A Preference Model for Structured Supervised Learning Tasks

Fabio Aiolli
Dip. di Matematica Pura e Applicata, Università di Padova,
Via G. Belzoni 7, 35131 Padova, Italy,
aiolli@math.unipd.it

Abstract

The preference learning model introduced in this paper gives a natural framework and a large-margin principled solution for a broad class of supervised learning problems with structured predictions, including ranking-based predictions (label and instance ranking), hierarchical classification, and ordinal regression. We show how all these problems can be cast as linear problems in an augmented space, and we propose a stochastic gradient method to efficiently solve them. Experiments performed on an ordinal regression task confirm the generality and the effectiveness of the approach.

Keywords: *Supervised Learning, Ranking, Ordinal Regression, Preferences.*

1 Introduction

Supervised learning deals with algorithms that give machines the ability to learn from experience. Many real-world learning problems are characterized by heterogeneous tasks which currently cannot be solved by general-purpose algorithms. These include ranking-based problems (either label or instance ranking) and ordinal regression. The typical approach followed to cope with these complex problems is to map them into a series of simpler, well-known settings and then to combine the resulting predictions. Often, these solutions lack a principled theory and/or require too much computational resources to be practical for data-mining applications.

Although some efforts have been recently made to generalize label ranking tasks [7, 5, 2], a general framework and a theory encompassing all these supervised learning settings is missing. In this paper we propose a quite detailed taxonomy of supervised learning problems, based on the different type of predictions and the supervision involved. Then, we show how supervision can be seen as a set of order preferences over the predictions of the learner. Finally, we show how all these problems can be seen as linear binary problems defined on an augmented space, thus suggesting very simple optimization procedures available for the binary case.

Another contribution of this paper is to define a preference model which is very flexible and allows a user to optimize the parameters on the basis of a proper evaluation function. Often, while the goal of a problem in terms of its evaluation function is clear, a crucial thing in the design of learning algorithms is how to define them in such a way to have some theoretical guarantee that a learning procedure leads to the effective minimization of that particular cost function. The model introduced in this paper gives a natural and uniform way to encode the cost function of a supervised learning problem and plug it into a learning algorithm.

In Section 2, starting from a definition of a detailed taxonomy of supervised learning tasks, the proposed learning model is presented. Examples of instantiations of the model to supervised learning problems is presented in Section 3. In Section 4, we propose principled batch and efficient online optimization procedures for training the model. Finally, in Section 5 the experimental results are presented.

2 A Model for Supervised Learning

In supervised learning, we assume supervision is provided according to an unknown probability distribution $\mathcal{D}$. Generally, it consists of pairs, example and corresponding correct prediction. For reasons that will be clearer in the following, we prefer to consider supervision as (soft) constraints over the learner predictions, that is constraints whose violation entails a cost for the solution. Specifically, assuming a learner makes its predictions on the basis of a set of parameters Θ, characterizing its *hypothesis space*, supervision S makes the learner suffering a cost $c(S|\Theta)$. This subsumes the case of supervision as pairs previously pointed out. In fact, this is obtained when a unitary cost is given to hypotheses generating an incorrect labeling.

Two main settings of learning can be identified. In the *on-line* paradigm, learning takes place in rounds. At each step the learner receives supervision and updates its parameters with the aim to minimize future costs. In *batch* learning a training set $\mathcal{S} = \{S_1, \ldots, S_n\}$ is available where the S_i are supposed to be drawn *i.i.d.* from $\mathcal{D}$ and a single training session is made with the explicit goal to minimize the expected cost on the true distribution $\mathcal{D}$.

Different learning problems are often characterized by different types of prediction and supervision. Nevertheless, we will show that a broad set of them can be studied in a common framework, whose general setting is as follows. We consider a space $\mathcal{X}$ of instances and a space $\mathcal{Y}$ of labels (both sets possibly infinite). Moreover, we assume the hypothesis space, based on which the learner makes its predictions, to consist of *relevance functions*

$$u : \mathcal{X} \times \mathcal{Y} \to \mathbb{R},$$

depending on some set of parameters Θ. The goal of the learner is then to select a function $\hat{u}$ from its hypothesis space, which is "consistent" with the supervision in a sense that will depend on the particular setting.

2.1 Prediction and Supervision

In this section, we present a detailed taxonomy of the main supervised learning tasks organized on the basis of the required predictions and supervision. To this end, we first need to define order relations.

