Online Structured Prediction via Coactive Learning

Pannaga Shivaswamy PANNAGA@CS.CORNELL.EDU
Thorsten Joachims TJ@CS.CORNELL.EDU
Department of Computer Science, Cornell University, Ithaca NY 14853

Abstract

We propose Coactive Learning as a model of interaction between a learning system and a human user, where both have the common goal of providing results of maximum utility to the user. At each step, the system (e.g. search engine) receives a context (e.g. query) and predicts an object (e.g. ranking). The user responds by correcting the system if necessary, providing a slightly improved – but not necessarily optimal – object as feedback. We argue that such feedback can often be inferred from observable user behavior, for example, from clicks in web-search. Evaluating predictions by their cardinal utility to the user, we propose efficient learning algorithms that have $\mathcal{O}(\frac{1}{\sqrt{T}})$ average regret, even though the learning algorithm never observes cardinal utility values as in conventional online learning. We demonstrate the applicability of our model and learning algorithms on a movie recommendation task, as well as ranking for web-search.

1. Introduction

In a wide range of systems in use today, the interaction between human and system takes the following form. The user issues a command (e.g. query) and receives a – possibly structured – result in response (e.g. ranking). The user then interacts with the results (e.g. clicks), thereby providing implicit feedback about the user's utility function. Here are three examples of such systems and their typical interaction patterns:

Web-search: In response to a query, a search engine presents the ranking $[A, B, C, D, ...]$ and observes that the user clicks on documents B and D.

Movie Recommendation: An online service recommends movie A to a user. However, the user rents movie B after browsing the collection.

Machine Translation: An online machine translator is used to translate a wiki page from language A to B. The system observes some corrections the user makes to the translated text.

In all the above examples, the user provides some feedback about the results of the system. However, the feedback is only an incremental improvement, not necessarily the optimal result. For example, from the clicks on the web-search results we can infer that the user would have preferred the ranking $[B, D, A, C, ...]$ over the one we presented. However, this is unlikely to be the best possible ranking. Similarly in the recommendation example, movie B was preferred over movie A, but there may have been even better movies that the user did not find while browsing. In summary, the algorithm typically receives a slightly improved result from the user as feedback, but not necessarily the optimal prediction nor any cardinal utilities. We conjecture that many other applications fall into this schema, ranging from news filtering to personal robotics.

Our key contributions in this paper are threefold. First, we formalize Coactive Learning as a model of interaction between a learning system and its user, define a suitable notion of regret, and validate the key modeling assumption – namely whether observable user behavior can provide valid feedback in our model – in a web-search user study. Second, we derive learning algorithms for the Coactive Learning Model, including the cases of linear utility models and convex cost functions, and show $\mathcal{O}(1/\sqrt{T})$ regret bounds in either case with a matching lower bound. The learning algorithms perform structured output prediction (see (Bakir et al., 2007)) and thus can be applied in a wide variety of problems. Several extensions of the model and the algorithm are discussed as well. Third, we provide extensive empirical evaluations of our algorithms on a movie recommendation and a web-search task, showing that the algorithms are highly efficient and effective in practical settings.

Appearing in *Proceedings of the 29th International Conference on Machine Learning*, Edinburgh, Scotland, UK, 2012.

2. Related Work

The Coactive Learning Model bridges the gap between two forms of feedback that have been well studied in online learning. On one side there is the multi-armed bandit model (Auer et al., 2002b;a), where an algorithm chooses an action and observes the utility of (only) that action. On the other side, utilities of all possible actions are revealed in the case of learning with expert advice (Cesa-Bianchi & Lugosi, 2006). Online convex optimization (Zinkevich, 2003) and online convex optimization in the bandit setting (Flaxman et al., 2005) are continuous relaxations of the expert and the bandit problems respectively. Our model, where information about two arms is revealed at each iteration sits between the expert and the bandit setting. Most closely related to Coactive Learning is the dueling bandits setting (Yue et al., 2009; Yue & Joachims, 2009). The key difference is that both arms are chosen by the algorithm in the dueling bandits setting, whereas one of the arms is chosen by the user in the Coactive Learning setting.

While feedback in Coactive Learning takes the form of a preference, it is different from ordinal regression and ranking. Ordinal regression (Crammer & Singer, 2001) assumes training examples (x, y), where y is a rank. In the Coactive Learning model, absolute ranks are never revealed. Closely related is learning with pairs of examples (Herbrich et al., 2000; Freund et al., 2003; Chu & Ghahramani, 2005) where absolute ranks are not needed; however, existing approaches require an *iid* assumption and typically perform batch learning. There is also a large body of work on ranking (see (Liu, 2009)). These approaches are different from Coactive Learning; they require training data (x, y) where y is the *optimal* ranking for query x.

