldots, k\}$. Denote by $R^2 = \max_t \|\bar{x}^t\|^2$. Assume that there is a matrix $\mathbf{M}^*$ of a unit vector-norm, $\|\mathbf{M}^*\| = 1$, that classifies the entire sequence correctly with margin $\gamma = \min_t\{\bar{M}^*_{y^t} \cdot \bar{x}^t - \max_{r\neq y^t} \bar{M}^*_r \cdot \bar{x}^t\} > 0$. Then, the number of mistakes that the algorithm makes is at most $2R^2/\gamma^2$.*

Proof. Assume that an error occurred when classifying the tth example $(\bar{x}^t, y^t)$ using the matrix $\mathbf{M}$. Denote by $\mathbf{M}^{'}$ the updated matrix after round t. That is, for $r = 1, 2, \ldots, k$ we have $\bar{M}_r^{'} = \bar{M}_r + \tau_r^t \bar{x}^t$. To prove the theorem we bound $\|\mathbf{M}\|_2^2$ from above and below. First, we derive a lower bound on $\|\mathbf{M}\|^2$ by bounding the term,

$$\sum_{r=1}^k \bar{M}_r^* \cdot \bar{M}_r^{'} = \sum_{r=1}^k \bar{M}_r^* \cdot (\bar{M}_r + \tau_r^t \bar{x}^t)$$

$$= \sum_{r=1}^k \bar{M}_r^* \cdot \bar{M}_r + \sum_r \tau_r^t \left(\bar{M}_r^* \cdot \bar{x}^t\right) \ . \qquad (1)$$

We further develop the second term of Eq. (1) using the second constraint of the algorithm $\left(\sum_{r=1}^{k}\tau_r^t = 0\right)$. Substituting $\tau_{y^t} = -\sum_{r\neq y^t}\tau_r^t$ we get,

$$\begin{aligned}\sum_r \tau_r^t\left(\bar{M}_r^*\cdot\bar{x}^t\right) &= \sum_{r\neq y^t}\tau_r^t\left(\bar{M}_r^*\cdot\bar{x}^t\right) + \tau_{y^t}\left(\bar{M}_{y^t}^*\cdot\bar{x}^t\right)\\ &= \sum_{r\neq y^t}\tau_r^t\left(\bar{M}_r^*\cdot\bar{x}^t\right) - \sum_{r\neq y^t}\tau_r^t\left(\bar{M}_{y^t}^*\cdot\bar{x}^t\right)\\ &= \sum_{r\neq y^t}\left(-\tau_r^t\right)\left(\bar{M}_{y^t}^* - \bar{M}_r^*\right)\cdot\bar{x}^t\ . \end{aligned}\tag{2}$$

Using the assumption that $\mathbf{M}^*$ classifies each instance with a margin of at least γ and that $\tau_y = 1$ (fourth constraint) we obtain,

$$\sum_r \tau_r^t\left(\bar{M}_r^*\cdot\bar{x}^t\right) \geq \sum_{r\neq y^t}\left(-\tau_r^t\right)\gamma = \tau_{y^t}^t\gamma = \gamma\ . \tag{3}$$

Combining Eq. (1) and Eq. (3) we get, $\sum_r \bar{M}_r^*\cdot\bar{M}_r^{'} \geq \sum_r \bar{M}_r^*\cdot\bar{M}_r + \gamma$. Thus, if the algorithm made m mistakes in T rounds then the matrix $\mathbf{M}$ satisfies,

$$\sum_r \bar{M}_r^*\cdot\bar{M}_r \geq m\gamma \tag{4}$$

Using the vector-norm definition and applying the Cauchy-Schwartz inequality we get,

$$\begin{aligned}\|\mathbf{M}\|^2\|\mathbf{M}^*\|^2 &= \left(\sum_{r=1}^{k}\|\bar{M}_r\|^2\right)\left(\sum_{r=1}^{k}\|\bar{M}_r^*\|^2\right)\\ &\geq \left(\bar{M}_1\cdot\bar{M}_1^* + \ldots + \bar{M}_k\cdot\bar{M}_k^*\right)^2 \quad = \left(\sum_{r=1}^{k}\bar{M}_r\cdot\bar{M}_r^*\right)^2\ .\end{aligned}\tag{5}$$