A *partial order* is a pair $(\mathcal{P}, \succeq)$ where $\mathcal{P}$ is a set and $\succeq$ is a reflexive, antisymmetric and transitive binary relation. A *partial ranking* of length r is a partial order where the set $\mathcal{P}$ can be partitioned in r sets $\mathcal{P}_1, \ldots, \mathcal{P}_r$ such that $z \in \mathcal{P}_i$, $z' \in \mathcal{P}_j$, $i < j$, implies $z \succeq z'$ and no further information is conveyed about the ordering within subsets $\mathcal{P}_k$. A *full order* on $\mathcal{P}$ is defined as a partial ranking of length $|\mathcal{P}|$. We denote by $PO(\mathcal{P})$, $PR(\mathcal{P})$, and $FO(\mathcal{P})$ the set of partial orders, partial rankings and full orders over the set $\mathcal{P}$, respectively.

2.1.1 Label Rankings

A first important family of supervised learning tasks is related to the ordering of the classes on the basis of their relevance for an instance. This family of problems is referred to as *label rankings*. Problems in this family take supervision in the form of general partial orders over the classes. In our notation, given $\mathbf{x} \in \mathcal{X}$, $Y \subseteq \mathcal{Y}$, we have $S \in PO(Y)$ and predictions are in $FO(Y)$. A few well-known instances are listed in the following:

Category Ranking (CR) In this setting, the goal is to order categories on the basis of their relevance for an instance. As an example, in a collaborative filtering setting, users could correspond to our instances and the different movies to our classes. Then, one could be interested into the ordering (by relevance) of the set of movies based on user preferences. This is trivially a particular case of label ranking where supervision is given as full orders over Y.

Bipartite Category Ranking (BCR) In this task, supervision is given as two groups of classes and it is required to predict full orders in which the first group of classes is ranked over the second. As a leading example, in information retrieval, given a document, one might have to rank the available topics with the aim to return the most relevant topics on the top of the list. This is again a specific case of label ranking where supervision is given as partial rankings of length two. This task has been also referred to as category ranking in literature [4]. Here a different terminology is adopted to avoid confusion between these two different tasks.[1]

Sometimes, we are also interested in predictions consisting of the most relevant classes, that is, of a prefix of the full order induced by the relevance function $u(\mathbf{x}, y)$. This family of tasks is commonly referred to as *classification* problems. They can however be considered as subcases of the BCR ranking task. A few examples of this kind of problems, listed by increasing specificity, is given here:

q-label classification (QC) In this task, the goal is to select the q most appropriated classes for a given instance, with q fixed. The supervision here is a partial ranking of length two where a set of exactly q labels are preferred over the rest.

Single-label classification (SC) In this well-known classification task, the goal is to select exactly one class (the most relevant) for an instance. This is a trivial subcase of QC with $q = 1$.

2.1.2 Instance Rankings

Another interesting family of tasks is *instance rankings* where the goal is to order instances on the basis of the relevance of a given class. In our notation, given $y \in \mathcal{Y}$, $X \subseteq \mathcal{X}$, prediction is in $FO(X)$ and supervision is given in the form $S \in PO(X)$.

The duality with respect to label rankings is self-evident. In principle, a corresponding problem setting could be defined for each of the label ranking settings. We can easily

[1] Note that this task and the two that follow, are conceptually different from the task to decide about the membership of an instance. Here, supervision only gives *qualitative* information about the fact that some classes are more relevant than others.

see that the well-known task, commonly known as *(Bipartite) Instance Ranking* (IR), corresponds to BCR and is the one to induce an order such that a given set of instances is top-ranked. A natural application of this kind of prediction is in information retrieval, e.g. when listing the results returned by a search engine. Similarly to BCR, here supervision consists of partial rankings (this time over the set X) of length two. Another interesting task which can be considered in this family is the one to learn preference relations from a given set of ranked instances. For example, an information retrieval task is that to learn the preference relations on the basis of basic preferences given as pairs of documents [8].

The two families of tasks above can be considered *qualitative tasks* since they are concerned with order relations between instance-class pairs. On the other side, *quantitative tasks* are the ones which are more concerned with the absolute values of the relevance of instance-class pairs.

2.1.3 Quantitative Predictions

Sometimes there is the necessity to do quantitative predictions about data at hand. For example, in binary classification, one has to decide about the membership of an instance to a class as opposed to rank instances by relevance. These settings are not directly subsumed by the settings presented above. As we will see this can be overcome by adding a set of thresholds and doing predictions based on these thresholds.

***Multivariate Ordinal Regression* (MOR)** There are many settings where it is natural to rank instances according to an ordinal scale, including collaborative filtering, where there is the need to predict people ratings on unseen items. Borrowing the movie-related application introduced above, suitable ranks for movies could be given as 'bad', 'fair', 'good', and 'recommended'. With no loss in generality, we can consider the target space as the integer set $\mathcal{Z} = \{0, \ldots, R-1\}$ of R available ranks. Following an approach similar to the one in [10], ranks are made corresponding to intervals of the real line. Specifically, a set of thresholds $T = \{\tau_0 = -\infty, \tau_1, \ldots, \tau_{R-1}, \tau_R = +\infty\}$ is defined and the prediction is based on the rule

$$\hat{z} = \{i : u(\mathbf{x}, y) \in (\tau_{i-1}, \tau_i)\}.$$

Given the target label z, a correct prediction will be consistent with the conditions: $u(\mathbf{x}, y) > \tau_i$ when $i < z$ and $u(\mathbf{x}, y) < \tau_i$ when $i \geq z$.