3. Coactive Learning Model

We now introduce coactive learning as a model of interaction (in rounds) between a learning system (e.g. search engine) and a human (e.g. search user) where both the human and learning algorithm have the same goal (of obtaining good results). At each round t, the learning algorithm observes a context $\mathbf{x}_t \in \mathcal{X}$ (e.g. a search query) and presents a structured object $\mathbf{y}_t \in \mathcal{Y}$ (e.g. a ranked list of URLs). The utility of $\mathbf{y}_t \in \mathcal{Y}$ to the user for context $\mathbf{x}_t \in \mathcal{X}$ is described by a utility function $U(\mathbf{x}_t, \mathbf{y}_t)$, which is unknown to the learning algorithm. As feedback the user returns an improved object $\bar{\mathbf{y}}_t \in \mathcal{Y}$ (e.g. reordered list of URLs), i.e.,

$$U(\mathbf{x}_t, \bar{\mathbf{y}}_t) > U(\mathbf{x}_t, \mathbf{y}_t), \tag{1}$$

when such an object $\bar{\mathbf{y}}_t$ exists. In fact, we will also allow violations of (1) when we formally model user feedback in Section 3.1. The process by which the user generates the feedback $\bar{\mathbf{y}}_t$ can be understood as an approximate utility-maximizing search, but over a user-defined subset $\bar{\mathcal{Y}}_t$ of all possible $\mathcal{Y}$. This models an approximately and boundedly rational user that may employ various tools (e.g., query reformulations, browsing) to perform this search. Importantly, however, the feedback $\bar{\mathbf{y}}_t$ is typically not the optimal label

$$\mathbf{y}_t^* := \text{argmax}_{\mathbf{y} \in \mathcal{Y}} U(\mathbf{x}_t, \mathbf{y}). \tag{2}$$

In this way, Coactive Learning covers settings where the user cannot manually optimize the argmax over the full $\mathcal{Y}$ (e.g. produce the best possible ranking in web-search), or has difficulty expressing a bandit-style cardinal rating for $\mathbf{y}_t$ in a consistent manner. This puts our preference feedback $\bar{\mathbf{y}}_t$ in stark contrast to supervised learning approaches which require $(\mathbf{x}_t, \mathbf{y}_t^*)$. But even more importantly, our model implies that reliable preference feedback (1) can be derived from observable user behavior (i.e., clicks), as we will demonstrate in Section 3.2 for web-search. We conjecture that similar feedback strategies also exist for other applications, where users can be assumed to act approximately and boundedly rational according to U.

Despite the weak preference feedback, the aim of a coactive learning algorithm is to still present objects with utility close to that of the optimal $\mathbf{y}_t^*$. Whenever, the algorithm presents an object $\mathbf{y}_t$ under context $\mathbf{x}_t$, we say that it suffers a regret $U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t)$ at time step t. Formally, we consider the average regret suffered by an algorithm over T steps as follows:

$$REG_T = \frac{1}{T} \sum_{t=1}^{T} \left(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t) \right). \tag{3}$$

The goal of the learning algorithm is to minimize REG_T, thereby providing the human with predictions $\mathbf{y}_t$ of high utility. Note, however, that a cardinal value of U is never observed by the learning algorithm, but U is only revealed ordinally through preferences (1).

3.1. Quantifying Preference Feedback Quality

To provide any theoretical guarantees about the regret of a learning algorithm in the coactive setting, we need to quantify the quality of the user feedback. Note that this quantification is a tool for theoretical analysis, not a prerequisite or parameter to the algorithm. We quantify feedback quality by how much improvement $\bar{\mathbf{y}}$ provides in utility space. In the simplest case, we say that user feedback is *strictly α-informative* when

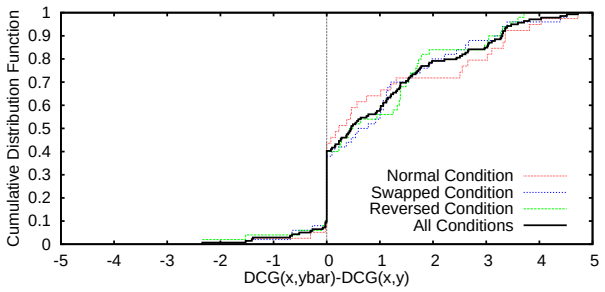

Figure 1. Cumulative distribution of utility differences between presented ranking $\mathbf{y}$ and click-feedback ranking $\bar{\mathbf{y}}$ in terms of DCG@10 for three experimental conditions and overall.

the following inequality is satisfied:

$$U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t) \geq \alpha(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t)). \quad (4)$$

In the above inequality, $\alpha \in (0, 1]$ is an unknown parameter. Feedback is such that utility of $\bar{\mathbf{y}}_t$ is higher than that of $\mathbf{y}_t$ by a fraction α of the maximum possible utility range $U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t)$. Violations of the above feedback model are allowed by introducing slack variables $\xi_t \geq 0$:[1]

$$U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t) \geq \alpha(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t)) - \xi_t. \quad (5)$$

We refer to the above feedback model as α-*informative* feedback. Note also that it is possible to express feedback of any quality using (5) with an appropriate value of ξ_t. Our regret bounds will contain ξ_t, quantifying to what extent the strict α-informative modeling assumption is violated.