Plugging Eq. (4) into Eq. (5) and using the assumption that $\mathbf{M}^*$ is of a unit vector-norm we get the following lower bound,

$$\|\mathbf{M}\|^2 \geq m^2\gamma^2\ . \tag{6}$$

Next, we bound the vector-norm of $\mathbf{M}$ from above. As before, assume that an error occurred when classifying the example $(\bar{x}^t, y^t)$ using the matrix $\mathbf{M}$ and denote by $\mathbf{M}^{'}$ the matrix after the update. Then,

$$\begin{aligned}\|\mathbf{M}^{'}\|^2 = \sum_r\|\bar{M}_r^{'}\|^2 &= \sum_r\|\bar{M}_r + \tau_r^t\bar{x}^t\|^2\\ &= \sum_r\|\bar{M}_r\|^2 + 2\sum_r\tau_r^t\left(\bar{M}_r\cdot\bar{x}^t\right) + \sum_r\|\tau_r^t\bar{x}^t\|^2\\ &= \|\mathbf{M}\|^2 + 2\sum_r\tau_r^t\left(\bar{M}_r\cdot\bar{x}^t\right) + \|\bar{x}^t\|^2\sum_r(\tau_r^t)^2\ .\end{aligned}\tag{7}$$

We further develop the second term using the second constraint of the algorithm and analogously to Eq. (2) we get,

$$\sum_r\tau_r^t\left(\bar{M}_r\cdot\bar{x}^t\right) = \sum_{r\neq y^t}\left(-\tau_r^t\right)\left(\bar{M}_{y^t} - \bar{M}_r\right)\cdot\bar{x}^t\ .$$

Since $\bar{x}^t$ was misclassified we need to consider the following two cases. The first case is when the label r was not the source of the error, that is $(\bar{M}_{y^t} - \bar{M}_r) \cdot \bar{x}^t > 0$. Then, using the third constraint ($r \notin E \cup \{y^t\} \Rightarrow \tau_r^t = 0$) we get that $\tau_r^t = 0$ and thus $(-\tau_r^t)\left(\bar{M}_{y^t} - \bar{M}_r\right) \cdot \bar{x}^t = 0$. The second case is when one of the sources of error was the label r. In that case $(\bar{M}_{y^t} - \bar{M}_r) \cdot \bar{x}^t \leq 0$. Using the first constraint of the algorithm we know that $\tau_r^t \leq 0$ and thus $(-\tau_r^t)\left(\bar{M}_{y^t} - \bar{M}_r\right) \cdot \bar{x}^t \leq 0$. Finally, summing over all r we get,

$$\sum_r \tau_r^t \left(\bar{M}_r \cdot \bar{x}^t\right) \leq 0 \ . \tag{8}$$

Plugging Eq. (8) into Eq. (7) we get, $\|\mathbf{M}^{'}\|^2 \leq \|\mathbf{M}\|^2 + \|\bar{x}^t\|^2 \sum_r (\tau_r^t)^2$. Using the bound $\|\bar{x}^t\|^2 \leq R^2$ and Lemma 1 we obtain,

$$\|\mathbf{M}^{'}\|^2 \leq \|\mathbf{M}\|^2 + 2\|R\|^2 \ . \tag{9}$$

Thus, if the algorithm made m mistakes in T rounds, the matrix $\mathbf{M}$ satisfies,

$$\|\mathbf{M}\|^2 \leq 2m\|R\|^2 \ . \tag{10}$$

Combining Eq. (6) and Eq. (10), we have that, $m^2\gamma^2 \leq \|\mathbf{M}\|^2 \leq 2m\|R\|^2$, and therefore, $m \leq 2R^2/\gamma^2$. ■

We would like to note that the bound of the above theorem reduces to the perceptrons mistake bound in the binary case ($k = 2$). To conclude this section we analyze the non-separable case by generalizing Thm. 2 of Freund and Schapire [8] to a multiclass setting.