The well-known *(Univariate) Ordinal Regression*(OR) [9, 12] task is a trivial subcase of MOR when a single class is available.

***Multi-label Classification* (MLC)** In this task, it is required to classify instances with a subset (the cardinality of which is not specified) of the available classes. For us, it is convenient to consider this task as a MOR problem where only two ranks are available, relevant and irrelevant, and $\mathcal{Z} = \{0, 1\}$.

The well-known *Binary Classification* (BC) can be considered a subcase of OR with two ranks $\mathcal{Z} = \{0, 1\}$. Note that this task is considered here conceptually different from SC with two classes.

Clearly, the taxonomy presented above is not exhaustive but highlights how many different kinds of structured supervision can be seen as simple constraints over the predictions of a learner. Specifically, they consist of constraints in conjunctive form (here referred to as *preference sets*, or p-sets) where each basic preference is defined over the scoring values and/or some threshold value.

In particular, we can differentiate between two types of order preferences: *qualitative* preferences in the form

$$(u(\mathbf{x}_i, y_r), u(\mathbf{x}_j, y_s))$$

telling that the value of $u(\mathbf{x}_i, y_r)$ should be higher than the value of $u(\mathbf{x}_j, y_s)$, and *quantitative* preferences in the form

$$(u(\mathbf{x}, y), \tau) \text{ or } (\tau, u(\mathbf{x}, y)), \tau \in \mathbb{R}$$

relating the value of $u(\mathbf{x}, y)$ to a given threshold τ.

In Table 1, a summary of supervision obtained for the most general settings are presented. Particular instantiations to more specific problems are immediate anyway.

Setting	Supervision P-sets
LR	$\{(u(\mathbf{x}, y_r), u(\mathbf{x}, y_s))\}_{(\mathbf{x}, y_r) \succeq_S (\mathbf{x}, y_s)}$
IR	$\{(u(\mathbf{x}_i, y), u(\mathbf{x}_j, y))\}_{(\mathbf{x}_i, y) \succeq_S (\mathbf{x}_j, y)}$
MOR	$\{(u(\mathbf{x}, y), \tau_i)\}_{i<z} \cup \{(\tau_i, u(\mathbf{x}, y))\}_{i \geq z}$

Table 1. Supervision of problems in Section 2.1. Label and instance rankings (LR and IR respectively), have a preference for each order relation induced by the supervision S. In ordinal regression (MOR), a preference is associated to each threshold and $z \in \mathcal{Z}$ is the rank given by the supervision.

2.2 A Model for the Learner

In the following, we will focus on a particular form of the relevance function, that is

$$u(\mathbf{x}, y) = w \cdot \phi(\mathbf{x}, y)$$

where $\phi(\mathbf{x}, y) \in \mathbb{R}^d$ is a joint representation of instance-class pairs and $w \in \mathbb{R}^d$ is a weight vector [11]. Note that this form encompasses the more standard form $u(\mathbf{x}, y) = w_y \cdot \phi(\mathbf{x})$ which has a weight vector for each different label. In fact, if $|\mathcal{Y}| = m$, we can write:

$$w = (w_1, \ldots, w_m)$$

and

$$\phi(\mathbf{x}, y) = (\underbrace{\mathbf{0}, \ldots, \mathbf{0}}_{y-1}, \phi(\mathbf{x}), \mathbf{0}, \ldots, \mathbf{0}).$$

With this assumption, it is possible to conveniently reformulate an order constraint as a linear constraint. Let $T = \{\tau_1, \ldots, \tau_{R-1}\}$ be the available thresholds, in the qualitative case, given $a \equiv (u(\mathbf{x}_i, y_r), u(\mathbf{x}_j, y_s))$, we obtain

$$u(\mathbf{x}_i, y_r) > u(\mathbf{x}_j, y_s) \Leftrightarrow$$
$$(w, \tau_1, \ldots, \tau_{R-1}) \cdot \underbrace{(\phi(\mathbf{x}_i, y_r) - \phi(\mathbf{x}_j, y_s), \underbrace{0, \ldots, 0}_{R-1})}_{\psi(a)} > 0$$

Viceversa, in the quantitative case, given $\delta \in \{-1, +1\}$, we have

$$\delta(u(\mathbf{x}, y) - \tau_r) > 0 \Leftrightarrow$$
$$(w, \tau_1, \ldots, \tau_{R-1}) \cdot \underbrace{(\delta\phi(\mathbf{x}, y), \underbrace{0, \ldots, 0}_{r-1}, -\delta, \underbrace{0, \ldots, 0}_{R-r-1})}_{\psi(a)} > 0.$$

In general, we can see that supervision constraints of all the problems discussed above, can be reduced into sets of particular linear preferences of the form $\mathbf{w} \cdot \psi(a) > 0$ where $\mathbf{w} = (w, \tau_1, \ldots, \tau_{R-1})$ is the vector of weights augmented with the set of available thresholds and $\psi(a)$ is an opportune representation of the preference under consideration.