Finally, we will also consider an even weaker feedback model where a positive utility gain is only achieved in expectation over user actions:

$$\mathbf{E}_t[U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t)] \geq \alpha(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t)) - \bar{\xi}_t. \quad (6)$$

We refer to the above feedback as *expected* α-*informative* feedback. In the above equation, the expectation is over the user's choice of $\bar{\mathbf{y}}_t$ given $\mathbf{y}_t$ under context $\mathbf{x}_t$ (i.e., under a distribution $\mathbf{P}_{\mathbf{x}_t}[\bar{\mathbf{y}}_t|\mathbf{y}_t]$ which is dependent on $\mathbf{x}_t$).

3.2. User Study: Preferences from Clicks

We now validate that reliable preferences as specified in Equation (1) can indeed be inferred from implicit user behavior. In particular, we focus on preference feedback from clicks in web-search and draw upon data from a user study (Joachims et al., 2007). In this study, subjects (undergraduate students, $n = 16$) were asked to answer 10 questions – 5 informational, 5 navigational – using the Google search engine. All queries, result lists, and clicks were recorded. For each subject, queries were grouped into query chains by question[2]. On average, each query chain contained 2.2 queries and 1.8 clicks in the result lists.

We use the following strategy to infer a ranking $\bar{\mathbf{y}}$ from the user's clicks: prepend to the ranking $\mathbf{y}$ from the first query of the chain all results that the user clicked throughout the whole query chain. To assess whether $U(\mathbf{x}, \bar{\mathbf{y}})$ is indeed larger than $U(\mathbf{x}, \mathbf{y})$ as assumed in our learning model, we measure utility in terms of a standard measure of retrieval quality from Information Retrieval. We use $DCG@10(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{10} \frac{r(\mathbf{x}, \mathbf{y}[i])}{\log i+1}$, where $r(\mathbf{x}, \mathbf{y}[i])$ is the relevance score of the i-th document in ranking $\mathbf{y}$ (see e.g. (Manning et al., 2008)). To get ground-truth relevance assessments $r(\mathbf{x}, d)$, five human assessors were asked to manually rank the set of results encountered during each query chain. We then linearly normalize the resulting ranks to a relative relevance score $r(\mathbf{x}, d) \in [0..5]$ for each document.

We can now evaluate whether the feedback ranking $\bar{\mathbf{y}}$ is indeed better than the ranking $\mathbf{y}$ that was originally presented, i.e. $DCG@10(\mathbf{x}, \bar{\mathbf{y}}) > DCG@10(\mathbf{x}, \mathbf{y})$. Figure 1 plots the Cumulative Distribution functions (CDFs) of $DCG@10(\mathbf{x}, \bar{\mathbf{y}}) - DCG@10(\mathbf{x}, \mathbf{y})$ for three experimental conditions, as well as the average over all conditions. All CDFs are shifted far to the right of 0, showing that preference feedback from our strategy is highly accurate and informative. Focusing first on the average over all conditions, the utility difference is strictly positive on $\sim 60\%$ of all queries, and strictly negative on only $\sim 10\%$. This imbalance is significant (binomial sign test, $p < 0.0001$). Among the remaining $\sim 30\%$ of cases where the DCG@10 difference is zero, 88% are due to $\bar{\mathbf{y}} = \mathbf{y}$ (i.e. click only on top 1 or no click). Note that a learning algorithm can easily detect those cases and may explicitly eliminate them as feedback. Overall, this shows that implicit feedback can indeed produce accurate preferences.

What remains to be shown is whether the reliability of the feedback is affected by the quality of the current prediction, i.e., $U(\mathbf{x}_t, \mathbf{y}_t)$. In the user study, some users actually received results for which retrieval quality was degraded on purpose. In particular, about one third of the subjects received Google's top 10 results in reverse order (condition "reversed") and another third received rankings with the top two positions swapped (condition "swapped"). As Figure 1 shows, we find that users provide accurate preferences across this sub-

[1] Strictly speaking, the value of the slack variable depends on the choice of α and the definition of utility. However, for brevity, we do not explicitly show this dependence.

[2] This was done manually, but can be automated with high accuracy (Jones & Klinkner, 2008).

Algorithm 1 Preference Perceptron.

Initialize $\mathbf{w}_1 \leftarrow \mathbf{0}$
for $t = 1$ **to** T **do**
  Observe $\mathbf{x}_t$
  Present $\mathbf{y}_t \leftarrow \text{argmax}_{\mathbf{y}\in\mathcal{Y}} \mathbf{w}_t^\top \phi(\mathbf{x}_t, \mathbf{y})$
  Obtain feedback $\bar{\mathbf{y}}_t$
  Update: $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)$
end for

stantial range of retrieval quality. Intuitively, a worse retrieval system may make it harder to find good results, but it also makes an easier baseline to improve upon. This intuition is formally captured in our definition of α-informative feedback. The optimal value of the α vs. ξ trade-off, however, will likely depend on many application-specific factors, like user motivation, corpus properties, and query difficulty. In the following, we therefore present algorithms that do not require knowledge of α, theoretical bounds that hold for any value of α, and experiments that explore a large range of α.