Theorem 2. *Let $(\bar{x}^1, y^1), \ldots, (\bar{x}^T, y^T)$ be an input sequence for any multiclass algorithm from the family described in Fig. 2, where $\bar{x}^t \in \mathbb{R}^n$ and $y^t \in \{1, 2, \ldots, k\}$. Denote by $R^2 = \max_t \|\bar{x}^t\|^2$. Let $\mathbf{M}^*$ be a prototype matrix of a unit vector-norm, $\|\mathbf{M}^*\| = 1$, and define, $d^t = \max\left\{0,\ \gamma - \left[\bar{M}^*_{y^t} \cdot \bar{x}^t - \max_{r \neq y^t} \bar{M}^*_r \cdot \bar{x}^t\right]\right\}$. Denote by $D^2 = \sum_{t=1}^T (d^t)^2$ and fix some $\gamma > 0$. Then the number of mistakes the algorithm makes is at most $2(R + D)^2/\gamma^2$.*

Proof. The case $D = 0$ follows from Thm. 1 thus we can assume that $D > 0$. The theorem is proved by transforming the non-separable setting to a separable one. To do so, we extend each instance $\bar{x}^t \in \mathbb{R}^n$ to $\bar{z}^t \in \mathbb{R}^{n+T}$ as follows. The first n coordinates of $\bar{z}^t$ are set to $\bar{x}^t$. The $n+t$ coordinate of $\bar{z}^t$ is set to Δ, which is a positive real number whose value is determined later; the rest of the coordinates of $\bar{z}^t$ are set to zero. We similarly extend the matrix $\mathbf{M}^*$ to $\mathbf{W}^* \in \mathbb{R}^{k \times (n+T)}$ as follows. We set the first n columns $\mathbf{W}^*$ to be $\frac{1}{Z}\mathbf{M}^*$. For each row r we set $W^*_{r,n+t}$ to $\frac{d^t}{Z\Delta}$ if $y^t = r$ and zero otherwise. We choose the value of Z so that $\|\mathbf{W}^*\|_2 = 1$, hence, $1 = \|\mathbf{W}^*\|_2^2 = \frac{1}{Z^2}\left(1 + \frac{D^2}{\Delta^2}\right)$ which gives that, $Z = \sqrt{1 + \frac{D^2}{\Delta^2}}$. We now show that $\mathbf{W}^*$ achieves a margin of $\frac{\gamma}{Z}$ on the extended data sequence. Note that for all r and t, $\bar{W}^*_r \cdot \bar{z}^t = \frac{1}{Z}\left(\bar{M}^*_r \cdot \bar{x}^t + \delta_{r,y^t} d^t\right)$. Now, using the definition of d^t we get,

$$\begin{aligned}
\bar{W}^*_{y^t} \cdot \bar{z}^t - \max_{r \neq y^t}\left\{\bar{W}^*_r \cdot \bar{z}^t\right\} &= \frac{1}{Z}\left(\bar{M}^*_{y^t} \cdot \bar{x}^t + d^t\right) - \max_{r \neq y^t}\left\{\frac{1}{Z}\left(\bar{M}^*_r \cdot \bar{x}^t\right)\right\} \\
&= \frac{1}{Z} d^t + \frac{1}{Z}\left[\bar{M}^*_{y^t} \cdot \bar{x}^t - \max_{r \neq y^t}\left\{\bar{M}^*_r \cdot \bar{x}^t\right\}\right] \\
&\geq \frac{1}{Z}\left(\gamma - \left[\bar{M}^*_{y^t} \cdot \bar{x}^t - \max_{r \neq y^t}\left\{\bar{M}^*_r \cdot \bar{x}^t\right\}\right]\right) \\
&\quad + \frac{1}{Z}\left[\bar{M}^*_{y^t} \cdot \bar{x}^t - \max_{r \neq y^t}\left\{\bar{M}^*_r \cdot \bar{x}^t\right\}\right] \\
&= \frac{\gamma}{Z} \; . \qquad (11)
\end{aligned}$$

We also have that,

$$\|\bar{z}^t\|^2 = \|\bar{x}^t\|^2 + \Delta^2 \leq R^2 + \Delta^2 \; . \qquad (12)$$

In summary, Eq. (11) and Eq. (12) imply that the sequence $(\bar{z}^1, y^1), \ldots, (\bar{z}^T, y^T)$ is classified correctly with margin $\frac{\gamma}{Z}$ and each instance $\bar{z}^t$ is bounded above by $R^2 + \Delta^2$. Thus, we can use Thm. 1 and conclude that the number of mistakes that the algorithm makes on $(\bar{z}^1, y^1), \ldots, (\bar{z}^T, y^T)$ is bounded from above by,

$$2\frac{R^2 + \Delta^2}{\left(\frac{\gamma}{Z}\right)^2} \; . \qquad (13)$$

Minimizing Eq. (13) over Δ we get that the optimal value for Δ is $\sqrt{DR}$ and the tightest mistake bound is, $2(D + R)^2/\gamma^2$. To complete the proof we show that the prediction of the algorithm in the extended space and in the original space are equal. Namely, let $\mathbf{M}^t$ and $\mathbf{W}^t$ be the value of the parameter matrix just before receiving $\bar{x}^t$ and $\bar{z}^t$, respectively. We need to show that the following conditions hold for $t = 1, \ldots, T$:

1. The first n columns of $\mathbf{W}^t$ are equal to $\mathbf{M}^t$.
2. The $(n+t)$th column of $\mathbf{W}^t$ is equal zero.
3. $\bar{M}^t_r \cdot \bar{x}^t = \bar{W}^t_r \cdot \bar{z}^t$ for $r = 1, \ldots, k$.

The proof of these conditions is straightforward by induction on t. ■

5 A norm-optimized multiclass algorithm

In the previous section we have described a family of algorithms where each algorithm of the family achieves the same mistake bound given by Thm. 1 and Thm. 2. This variety of equivalent algorithms suggests that there are some degrees of freedom that we might be able to exploit. In this section we describe an online algorithm that chooses a feasible vector $\bar{\tau}^t$ such that the vector-norm of the matrix $\mathbf{M}$ will be as small as possible.

```
Initialize: Set $\mathbf{M} \neq 0$  $\mathbf{M} \in \mathbb{R}^{k \times n}$.
Loop: For $t = 1, 2, \ldots, T$

  – Get a new instance $\bar{x}^t$.
  – Predict $\hat{y}^t = \arg\max_r \{\bar{M}_r \cdot \bar{x}^t\}$.
  – Get a new label $y^t$.
  – Find $\bar{\tau}^t$ that solves the following optimization problem:

        $\min_{\tau} \frac{1}{2} \sum_r \|\bar{M}_r + \tau_r \bar{x}^t\|_2^2$
        subject to : (1) $\tau_r \leq \delta_{r,y^t}$ for $r = 1, \ldots, k$
                     (2) $\sum_{r=1}^{k} \tau_r = 0$

  – Update : $\bar{M}_r \leftarrow \bar{M}_r + \tau_r^t \bar{x}^t$  for  $r = 1, 2, \ldots, k$ .

Output : $H(\bar{x}) = \arg\max_r \{\bar{M}_r \cdot \bar{x}\}$.
```

Fig. 3. The Margin Infused Relaxed Algorithm (MIRA).

To derive the new algorithm we omit the fourth constraint ($\tau_y = 1$) and thus allow more flexibility in choosing $\bar{\tau}^t$, or smaller changes in the prototype matrix. Previous bounds provides motivation for the algorithms in this section. We choose a vector $\bar{\tau}^t$ which minimizes the vector-norm of the new matrix $\mathbf{M}$ subject to the first two constraints only. As we show in the sequel, the solution of the optimization problem automatically satisfies the third constraint. The algorithm attempts to update the matrix $\mathbf{M}$ on *each* round regardless of whether there was a prediction error or not. We show below that the algorithm is ultraconservative and thus $\bar{\tau}^t$ is the zero vector if $\bar{x}^t$ is correctly classified (and no update takes place). Following the trend set by Li and Long [13] and Gentile [10], we term our algorithm MIRA for Margin Infused Relaxed Algorithm. The algorithm is described in Fig. 3.

Before investigating the properties of the algorithm, we rewrite the optimization problem that MIRA solves on each round in a more convenient form. Omitting the example index t the objective function becomes,

$$\frac{1}{2}\sum_r \|\bar{M}_r + \tau_r \bar{x}\|^2 = \frac{1}{2}\sum_r \|\bar{M}_r\|^2 + \sum_r \tau_r \left(\bar{M}_r \cdot \bar{x}\right) + \frac{1}{2}\sum_r \tau_r^2 \|\bar{x}\|^2 \quad .$$

Omitting $\frac{1}{2}\sum_r \|\bar{M}_r\|^2$ which is constant, the quadratic optimization problem becomes,

$$\min_{\tau} \quad \mathcal{Q}(\tau) = \frac{1}{2} A \sum_{r=1}^{k} \tau_r^2 + \sum_{r=1}^{k} B_r \tau_r \tag{14}$$
$$\text{subject to : } \forall r \quad \tau_r \leq \delta_{r,y} \quad \text{and} \quad \textstyle\sum_r \tau_r = 0$$

where,

$$A = \|\bar{x}\|^2 \quad \text{and} \quad B_r = \bar{M}_r \cdot \bar{x} \; . \tag{15}$$

Since $\mathcal{Q}$ is a quadratic function, and thus convex, and the constraints are linear, the problem has a unique solution.