The quantity

$$\rho_A(a|\mathbf{w}) = \mathbf{w} \cdot \psi(a)$$

will be also referred to as the margin of the hypothesis w.r.t. the preference. Note that this value is greater than zero when the preference is satisfied and less than zero otherwise. We will say that a preference a is *consistent* with an hypothesis when $\rho_A(a|\mathbf{w}) > 0$ (and we write $a \sqsubset \mathbf{w}$). The margin of an hypothesis w.r.t. the whole supervision S, can be consequently defined as the minimum of the margins of preferences in $g[S]$, i.e.

$$\rho(g[S]) = \min_{a \in g[S]} \rho_A(a).$$

This definition turns out to be consistent with definitions of the margin commonly used in different problems. In particular, the margin is positive if and only if the prediction is consistent with the supervision.

Summarizing, all the problems defined in the taxonomy in Section 2.1 can be seen as an homogeneous linear binary problem in a opportune augmented space. Specifically, any algorithm for linear classification (e.g. perceptron or linear programming) can be used to solve it, provided the problem has a solution.

2.3 Evaluation and GPLM

The mere consistency of supervision constraints is not necessarily the ultimate goal of a supervised learning setting. Rather, cost functions are often preferred measuring the disagreement between the current hypothesis and the supervision. These functions may either depend on the particular structure of the prediction or other factors.

In [2] a general model for label rankings has been proposed. Here, we extend the same idea to general supervised settings by mapping supervision into sets of preferences with costs. We will refer to this method as *Generalized Preference Learning Model* (or simply GPLM).

Definition 2.1 Preference Sets w/ Costs *A* (conjunctive) preference set with costs, *or simply "*cp-set*", is a p-set where preferences have costs associated. Preferences of a cp-set will be denoted by* $a_{\gamma(a)}$. *When the cost is not indicated* $\gamma(a) = 1$ *will be considered.*

With this definition in mind, given a cp-set g, an hypothesis suffers a cost which is defined as the maximum among the costs of its unfulfilled preferences, i.e.

$$c(g|\mathbf{w}) = \max_{a \in g, a \not\sqsubset \mathbf{w}} \gamma(a). \tag{1}$$

In GPLM, we consider supervision S as a p-set $g[S]$ and we consider a cost mapping

$$\mathcal{G} : g[S] \mapsto \{g_1(S), \ldots, g_{q_S}(S)\}$$

where each cp-set $g_i(S)$ is a subset of $g[S]$ with some costs assigned to the preferences.

Once the cost mapping $\mathcal{G}$ is fixed, the total cost suffered by an hypothesis $\mathbf{w}$ for the supervision S is defined as the cumulative cost of cp-sets, i.e.

$$c(g[S]|\mathbf{w}) = \sum_{j=1}^{q_S} c(g_j(S)|\mathbf{w}). \tag{2}$$

Let g_p be a p-set, natural mappings already proposed in [5] for preference graphs can be easily adapted to our setting. This is made by considering classes of equivalence among preferences and by defining mappings in which a different cp-set is built for each partition. Specifically, let $a \equiv (a_s, a_e)$ and $a' \equiv (a'_s, a'_e)$ denote a pair of preferences, we have the following:

(i) the *identity mapping*, denoted by $\mathcal{G}_I$, where g_p is mapped on a single cp-set g_c. This corresponds to define the trivial equivalence relation $(a_s, a_e) \equiv (a'_s, a'_e)$;

(ii) the *domination mapping*, denoted by $\mathcal{G}_D$, where g_p is split into a set of cp-sets on the basis of the equivalence relation $(a_s, a_e) \equiv (a'_s, a'_e) \Leftrightarrow a_s = a'_s$;

(ii) the *dominated mapping*, denoted by $\mathcal{G}_{dom}$, where g_p is split into a set of cp-sets on the basis of the equivalence relation $(a_s, a_e) \equiv (a'_s, a'_e) \Leftrightarrow a_e = a'_e$;

(iv) the *disagreement mapping*, denoted by $\mathcal{G}_d$, where g_p is split into a set of cp-sets on the basis of equivalence relations $(a_s, a_e) \equiv (a'_s, a'_e) \Leftrightarrow a_s = a'_s \wedge a_e = a'_e$.