4. Coactive Learning Algorithms

In this section, we present algorithms for minimizing regret in the coactive learning model. In the rest of this paper, we use a linear model for the utility function,

$$U(\mathbf{x}, \mathbf{y}) = \mathbf{w}_*^\top \phi(\mathbf{x}, \mathbf{y}), \tag{7}$$

where $\mathbf{w}_* \in \mathbf{R}^N$ is an unknown parameter vector and $\phi : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbf{R}^N$ is a joint feature map such that $\|\phi(\mathbf{x}, \mathbf{y})\|_{\ell_2} \leq R$ for any $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{Y}$. Note that both $\mathbf{x}$ and $\mathbf{y}$ can be structured objects.

We start by presenting and analyzing the most basic algorithm for the coactive learning model, which we call the *Preference Perceptron* (Algorithm 1). The Preference Perceptron maintains a weight vector $\mathbf{w}_t$ which is initialized to $\mathbf{0}$. At each time step t, the algorithm observes the context $\mathbf{x}_t$ and presents an object $\mathbf{y}$ that maximizes $\mathbf{w}_t^\top \phi(\mathbf{x}_t, \mathbf{y})$. The algorithm then observes user feedback $\bar{\mathbf{y}}_t$ and the weight vector $\mathbf{w}_t$ is updated in the direction $\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)$.

Theorem 1 *The average regret of the preference perceptron algorithm can be upper bounded, for any* $\alpha \in (0, 1]$ *and for any* $\mathbf{w}_*$ *as follows:*

$$REG_T \leq \frac{1}{\alpha T} \sum_{t=1}^{T} \xi_t + \frac{2R\|\mathbf{w}_*\|}{\alpha\sqrt{T}}. \tag{8}$$

Proof First, consider $\|\mathbf{w}_{T+1}\|^2$, we have,

$$\begin{aligned}
&\mathbf{w}_{T+1}^\top \mathbf{w}_{T+1} = \mathbf{w}_T^\top \mathbf{w}_T + 2\mathbf{w}_T^\top (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T)) \\
&+ (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))^\top (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T) \\
&\leq \mathbf{w}_T^\top \mathbf{w}_T + 4R^2 \leq 4R^2 T.
\end{aligned}$$

On line one, we simply used our update rule from algorithm 1. On line two, we used the fact that $\mathbf{w}_T^\top (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T)) \leq 0$ from the choice of $\mathbf{y}_T$ in Algorithm 1 and that $\|\phi(\mathbf{x}, \mathbf{y})\| \leq R$. Further, from the update rule in algorithm 1, we have,

$$\begin{aligned}
\mathbf{w}_{T+1}^\top \mathbf{w}_* &= \mathbf{w}_T^\top \mathbf{w}_* + (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))^\top \mathbf{w}_* \\
&= \sum_{t=1}^{T} \left(U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t) \right).
\end{aligned} \tag{9}$$

We now use the fact that $\mathbf{w}_{T+1}^\top \mathbf{w}_* \leq \|\mathbf{w}_*\| \|\mathbf{w}_{T+1}\|$ (Cauchy-Schwarz inequality), which implies

$$\sum_{t=1}^{T} \left(U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t) \right) \leq 2R\sqrt{T}\|\mathbf{w}_*\|.$$

From the α-informative modeling of the user feedback in (5), we have

$$\alpha \sum_{t=1}^{T} \left(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t) \right) - \sum_{t=1}^{T} \xi_t \leq 2R\sqrt{T}\|\mathbf{w}_*\|,$$

from which the claimed result follows. ■

The first term in the regret bound denotes the quality of feedback in terms of violation of the strict α-informative feedback. In particular, if the user feedback is strictly α-informative, then all slack variables in (8) vanish and $REG_T = \mathcal{O}(1/\sqrt{T})$.

Though user feedback is modeled via α-informative feedback, the algorithm itself does not require the knowledge of α; α plays a role only in the analysis.

Although the preference perceptron appears similar to the standard perceptron for multi-class classification problems, there are key differences. First, the standard perceptron algorithm requires the *true label* $\mathbf{y}^*$ as feedback, whereas much weaker feedback $\bar{\mathbf{y}}$ suffices for our algorithm. Second, the standard analysis of the perceptron bounds the number of mistakes made by the algorithm based on margin and the radius of the examples. In contrast, our analysis bounds a different regret that captures a graded notion of utility.

An appealing aspect of our learning model is that several interesting extensions are possible. We discuss some of them in the rest of this section.

4.1. Lower Bound

We now show that the upper bound in Theorem 1 cannot be improved in general.

Lemma 2 *For any coactive learning algorithm $\mathcal{A}$ with linear utility, there exist $\mathbf{x}_t$, objects $\mathcal{Y}$ and $\mathbf{w}_*$ such that REG_T of $\mathcal{A}$ in T steps is $\Omega(1/\sqrt{T})$.*