We now show that MIRA automatically satisfies the third constraint of the family of algorithms from Sec. 4, which implies that it is ultraconservative. We use the following auxiliary lemma.

Lemma 2. *Let $\bar{\tau}$ be the optimal solution of the constrained optimization problem given by Eq. (14) for an instance-label pair $(\bar{x}, y)$. For each $r \neq y$ such that $B_r \leq B_y$ we have $\tau_r = 0$.*

The lemma implies that if a label r is not a source of error, then the rth prototype, $\bar{M}_r$, is not updated after $(\bar{x}, y)$ has been observed. In other words, the solution of Eq. (14) satisfies that $\tau_r = 0$ for all $r \neq y$ with $\left(\bar{M}_r \cdot \bar{x} \leq \bar{M}_y \cdot \bar{x}\right)$.

Corollary 1. *MIRA is ultraconservative.*

Proof. Let $(\bar{x}, y)$ be a new example fed to the algorithm. And let $\bar{\tau}$ be the coefficients found by the algorithm. From Lemma 2 we get that for each label r whose score $(\bar{M}_r \cdot \bar{x})$ is not larger than the score of the correct label $(\bar{M}_y \cdot \bar{x})$ its corresponding value τ_r is set to zero. This implies that only the indices which belong to the set $E \cup \{y\} = \{r \neq y : \bar{M}_r \cdot \bar{x} \geq \bar{M}_y \cdot \bar{x}\} \cup \{y\}$ may be updated. Furthermore, if the algorithm predicts correctly that the label is y, we get that $E = \emptyset$ and $\tau_r = 0$ for all $r \neq y$. In this case τ_y is set to zero due to the constraint $\sum_r \tau_r = \tau_y + \sum_{r \neq y} \tau_r = 0$. Hence, $\bar{\tau} = 0$ and the algorithm does not modify $\mathbf{M}$ on $(\bar{x}, y)$. Thus, the conditions required for ultraconservativeness are satisfied. ■

In Sec. 5.2 we give a detailed analysis of MIRA that incorporates the margin achieved on each example, and can be used to derive a mistake bound. Let us first show that the cumulative l_1-norm of the coefficients $\bar{\tau}^t$ is bounded.

Theorem 3. *Let $(\bar{x}^1, y^1), \ldots, (\bar{x}^T, y^T)$ be an input sequence to MIRA where $\bar{x}^t \in \mathbb{R}^n$ and $y^t \in \{1, 2, \ldots, k\}$. Let $R = \max_t \|\bar{x}^t\|^2$ and assume that there is a prototype matrix $\mathbf{M}^*$ of a unit vector-norm, $\|\mathbf{M}^*\| = 1$, which classifies the entire sequence correctly with margin $\gamma = \min_t \{\bar{M}^*_{y^t} \cdot \bar{x}^t - \max_{r \neq y^t} \bar{M}^*_r \cdot \bar{x}^t\} > 0$. Let $\bar{\tau}^t$ be the coefficients that MIRA finds for $(\bar{x}^t, y^t)$. Then, $\sum_{t=1}^T \|\bar{\tau}^t\|_1 \leq 4R^2/\gamma^2$.*

The proof is omitted due to lack of space.

5.1 Characteristics of the solution

Let us now further examine the characteristics of the solution obtained by MIRA. In [5] we investigated a related setting that uses error correcting output codes for multiclass problems. Using the results from [5] it is simple to show that the optimal $\bar{\tau}$ in Eq. (14) is given by