3 Examples of GPLM Cost Mappings

In this section, a set of examples of supervised learning problems and suitable GPLM cost mappings are discussed. In particular, we show how general the models is and how many common cost functions can be defined with the tools offered by our model.

3.1 Cost functions for Label Rankings.

Basic mappings for rankings and classification can be found in [2] and can be reproduced with the model proposed in this paper. In fact, it can be shown quite easily that, for label rankings, PLM preference graphs and GPLM cp-sets with unitary costs are equivalent.

Applying these simple mappings we are able to reproduce many of the different losses used for ranking problems. For example, the cost mapping $\mathcal{G}_D$ seems particularly suitable for q-label classification since it gives a 'soft' indication of *how many* relevant labels are wrongly classified as irrelevant. On the other side, the $\mathcal{G}_I$ mappings gives a cost function which returns a binary value indicating if any of the relevant labels are wrongly classified as irrelevant. The multiclass loss commonly used for single-label classification can be obtained using the $\mathcal{G}_I$ mapping. Note that, using the $\mathcal{G}_d$ mapping would have lead to a cost function which returns the number of incorrect classes which have a relevance higher than the relevance of the correct class. Another example is the so called *ranking loss* that has been proposed in [4] for the binary category ranking problem. This cost function corresponds to the number of pairs which are not correctly ordered and corresponds to the cost mapping $\mathcal{G}_d$.

However, the extension presented here introduces far more flexibility on the choice of the cost function for label rankings because of the use of cp-sets in place of preference graphs.

A typical example is classification where misclassifications can have different costs. This can be the case in single-label classification when categories are not represented with the same frequencies in the training and the test set. Another interesting case is when there is some structure between the available classes and a different metric for misclassification costs is introduced. For example, in hierarchical classification, it makes sense to pay costs proportional to the path length in the tree between the true class and the predicted one. In all these cases, a cost matrix Δ is used to have a better control over the learning algorithm, where the element $\Delta(y_r, y_s)$ represents the cost of classifying a pattern as y_r when it is actually in y_s. In our model, the same can be easily obtained by associating costs to cp-sets of the GPLM mapping.

3.2 Cost functions for Instance Rankings

A common loss function used in IR is the so called AUC (Area under ROC curve) measure. It can be shown that it directly derives using the cost mapping $\mathcal{G}_d$. Interestingly, our model suggests new possible settings and loss definitions one might use for the tasks in the family of instance rankings.

3.3 Cost functions for Ratings

A brief review of standard loss functions used for rating tasks and the implementation in our model is now presented.

Ordinal Regression Recalling the natural definition of cost for ordinal regression problems, i.e. $c = |\hat{z}(\mathbf{x}) - z(\mathbf{x})|$, where $\hat{z}(\mathbf{x})$ is the rank given as output by the hypothesis and $z(\mathbf{x})$ the correct rank, we would like to define a cost mapping for GPLM consistent with the same cost function.

At least two different cost mappings have this property. The easiest one is the mapping $\mathcal{G}_d$. In this case, the resulting cost will be the number of thresholds which are not correctly ordered w.r.t. $u(\mathbf{x}, y)$. This is exactly the cost as given before. A second possibility is to define a mapping $\mathcal{G}_I$ followed by an assignment of costs where the r-th preference is set to $(u(\mathbf{x}, y), \tau_r)_{z-i+r}$ whenever $r \leq z$, and $(\tau_r, u(\mathbf{x}, y))_{r-z}$ otherwise.

As an example of this second situation, consider a $R = 4$ univariate ordinal regression problem. Then, we have three thresholds $T = \{\tau_1, \tau_2, \tau_3\}$ and cost mappings defined as in the following:

$$\begin{aligned}
\mathcal{G}(g[0]) &= \{(\tau_1, u(\mathbf{x}, y))_1, (\tau_2, u(\mathbf{x}, y))_2, (\tau_3, u(\mathbf{x}, y))_3\} \\
\mathcal{G}(g[1]) &= \{(u(\mathbf{x}, y), \tau_1)_1, (\tau_2, u(\mathbf{x}, y))_1, (\tau_3, u(\mathbf{x}, y))_2\} \\
\mathcal{G}(g[2]) &= \{(u(\mathbf{x}, y), \tau_1)_2, (u(\mathbf{x}, y), \tau_2)_1, (\tau_3, u(\mathbf{x}, y))_1\} \\
\mathcal{G}(g[3]) &= \{(u(\mathbf{x}, y), \tau_1)_3, (u(\mathbf{x}, y), \tau_2)_2, (u(\mathbf{x}, y), \tau_3)_1\}
\end{aligned}$$

It is easy to verify that this mapping respects the costs as they could be obtained by the natural cost definition given

above. For example, considering the instance $\mathbf{x}$ with target rank 1 being ranked 3. Then, it means that the scoring function is such that $u(\mathbf{x}, y|\Theta) \in (\tau_3, +\infty)$, i.e.

$$-\infty \leq \tau_1 \leq \tau_2 \leq \tau_3 \leq u(\mathbf{x}, y|\Theta) \leq +\infty,$$

and hence the cost suffered by the hypothesis is correctly computed by $c(g[1]|\Theta) = \max\{0, +1, +2\} = +2$.