Proof Consider a problem where $\mathcal{Y} = \{-1, +1\}, \mathcal{X} = \{\mathbf{x} \in \mathbf{R}^T : \|\mathbf{x}\| = 1\}$. Define the joint feature map as $\phi(\mathbf{x}, \mathbf{y}) = \mathbf{y}\mathbf{x}$. Consider T contexts $\mathbf{e}_1, \ldots, \mathbf{e}_T$ such that $\mathbf{e}_j$ has only the j^{th} component equal to one and all the others equal to zero. Let $\mathbf{y}_1, \ldots \mathbf{y}_T$ be the sequence of outputs of $\mathcal{A}$ on contexts $\mathbf{e}_1, \ldots, \mathbf{e}_T$. Construct $\mathbf{w}_* = [-\mathbf{y}_1/\sqrt{T} \ -\mathbf{y}_2/\sqrt{T} \cdots - \mathbf{y}_T/\sqrt{T}]^\top$, we have for this construction $\|\mathbf{w}_*\| = 1$. Let the user feedback on the t^{th} step be $-\mathbf{y}_t$. With these choices, the user feedback is always α-informative with $\alpha = 1$ since $\mathbf{y}_t^* = -\mathbf{y}_t$. Yet, the regret of the algorithm is $\frac{1}{T}\sum_{t=1}^{T}(\mathbf{w}_*^\top \phi(\mathbf{e}_t, \mathbf{y}_t^*) - \mathbf{w}_*^\top \phi(\mathbf{e}_t, \mathbf{y}_t)) = \Omega(\frac{1}{\sqrt{T}})$. ■

4.2. Batch Update

In some applications, due to high volumes of feedback, it might not be possible to do an update after every round. For such scenarios, it is natural to consider a variant of Algorithm 1 that makes an update every k iterations; the algorithm simply uses $\mathbf{w}_t$ obtained from the previous update until the next update. It is easy to show the following regret bound for batch updates:

$$REG_T \leq \frac{1}{\alpha T}\sum_{t=1}^{T}\xi_t + \frac{2R\|\mathbf{w}_*\|\sqrt{k}}{\alpha\sqrt{T}}.$$

4.3. Expected α-Informative Feedback

So far, we have characterized user behavior in terms of deterministic feedback actions. However, if a bound on the expected regret suffices, the weaker model of Expected α-Informative Feedback from Equation (6) is applicable.

Corollary 3 *Under expected α-informative feedback model, the expected regret (over user behavior distribution) of the preference perceptron algorithm can be upper bounded as follows:*

$$\mathbf{E}[REG_T] \leq \frac{1}{\alpha T}\sum_{t=1}^{T}\bar{\xi}_t + \frac{2R\|\mathbf{w}_*\|}{\alpha\sqrt{T}}. \qquad (10)$$

The above corollary can be proved by following the argument of Theorem 1, but taking expectations over user feedback: $\mathbf{E}[\mathbf{w}_{T+1}^\top \mathbf{w}_{T+1}] = \mathbf{E}[\mathbf{w}_T^\top \mathbf{w}_T] + \mathbf{E}[2\mathbf{w}_T^\top(\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))] + \mathbf{E}_T[(\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))^\top(\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))] \leq \mathbf{E}[\mathbf{w}_T^\top \mathbf{w}_T] + 4R^2$. In the above, $\mathbf{E}$ denotes expectation over all user feedback $\bar{\mathbf{y}}_t$ given $\mathbf{y}_t$ under the context $\mathbf{x}_t$. It follows that $\mathbf{E}[\mathbf{w}_{T+1}^\top \mathbf{w}_{T+1}] \leq 4TR^2$.

Applying Jensen's inequality on the concave function $\sqrt{\cdot}$, we get: $\mathbf{E}[\mathbf{w}_T^\top \mathbf{w}_*] \leq \|\mathbf{w}_*\|\mathbf{E}[\|\mathbf{w}_T\|] \leq \|\mathbf{w}_*\|\sqrt{\mathbf{E}[\mathbf{w}_T^\top \mathbf{w}_T]}$. The corollary follows from the definition of expected α-informative feedback.

Algorithm 2 Convex Preference Perceptron.

```
Initialize w_1 ← 0
for t = 1 to T do
  Set η_t ← 1/√t
  Observe x_t
  Present y_t ← argmax_{y∈Y} w_t^⊤ φ(x_t, y)
  Obtain feedback ȳ_t
  Update: w̄_{t+1} ← w_t + η_t G(φ(x_t, ȳ_t) − φ(x_t, y_t))
  Project: w_{t+1} ← arg min_{u∈B} ‖u − w̄_{t+1}‖^2
end for
```

4.4. Convex Loss Minimization

We now generalize our results to minimize convex losses defined on the linear utility differences. We assume that at every time step t, there is an (unknown) convex loss function $c_t : \mathbf{R} \to \mathbf{R}$ which determines the loss $c_t(U(\mathbf{x}_t, \mathbf{y}_t) - U(\mathbf{x}_t, \mathbf{y}_t^*))$ at time t. The functions c_t are assumed to be non-increasing. Further, subderivatives of the c_t's are assumed to be bounded (i.e., $c_t'(\theta) \in [-G, 0]$ for all t and for all $\theta \in \mathbf{R}$). The vector $\mathbf{w}_*$ which determines the utility of $\mathbf{y}_t$ under context $\mathbf{x}_t$ is assumed from a closed and bounded convex set $\mathcal{B}$ whose diameter is denoted as $|\mathcal{B}|$.