$$\tau_r = \min\{\theta^* - \frac{B_r}{A}, \delta_{y,r}\} \tag{16}$$

where $A = \|\bar{x}\|^2$ and $B_r = \bar{M}_r \cdot \bar{x}$ is the similarity-score of $(\bar{x}, y)$ for label r, as defined by Eq. (15). The optimal value θ^* is uniquely defined by the equality constraint $\sum_r \tau_r = 0$ of Eq. (14) and satisfies, $\sum_{r=1}^{k} \min\{\theta^* - \frac{B_r}{A}, \delta_{y,r}\} = 0$. We now can view MIRA in the following alternative light. Assume that the instance $(\bar{x}, y)$ was misclassified by MIRA and set $E = \{r \neq y \ : \bar{M}_r \cdot \bar{x} \geq \bar{M}_y \cdot \bar{x}\} \neq \emptyset$. The similarity-score for label r of the new matrix on the current instance $\bar{x}$ is,

$$\left(\bar{M}_r + \tau\,\bar{x}\right) \cdot \bar{x} = B_r + \tau_r A \ . \tag{17}$$

Plugging Eq. (16) into Eq. (17) we get that the similarity-score for class r on the current instance is, $\min\{A\theta^*, B_r + A\delta_{y,r}\}$. Since $\tau_r \leq \delta_{y,r}$, the maximal similarity score the updated matrix can attain on $\bar{x}$ is $B_r + A\delta_{r,y}$. Thus, the similarity-score for class r after the update is either a constant that is common to all classes, $A\theta^*$, or the largest similarity-score the class r can attain, $B_r + A\delta_{r,y}$. The constant $A\theta^*$ places an upper bound on the similarity-score for all classes after the update. This bound is tight, that is at least one similarity-score value is equal to $A\theta^*$.

5.2 Margin analysis of MIRA

In this section we further analyze MIRA by relating its mistake bound to the instantaneous margin of the individual examples. The margin analysis we present in this section sheds some more light on the source of difficulty in achieving a mistake bound for MIRA. Our analysis here also leads to an alternative version of MIRA that incorporates the margin into the quadratic optimization problem that we need to solve on each round. Our starting point is Thm. 3. We first give a lower bound on τ_y on each round. If MIRA made a mistake on $(\bar{x}, y)$, then we know that $\max_{r \neq y} B_r - B_y > 0$. Therefore, we can bound the minimal value of τ_y by a function of the (negative) margin, $B_y - \max_{r \neq y} B_r$.

Lemma 3. *Let $\bar{\tau}$ be the optimal solution of the constrained optimization problem given by Eq. (14) for an instance-label pair $(\bar{x}, y)$ with $A \leq R^2$. Assume that the margin $B_y - \max_{r \neq y} B_r$ is bounded from above by $-\beta$, where $0 < \beta \leq 2R^2$. Then τ_y is at least $\beta/(2R^2)$.*

We would like to note that for the above lemma if $\beta \geq 2R^2$ then $\tau_y = 1$ regardless of the margin achieved. We are now ready to prove the main result of this section.

Theorem 4. *Let $(\bar{x}^1, y^1), \ldots, (\bar{x}^T, y^T)$ be an input sequence to MIRA where $\bar{x}^t \in \mathbb{R}^n$ and $y^t \in \{1, 2, \ldots, k\}$. Denote by $R = \max_t \|\bar{x}^t\|$ and assume that there is a prototype matrix $\mathbf{M}^*$ of a unit vector-norm, $\|\mathbf{M}^*\|_2 = 1$, which classifies the entire sequence correctly with margin $\gamma = \min_t\{\bar{M}^*_{y^t} \cdot \bar{x}^t - \max_{r \neq y^t} \bar{M}^*_r \cdot \bar{x}^t\} > 0$. Denote by n_β the number of rounds for which $B_{y^t} - \max_{r \neq y^t} B_r \leq -\beta$, for some $0 < \beta \leq 2R^2$. Then the following bound holds, $n_\beta \leq 4R^4/(\beta\gamma^2)$.*

Proof. The proof is a simple application of Thm. 3 and Lemma 3. Using the second constraint of MIRA ($\sum_r \tau_r = 0$) and Thm. 3 we get that,

$$\sum_{t=1}^{T} \tau^t_{y^t} \leq 2\frac{R^2}{\gamma^2} \ . \tag{18}$$

From Lemma 3 we know that whenever $\max_{r \neq y^t} B_r - B_{y^t} \geq \beta$ then $1 \leq \frac{2R^2}{\beta}\tau^t_{y^t}$ and therefore,

$$n_\beta \leq \sum_{t=1}^{T} \frac{2R^2}{\beta} \tau^t_{y^t} \ . \tag{19}$$

Combining Eq. (18) and Eq. (19) we obtain the required bound,

$$n_\beta \leq 2\frac{R^2}{\beta} \sum_{t=1}^{T} \tau^t_{y^t} \leq 2\frac{R^2}{\beta} 2\frac{R^2}{\gamma^2} \leq 4\frac{R^4}{\beta\gamma^2} \ .$$