Binary Classification The natural cost function for BC problems is trivially obtained by using $\mathcal{G}_I$, i.e. by setting $(u(\mathbf{x}, y), \tau)$ when $\mathbf{x}$ is a positive example for the class, and $(\tau, u(\mathbf{x}, y))$ otherwise.

Very similar examples, omitted for space reasons, can be given for the multi-variate versions of ratings problems.

4 Learning in GPLM

In earlier sections we have discussed the structure behind the supervision and how it can be modelled using cp-sets. Now, we see how to give learning algorithms for the batch and the on-line settings.

In GPLM we propose to minimize costs $c(S|\mathbf{w})$. Since these are not continuous w.r.t. $\mathbf{w}$, we approximate them by introducing a continuous non-increasing function $l : \mathbb{R} \to \mathbb{R}^+$ approximating the indicator function. Then, we define the approximate cost

$$\tilde{c}(S|\mathbf{w}) = \sum_{g \in \mathcal{G}(g[S])} \max_{a \in g} \gamma(a) l(\rho_A(a|\mathbf{w})).$$

Examples of losses one can use are presented in Table 2.

Methods	$l(\rho)$
Perceptron	$\max(0, -\rho)$
β-margin	$\max(0, \beta - \rho)$
Exponential	$e^{-\rho}$
Sigmoidal	$(1 + e^{\lambda(\rho-\theta)})^{-1}$

Table 2. Approximation losses as a function of the margin. $\beta > 0, \lambda > 0, \theta \in \mathbb{R}$ are external parameters.

4.1 Batch Learning for GPLM

The goal in batch learning is to find the parameters $\mathbf{w}$ such to minimize the expected cost over $\mathcal{D}$, the actual distribution ruling the supervision feed, which is defined by

$$R_t[\mathbf{w}] = E_{S \sim \mathcal{D}}[c(g[S]|\mathbf{w})].$$

Although $\mathcal{D}$ is unknown, we can still try to minimize this function by exploiting the same structure of supervision and as much of the information we can gather from the training set. The general problem can be given as in the following:

- Given a set $\mathcal{V}(\mathcal{S}) = \bigcup_{S \in \mathcal{S}} g[S]$ of cp-sets
- Find a set of parameters $\mathbf{w}$ in such a way to minimize the functional

$$\mathcal{Q}(\mathbf{w}) = \mathcal{L}(\mathcal{V}(\mathcal{S})|\mathbf{w}) + \mu\mathcal{R}(\mathbf{w}) \qquad (3)$$

  where $\mathcal{L}(\mathcal{V}(\mathcal{S})|\mathbf{w}) = \sum_{S \in \mathcal{S}} \tilde{c}(S|\mathbf{w})$ is related to the empirical cost and $\mathcal{R}(\mathbf{w})$ is a regularization term over the set of parameters. Note that, for the solution to be admissible when multiple thresholds are used and there are constraints defined over their values (as in the ordinal regression settings), these constraints should be explicitly enforced.

The use of a regularization term on a problem of this type has many different motivations, including the theory on regularization networks (see e.g. [6]). However, given the huge amount of data available in many data-mining applications, this term can usually be disregarded without affecting the performance.

Moreover, we can see that by choosing a convex loss function and a convex regularization term (let say the quadratic term $\mathcal{R}(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2$) it warranties the convexity of the functional $\mathcal{Q}(\mathbf{w})$ in Eq. 3 and then the uniqueness of the solution. Indeed, current kernel-based approaches defined for basic supervised learning tasks can be seen in this form when using the β-margin with $\beta = 1$. This suggests a new *universal* kernel method which is able to solve many complex learning tasks [1].

Given the large amount of examples in data-mining applications, a drawback of this learning setting is the onerous computational requirements. In the following section, a principled stochastic approximation is presented aiming at minimizing the same functional efficiently.

4.2 Stochastic On-line Learning for GPLM

As already pointed out, in on-line learning, supervision becomes available one by one and each time the learner updates the hypothesis to minimize future costs. A suitable measure of performance after m rounds is the *cumulative cost* function

$$R_t^m[\mathbf{w}] = \sum_{i=1}^{m} c(g[S_i]|\mathbf{w}_i)$$

where $\mathbf{w}_i$ is the hypothesis obtained after seeing supervision $S_1, \ldots, S_{i-1}$.