Algorithm 2 minimizes the average convex loss. There are two differences between this algorithm and Algorithm 1. Firstly, there is a rate η_t associated with the update at time t. Moreover, after every update, the resulting vector $\bar{\mathbf{w}}_{t+1}$ is projected back to the set $\mathcal{B}$. We have the following result for Algorithm 2, a proof of which is provided in an extended version of this paper (Shivaswamy & Joachims, 2012).

Theorem 4 *For the convex preference perceptron, we have, for any $\alpha \in (0, 1]$ and any $\mathbf{w}_* \in \mathcal{B}$,*

$$\frac{1}{T}\sum_{t=1}^{T} c_t(U(\mathbf{x}_t, \mathbf{y}_t) - U(\mathbf{x}_t, \mathbf{y}_t^*)) \leq \frac{1}{T}\sum_{t=1}^{T} c_t(0) + \frac{2G}{\alpha T}\sum_{t=1}^{T}\xi_t + \frac{1}{\alpha}\left(\frac{|\mathcal{B}|G}{2\sqrt{T}} + \frac{|\mathcal{B}|G}{T} + \frac{4R^2G}{\sqrt{T}}\right). \qquad (11)$$

In the bound (11), $c_t(0)$ is the minimum possible convex loss since $U(\mathbf{x}_t, \mathbf{y}_t) - U(\mathbf{x}_t, \mathbf{y}_t^*)$ can never be greater than zero by definition of $\mathbf{y}_t^*$. Thus the theorem upper bounds the average convex loss via the minimum achievable loss and the quality of feedback. Like the previous result (Theorem 1), under strict α-informative feedback, the average loss approaches the best achievable loss at $\mathcal{O}(1/\sqrt{T})$ albeit with larger constant factors.

5. Experiments

We empirically evaluated the Preference Perceptron algorithm on two datasets. The two experiments differed in the nature of prediction and feedback. While the algorithm operated on structured objects (rankings) in one experiment, atomic items (movies) were presented and received as feedback in the other.

5.1. Structured Feedback: Learning to Rank

We evaluated our Preference Perceptron algorithm on the Yahoo! learning to rank dataset (Chapelle & Chang, 2011). This dataset consists of query-url feature vectors (denoted as $\mathbf{x}_i^q$ for query q and URL i), each with a relevance rating r_i^q that ranges from zero (irrelevant) to four (perfectly relevant). To pose ranking as a structured prediction problem, we defined our joint feature map as follows:

$$\mathbf{w}^\top \phi(q, \mathbf{y}) = \sum_{i=1}^{5} \frac{\mathbf{w}^\top \mathbf{x}_{\mathbf{y}_i}^q}{\log(i+1)}. \tag{12}$$

In the above equation, $\mathbf{y}$ denotes a ranking such that $\mathbf{y}_i$ is the index of the URL which is placed at position i in the ranking. Thus, the above measure considers the top five URLs for a query q and computes a score based on a graded relevance. Note that the above utility function defined via the feature-map is analogous to DCG@5 (see e.g. (Manning et al., 2008)) after replacing the relevance label with a linear prediction based on the features.

For query q_t at time step t, the Preference Perceptron algorithm presents the ranking $\mathbf{y}_t^q$ that maximizes $\mathbf{w}_t^\top \phi(q_t, \mathbf{y})$. Note that this merely amounts to sorting documents by the scores $\mathbf{w}_t^\top \mathbf{x}_i^{q_t}$, which can be done very efficiently. The utility regret in Eqn. (3), based on the definition of utility in (12), is given by $\frac{1}{T}\sum_{t=1}^{T} \mathbf{w}_*^\top(\phi(q_t, \mathbf{y}^{q_t *}) - \phi(q_t, \mathbf{y}^{q_t}))$. Here $\mathbf{y}^{q_t *}$ denotes the optimal ranking with respect to $\mathbf{w}_*$, which is the best least squares fit to the relevance labels from the features using the entire dataset. Query ordering was randomly permuted twenty times and we report average and standard error of the results.

5.1.1. Strong Vs Weak Feedback

The goal of the first experiment was to see how the regret of the algorithm changes with feedback quality. To get feedback at different quality levels α, we used the following mechanism. Given the predicted ranking $\mathbf{y}_t$, the user would go down the list until she found five URLs such that, when placed at the top of the list, the resulting $\bar{\mathbf{y}}_t$ satisfied the strictly α-informative feedback condition w.r.t. the optimal $\mathbf{w}_*$.

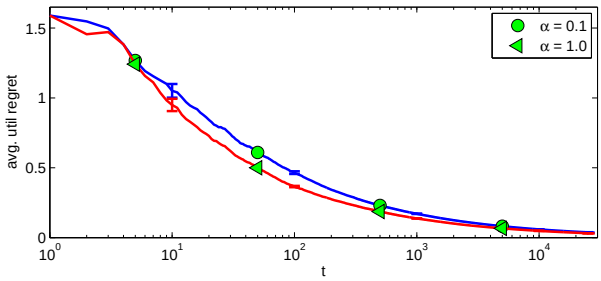

Figure 2. Regret based on strictly α-informative feedback.