■

Note that Thm. 4 still does not provide a mistake bound for MIRA since in the limit of $\beta \to 0$ the bound diverges. Note also that for $\beta = 2R^2$ the bound reduces to the bounds of Thm. 1 and Thm. 3. The source of the difficulty in obtaining a mistake bound is rounds on which MIRA achieves a small negative margin and thus makes small changes to $\mathbf{M}$. On such rounds τ_y can be arbitrarily small and we cannot translate the bound on $\sum_t \tau^t_{y^t}$ into a mistake bound. This implies that MIRA is not robust to small changes in the input instances. We therefore describe now a simple modification to MIRA for which we can prove a mistake bound and, as we will see later, performs better empirically.

The modified MIRA aggressively updates $\mathbf{M}$ on every round for which the margin is smaller than some predefined value denoted again by β. This technique is by no means new, see for instance [13]. The result is a mixed algorithm which is both aggressive and ultraconservative. On one hand, the algorithm updates $\mathbf{M}$ whenever a minimal margin is not achieved, including rounds on which $(\bar{x}, y)$ is classified correctly but with a small margin. On the other hand, on each update of $\mathbf{M}$ only the rows whose corresponding similarity-scores are mistakenly too high are updated. We now describe how to modify MIRA along these lines.

To achieve a minimal margin of at least $\beta \leq 2R^2$ we modify the optimization problem given by Eq. (14). A minimal margin of β is achieved if for all r we require $\bar{M}_y \cdot \bar{x} - \bar{M}_r \cdot \bar{x} \geq \beta$ or, alternatively, $(\bar{M}_y \cdot \bar{x} - \beta) - (\bar{M}_r \cdot \bar{x}) \geq 0$. Thus, if we replace B_y with $B_y - \beta$, $\mathbf{M}$ will be updated whenever the margin is smaller than β. We thus let MIRA solve for each example $(\bar{x}, y)$ the following constrained optimization problem,

$$\begin{aligned} \min_{\tau} \quad & \mathcal{Q}(\tau) = \frac{1}{2}\tilde{A}\sum_{r=1}^{k} \tau_r^2 + \sum_{r=1}^{k} \tilde{B}_r \tau_r \\ \text{subject to :} \quad & \forall r \quad \tau_r \leq \delta_{r,y} \quad \text{and} \quad \textstyle\sum_r \tau_r = 0 \end{aligned} \tag{20}$$

$$\text{where :} \quad \tilde{A} = A = \|\bar{x}\|^2 \quad ; \quad \tilde{B}_r = B_r - \beta\delta_{y,r} = \bar{M}_r \cdot \bar{x} - \beta\delta_{y,r} \ .$$

To get a mistake bound for this modified version of MIRA we apply Thm. 4 almost verbatim by replacing B_r with $\tilde{B}_r$ in the theorem. Note that if $\tilde{B}_y - \max_{r \neq y} \tilde{B}_r \leq -\beta$ then $B_y - \beta - \max_{r \neq y} B_r \leq -\beta$ and hence $B_y - \max_{r \neq y} B_r \leq 0$. Therefore, for any $0 \leq \beta \leq 2R^2$ we get that the number of mistakes of the modified algorithm is equal to n_β which is bounded by $4R^4/\beta\gamma^2$. This gives the following corollary.

Corollary 2. *Let $(\bar{x}^1, y^1), \ldots, (\bar{x}^T, y^T)$ be an input sequence to the aggressive version of MIRA with margin β, where $\bar{x}^t \in \mathbb{R}^n$ and $y^t \in \{1, 2, \ldots, k\}$. Denote by $R = \max_t \|\bar{x}^t\|$ and assume that there is a prototype matrix $\mathbf{M}^*$ of a unit vector-norm, $\|\mathbf{M}^*\|_2 = 1$, which classifies the entire sequence correctly with margin $\gamma = \min_t\{\bar{M}^*_{y^t} \cdot \bar{x}^t - \max_{r \neq y^t} \bar{M}^*_r \cdot \bar{x}^t\} > 0$. Then, the number of mistakes the algorithm makes is bounded above by, $4R^4/(\beta\gamma^2)$.*

Note that the bound is a decreasing function of β. This means that the more aggressive we are by requiring a minimal margin the smaller the bound on the number of mistakes the aggressively modified MIRA makes. However, this also implies that the algorithm will update $\mathbf{M}$ more often and the solution will be less sparse.