Following a typical approach for on-line learning, we propose to perform a stochastic gradient descent [13] with

respect to the instantaneous cost $\mathcal{Q}(\mathbf{w}_t) = \tilde{c}(S_t|\mathbf{w}_t)+\mu\mathcal{R}(\mathbf{w}_t)$. Then, assuming $\mathcal{R}(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2$, the update will be in the form $\mathbf{w}_t = \mathbf{w}_t - \lambda\mathcal{Q}'_{\mathbf{w}}$ and

$$\begin{aligned}\mathcal{Q}'_{\mathbf{w}} &= \mathbf{w} + \textstyle\sum_{g\in\mathcal{G}(g[S_t])} \gamma(\hat{a}[g]) l'_\rho(\rho(\hat{a}[g])) \rho'_{\mathbf{w}}(\hat{a}(g)) \\ &= \mathbf{w} + \textstyle\sum_{g\in\mathcal{G}(g[S_t])} \gamma(\hat{a}[g]) l'_\rho(\rho(\hat{a}[g])) \psi(\hat{a}[g])\end{aligned}$$

where $\hat{a}[g] = \arg\max_{a\in g} \gamma(a) l(\rho(a))$ and $f'_x(v)$ stands for the gradient of f w.r.t. the parameters x evaluated in v.

It can be easily shown that this update rule makes the weight vector w taking the (sparse) form:

$$w = \sum_{i,r} \alpha_i^r \phi(\mathbf{x}_i, y_r)$$

where $\alpha_i^r \in \mathbb{R}$, thus obtaining an (sparse) implicit representation of the relevance function as:

$$u(\mathbf{x}, y) = \sum_{i,r} \alpha_i^r \phi(\mathbf{x}_i, y_r)\phi(\mathbf{x}, y).$$

As in the batch setting, here we have the problem to enforce the *hard* constraints defined over the thresholds. However, this is a far less stringent issue in a stochastic method. In our implementation, this is made by projecting the values of the thresholds back into the constraint after each iteration.

5 Experiments

To demonstrate the flexibility and validate the general model proposed in this paper, we performed a set of experiments on a synthetic dataset. The explicit purpose was the one to try different cost mappings and loss functions in a relatively self-contained task in such a way to have a better control and to do fair comparisons between different configurations.

In particular, we have considered an ordinal regression problem ($|\mathcal{Y}| = 1$) in the online paradigm. Since $|\mathcal{Y}| = 1$, in this case we have that the relevant function is $u(\mathbf{x}) = w\cdot\phi(\mathbf{x})$. Moreover, since we dealt with kernels, the implicit representation $u(\mathbf{x}) = \sum_{t,r} \alpha_t^r K(\mathbf{x}_t, \mathbf{x})$ is actually used. Finally, no regularization has been performed, i.e. $\mu = 0$.

5.1 Experimental Setting and Results

The experimental setting is the same used in [3]. The dataset is synthetic. Points $\mathbf{x} = (x_1, x_2)$ are uniformly distributed in the unit square $[0, 1]^2$. The ranks are then assigned basing on the following rule:

$$r \in \{0, \ldots, 4\} : 10(x_1 - 0.5)(x_2 - 0.5) + \epsilon \in (b_r, b_{r+1})$$

where $b = \{b_0, \ldots, b_5\} = \{-\infty, -1, -0.1, 0.25, 1, +\infty\}$ and ϵ is a normally distributed noise $\epsilon \sim N(0, \sigma)$. We generated 100 sequences of 100,000 examples each. Moreover, a non-homogeneous second order polynomial kernel $K(\mathbf{x}_1, \mathbf{x}_2) = \phi(\mathbf{x}_1)\cdot\phi(\mathbf{x}_2) = (\mathbf{x}_1 \cdot \mathbf{x}_2 + 1)^2$ has been used. The performance on a sequence is obtained by feeding all the instances of the sequence and computing the *cumulative cost* at each iteration m as $c_m = \sum_{t=1}^m |\hat{r}_t - r_t|$. Finally, the obtained costs are averaged over the 100 sequences to obtain higher statistical significance.

Experiments have been performed using configurations produced according to three dimensions:

- *Cost Mapping*: Three cost mappings have been used. Two of them are the ones presented in Section 3.3, i.e. the mapping $\mathcal{G}_I$ with costs (denoted $\mathcal{G}_I^c$) and the mapping $\mathcal{G}_d$. The last mapping is basically the mapping $\mathcal{G}_I$ where the cost assignment is not performed. Note that, this mapping represents the cost function which gives a unitary cost for uncorrect predicted ranks.

- *Complexity of the task*: Different values of the standard deviation $\sigma \in \{0, 0.125, 0.5, 1.0\}$ have been used. A greater σ leads to a more difficult task.