Figure 2 shows the results for this experiment for two different α values. As expected, the regret with $\alpha = 1.0$ is lower compared to the regret with respect $\alpha = 0.1$. Note, however, that the difference between the two curves is much smaller than a factor of ten. This is because strictly α-informative feedback is also strictly β-informative feedback for any $\beta \leq \alpha$. So, there could be several instances where user feedback was much stronger than what was required. As expected from the theoretical bounds, since the user feedback is based on a linear model with no noise, utility regret approaches zero.

5.1.2. Noisy Feedback

In the previous experiment, user feedback was based on actual utility values computed from the optimal $\mathbf{w}_*$. We next make use of the actual relevance labels provided in the dataset for user feedback. Now, given a ranking for a query, the user would go down the list inspecting the top 10 URLs (or all the URLs if the list is shorter) as before. Five URLs with the highest relevance labels (r_i^q) are placed at the top five locations in the user feedback. Note that this produces noisy feedback since no linear model can perfectly fit the relevance labels on this dataset.

As a baseline, we repeatedly trained a conventional Ranking SVM[3]. At each iteration, the previous SVM model was used to present a ranking to the user. The user returned a ranking based on the relevance labels as above. The pairs of examples $(q_t, \mathbf{y}_{svm}^{q_t})$ and $(q_t, \bar{\mathbf{y}}_{svm}^{q_t})$ were used as training pairs for the ranking

[3] `http://svmlight.joachims.org`

SVMs. Note that training a ranking SVM after each iteration would be prohibitive, since it involves solving a quadratic program and cross-validating the regularization parameter C. Thus, we retrained the SVM whenever 10% more examples were added to the training set. The first training was after the first iteration with just one pair of examples (starting with a random $\mathbf{y}^{q_1}$), and the C value was fixed at 100 until there were 50 pairs of examples, when reliable cross-validation became possible. After there were more than 50 pairs in the training set, the C value was obtained via five-fold cross-validation. Once the C value was determined, the SVM was trained on all the training examples available at that time. The same SVM model was then used to present rankings until the next retraining.

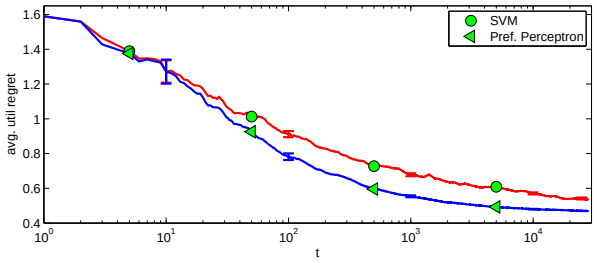

Figure 3. Regret vs time based on noisy feedback.

Results of this experiment are presented in Figure 3. Since the feedback is now based on noisy relevance labels, the utility regret converges to a non-zero value as predicted by our theoretical results. Over most of the range, the Preference Perceptron performs significantly[4] better than the SVM. Moreover, the perceptron experiment took around 30 minutes to run, whereas the SVM experiment took about 20 hours on the same machine. We conjecture that the regret values for both the algorithms can be improved with better features or kernels, but these extensions are orthogonal to the main focus of this paper.

5.2. Item Feedback: Movie Recommendation

In contrast to the structured prediction problem in the previous section, we now evaluate the Preference Perceptron on a task with atomic predictions, namely movie recommendation. In each iteration a movie is presented to the user, and the feedback consists of a movie as well. We use the MovieLens dataset, which consists of a million ratings over 3090 movies rated by 6040 users. The movie ratings ranged from one to five.

We randomly divided users into two equally sized sets. The first set was used to obtain a feature vector $\mathbf{m}_j$ for each movie j using the "SVD embedding" method for collaborative filtering (see (Bell & Koren, 2007), Eqn. (15)). The dimensionality of the feature vectors and the regularization parameters were chosen to optimize cross-validation accuracy on the first dataset in terms of squared error. For the second set of users, we then considered the problem of recommending movies based on the movie features $\mathbf{m}_j$. This experiment setup simulates the task of recommending movies to a new user based on movie features from old users.

For each user i in the second set, we found the best least squares approximation $\mathbf{w}_{i*}^T\mathbf{m}_j$ to the user's utility functions on the available ratings. This enables us to impute utility values for movies that were not explicitly rated by this user. Furthermore, it allows us to measure regret for each user as $\frac{1}{T}\sum_{t=1}^{T}\mathbf{w}_{i*}^\top(\mathbf{m}_{t*}-\mathbf{m}_t)$, which is the average difference in utility between the recommended movie $\mathbf{m}_t$ and the best available movie $\mathbf{m}_{t*}$. We denote the best available movie at time t by $\mathbf{m}_{t*}$, since in this experiment, once a user gave a particular movie as feedback, both the recommended movie and the feedback movie were removed from the set of candidates for subsequent recommendations.

5.2.1. Strong Vs Weak Feedback

Analogous to the web-search experiments, we first explore how the performance of the Preference Perceptron changes with feedback quality α. In particular, we recommended a movie with maximum utility according to the current $\mathbf{w}_t$ of the algorithm, and the user returns as feedback a movie with the smallest utility that still satisfied strictly α-informative feedback according to $\mathbf{w}_{i*}$. For every user in the second set, the algorithm iteratively recommended 1500 movies in this way. Regret was calculated after each iteration and separately for each user, and all regrets were averaged over all the users in the second set.