6 Discussion and current research directions

In this paper we described a general framework for deriving ultraconservative algorithms for multiclass categorization problems and analyzed the proposed algorithms in the mistake bound model. We investigated in detail an additive family of online algorithms. The entire family reduces to the perceptron algorithm in the binary case. In addition, we gave a method for choosing a unique member of the family by imposing a quadratic objective function that minimizes the norm of the prototype matrix after each update. Note that the similarity-score of a new instance is a linear combination of inner-products between pairs of instances. Therefore, all the algorithms we presented can be straightforwardly combined with kernel methods [18].

An interesting direction we are currently working on is combining our framework with other online learning algorithms for binary problems. Specifically, we have been able to combine Winnow [14] and Li and Long's ROMMA algorithm [13] with our framework, and to construct a multiclass version for those algorithms. A question that remains open is how to impose constraints similar to the one MIRA employs in both cases.

An interesting question in this case is whether our framework can be combined with the family of quasi-additive binary algorithms of Grove, Littlestone and Schuurmans [11] and other p-norm algorithms [10]. Another interesting direction that generalizes our framework is algorithms that maintain multiple prototypes per class. It is not clear in this case what form the updates should take.

We have performed preliminary experiments on synthetic data that show the advantages in using our proposed framework over previously studied online algorithms for multiclass problems. A complete description of the results obtained in the various experiments will appear in a forthcoming full version.

A current research direction we are pursuing is methods for reducing the number of updates MIRA performs, and thus the number of instances that are used in building the classifier. We are currently working on the design of o post processing stage of the algorithms. Combining the algorithms presented in this paper with a post processing can be used in batch setting. Preliminary results show that this direction may give a viable algorithmic alternative for building support vector machines.

Acknowledgements We would like to thank Elisheva Bonchek for carefully reading a draft of the manuscript and to Noam Slonim for useful comments.

References

1. E.L. Allwein, R.E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. In *Machine Learning: Proceedings of the Seventeenth International Conference*, 2000.
2. J. K. Anlauf and M. Biehl. The adatron: an adaptive perceptron algorithm. *Europhysics Letters*, 10(7):687–692, Dec 1989.
3. Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and Regression Trees*. Wadsworth & Brooks, 1984.
4. Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, September 1995.
5. Koby Crammer and Yoram Singer. On the learnability and design of output codes for multiclass problems. In *Proceedings of the Thirteenth Annual Conference on Computational Learning Theory*, 2000.
6. T.G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *J. of Artificial Intelligence Research*, 2:263–286, 1995.
7. R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. Wiley, 1973.
8. Yoav Freund and Robert E. Schapire. Large margin classification using the perceptron algorithm. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, 1998. To appear, *Machine Learning*.
9. Thilo Friess, Nello Cristianini, and Colin Campbell. The kernel-adatron: A fast and simple learning procedure for support vector machines. In *Machine Learning: Proceedings of the Fifteenth International Conference*, 1998.
10. Claudio Gentile. Approximate maximal margin classification with respect to an arbitrary norm. In *Advances in Neural Information Processing Systems 14*, 2000.
11. Adam J. Grove, Nick Littlestone, and Dale Schuurmans. General convergence results for linear discriminant updates. In *Proceedings of the Tenth Annual Conference on Computational Learning Theory*, 1997.
12. Jyrki Kivinen and Manfred K. Warmuth. Additive versus exponentiated gradient updates for linear prediction. *Information and Computation*, 132(1):1–64, 1997.
13. Yi Li and Phil M. Long. The relaxed online maximum margin algorithm. In *Advances in Neural Information Processing Systems 13*, 1999.
14. Nick Littlestone. Learning when irrelevant attributes abound. In *28th Annual Symposium on Foundations of Computer Science*, pages 68–77, October 1987.
15. J.C. Platt. Fast training of Support Vector Machines using sequential minimal optimization. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1998.
16. J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
17. F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958.
18. Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley, 1998.