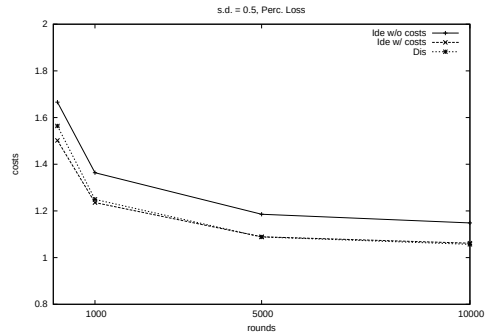
- *Preference Loss*: Two losses from the ones in Table 2 have been used, i.e. the Perceptron loss, and the sigmoidal loss with parameter $\lambda = 1$, $\theta = -1$.

One may notice that the configuration $(\mathcal{G}_d, \cdot, \text{PLoss})$ is equivalent to the PRank algorithm proposed in [3].

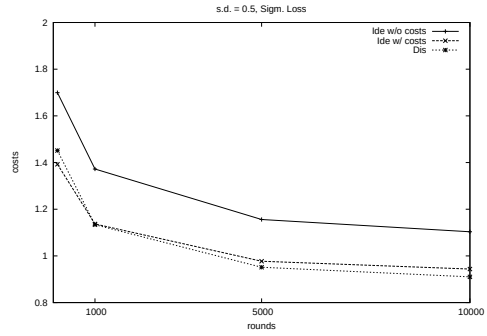
In Fig. 1, the curves of cost obtained for the three mappings and $\sigma = 0.5$ are shown. Different plots refer to the two preference losses. In Table 3 a detail of results after 10000 presentations is shown. Results show that the baseline cost mapping $\mathcal{G}_I$ is consistently worse than the other two, while the performance of $\mathcal{G}_I^c$ and $\mathcal{G}_d$ ar quite similar. Interestingly, a far larger improvement is obtained for the sigmoidal loss and this can be due to the better approximation of the true cost.

	— Perc. Loss —			**— Sigm. Loss —**		
σ	$\mathcal{G}_I$	$\mathcal{G}_I^c$	$\mathcal{G}_d$	$\mathcal{G}_I$	$\mathcal{G}_I^c$	$\mathcal{G}_d$
0.000	0.369	0.339	0.317	0.326	0.259	0.236
0.125	0.502	0.470	0.452	0.454	0.384	0.364
0.500	1.148	1.062	1.057	1.104	0.944	0.910
1.000	1.661	1.575	1.620	1.626	1.474	1.447

Table 3. Costs for different methods and task complexities.



(a)



(b)

Figure 1. Curves of the cost obtained for $\sigma = 0.5$ with different cost mappings. (a) Perceptron loss, (b) Sigmoidal loss.

6 Conclusion

We have proposed a general preference model for supervised learning and its application to on-line and batch algorithms. The model allows to codify cost functions as preferences and naturally plug them into the same training algorithm. In this view, the role of the cost functions here resembles the role of kernels in kernel-machines. Furthermore, the proposed method gives a tool for comparing different methods and cost functions on a same learning problem. Experiments performed on an ordinal regression problem have confirmed the validity of the approach and highlighted the important role of the loss functions used for training.

References

[1] F. Aiolli. *Large Margin Multiclass Learning: Models and Algorithms*. PhD thesis, Dept. of Computer Science, University of Pisa, 2004. http://www.di.unipi.it/~ aiolli/thesis.ps.

[2] F. Aiolli and A. Sperduti. Preference learning for multiclass problems. In *Advances in Neural Information Processing Systems*. MIT Press, 2004.

[3] K. Crammer and Y. Singer. Pranking with ranking. In *Advances in Neural Information Processing Systems*, 2001.

[4] K. Crammer and Y. Singer. A new family of online algorithms for category ranking. *Journal of Machine Learning Research*, 2003.

[5] O. Dekel, C.D. Manning, and Y. Singer. Log-linear models for label ranking. In *Advances in Neural Information Processing Systems*, 2003.

[6] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13:1–50, 2000.

[7] S. Har-Peled, D. Roth, and D. Zimak. Constraint classification for multiclass classification and ranking. In *Advances in Neural Information Processing Systems*, 2002.

[8] R. Herbrich, T. Graepel, P. Bollmann-Sdorra, and K. Ober-mayer. Learning a preference relation for information retrieval. In *Proceedings of the AAAI Workshop Text Categorization and Machine Learning*, 1998.

[9] R. Herbrich, T. Graepel, and K. Obermayer. Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*, pages 115–132,. MIT Press, 2000.

[10] P. McCullagh and J.A. Nelder. *Generalized Linear Models*. Chapman & Hall, 1983.

[11] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the Twenty-first international conference on Machine learning*, 2004.

[12] Hong Wu, Hanqing Lu, and Songde Ma. A practical svm-based algorithm for ordinal regression in image retrieval. In *Proceedings of the eleventh ACM international conference on Multimedia*, 2003.

[13] T. Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the International Conference on Machine learning*, 2004.