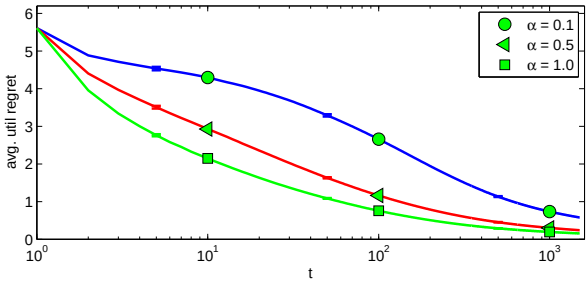

Figure 4. Regret for strictly α-informative feedback.

Figure 4 shows the results for this experiment. Since the feedback in this case is strictly α-informative, the average regret in all the cases decreases towards zero as expected. Note that even for a moderate value of α, regret is already substantially reduced after 10's of iterations. With higher α values, the regret converges to zero at a much faster rate than with lower α values.

[4] The error bars are extremely tiny at higher iterations.

5.2.2. Noisy Feedback

We now consider noisy feedback, where the user feedback does not necessarily match the linear utility model used by the algorithm. In particular, feedback is now given based on the actual ratings when available, or the score $\mathbf{u}_{i*}^\top \mathbf{m}_j$ rounded to the nearest allowed rating value. In every iteration, the user returned a movie with one rating higher than the one presented to her. If the algorithm already presented a movie with the highest rating, it was assumed that the user gave the same movie as feedback.

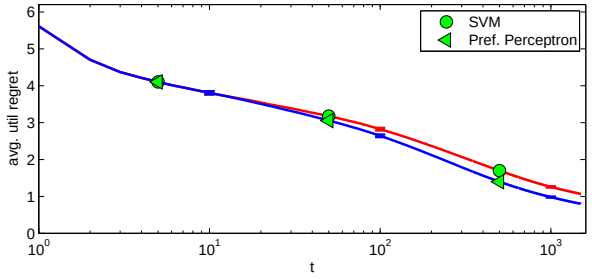

Figure 5. Regret based on noisy feedback.

As a baseline, we again ran a ranking SVM. Like in the web-search experiment, it was retrained whenever 10% more training data was added. The results for this experiment are shown in Figure 5. The regret of the Preference Perceptron is again significantly lower than that of the SVM, and at a small fraction of the computational cost.

6. Conclusions

We proposed a new model of online learning where preference feedback is observed but cardinal feedback is never observed. We proposed a suitable notion of regret and showed that it can be minimized under our feedback model. Further, we provided several extensions of the model and algorithms. Furthermore, experiments demonstrated its effectiveness for web-search ranking and a movie recommendation task. A future direction is to consider λ-strongly convex functions, and we conjecture it is possible to derive algorithms with $\mathcal{O}(\log(T)/T)$ regret in this case.

Acknowledgements We thank Peter Frazier, Bobby Kleinberg, Karthik Raman and Yisong Yue for helpful discussions. This work was funded in part under NSF awards IIS-0905467 and IIS-1142251.

References

Auer, P., Cesa-Bianchi, N., and Fischer, P. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002a.

Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. The non-stochastic multi-armed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002b.

Bakir, G.H., Hofmann, T., Schölkopf, B., Smola, A.J., Taskar, B., and Vishwanathan, S.V.N. (eds.). *Predicting Structured Data*. The MIT Press, 2007.

Bell, R. M. and Koren, Y. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *ICDM*, 2007.

Cesa-Bianchi, N. and Lugosi, G. *Prediction, learning, and games*. Cambridge University Press, 2006.

Chapelle, O. and Chang, Y. Yahoo! learning to rank challenge overview. *JMLR - Proceedings Track*, 14:1–24, 2011.

Chu, W. and Ghahramani, Z. Preference learning with gaussian processes. In *ICML*, 2005.

Crammer, K. and Singer, Y. Pranking with ranking. In *NIPS*, 2001.

Flaxman, A., Kalai, A. T., and McMahan, H. B. Online convex optimization in the bandit setting: gradient descent without a gradient. In *SODA*, 2005.

Freund, Y., Iyer, R. D., Schapire, R. E., and Singer, Y. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.

Herbrich, R., Graepel, T., and Obermayer, K. Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*. MIT Press, 2000.

Joachims, T., Granka, L., Pan, Bing, Hembrooke, H., Radlinski, F., and Gay, G. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Transactions on Information Systems (TOIS)*, 25(2), April 2007.

Jones, R. and Klinkner, K. Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. In *CIKM*, 2008.

Liu, T-Y. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3, March 2009.

Manning, C., Raghavan, P., and Schütze, H. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

Shivaswamy, P. and Joachims, T. Online structured prediction via coactive learning. *arXiv:1205.4213*, 2012.

Yue, Y. and Joachims, T. Interactively optimizing information retrieval systems as a dueling bandits problem. In *ICML*, 2009.

Yue, Y., Broder, J., Kleinberg, R., and Joachims, T. The k-armed dueling bandits problem. In *COLT*, 2009.

Zinkevich, M. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, 2003.