Journal of Artificial Intelligence Research 53 (2015) 1-40 Submitted 08/14; published 05/15

Coactive Learning

Pannaga Shivaswamy PSHIVASWAMY@LINKEDIN.COM
LinkedIn Corporation
2029 Stierlin Ct
Mountain View, CA 94043, USA

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

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. Interactions in the Coactive Learning model take the following form: 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 preference feedback can be inferred in large quantity from observable user behavior (e.g., clicks in web search), unlike the optimal feedback required in the expert model or the cardinal valuations required for bandit learning. Despite the relaxed requirements for the feedback, we show that it is possible to adapt many existing online learning algorithms to the coactive framework. In particular, we provide algorithms that achieve $\mathcal{O}(1/\sqrt{T})$ average regret in terms of cardinal utility, even though the learning algorithm never observes cardinal utility values directly. We also provide an algorithm with $\mathcal{O}(\log(T)/T)$ average regret in the case of λ-strongly convex loss functions. An extensive empirical study demonstrates the applicability of our model and 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. And in the machine translation example, the corrected text need not be the best possible translation from language A to language B. In all three examples, the algorithm typically receives *a slightly improved result from the user as feedback, but not necessarily the optimal prediction or any cardinal utilities.* We conjecture that many other applications fall into this schema, ranging from news filtering to personal robotics.

In this paper, we propose *Coactive Learning* as a model of such system-user interactions. We formalize Coactive Learning as a general 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 user study for web search. The new model can be viewed as a cooperative learning process between system and user, where both parties aim to optimize utility but lack the means to achieve this goal on their own. Specifically, the (boundedly rational) user is computationally limited in maximizing utility over the space of alternatives, while the system is limited in how well it knows the user's utility function.

The proposed online learning framework differs significantly from existing online learning models in terms of the observed feedback (see the related works section for a comparison). A strength of the proposed framework is that it is possible to derive a wide range of coactive learning algorithms by adapting existing online algorithms for convex optimization. We provide a template for Coactive Learning algorithms and then show several instances of this template in this paper, and in each case, we prove that the worst case analysis of the algorithm carries over from the conventional online learning framework to coactive learning despite the differences between the two models. In particular, in the cases of linear utility models and convex cost functions we show $\mathcal{O}(1/\sqrt{T})$ regret bounds with a matching lower bound. We also show that the regret bound can be improved with a second order algorithm for strongly convex functions. The learning algorithms perform structured output prediction (see Bakir, Hofmann, Schölkopf, Smola, Taskar, & Vishwanathan, 2007) and thus can be applied in a wide variety of problems. We study several interesting extensions of the framework using batch updates, expected feedback, and an exponentiated learning algorithm. Finally, 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.

The rest of this paper is organized as follows. We discuss related work in Section 2. In Section 3 we formally introduce the coactive learning model and also motivate the model with a real-world user study. We present the linear version of our algorithm along with several extensions in Section 4. In Section 5, we then detail a general schema for deriving coactive learning algorithms and their regret bounds. In particular, we derive an exponentiated gradient algorithm in Section 5.1, and we propose coactive learning algorithms for minimizing general convex losses and λ-strongly convex losses in Sections 5.2 and 5.3. An empirical evaluation of the proposed framework and algorithms is done in Section 6 and we conclude in Section 7. We include most of our proofs in the Appendix.

2. Related Works

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 (e.g., Auer, Cesa-Bianchi, Freund, & Schapire, 2002b; Auer, Cesa-Bianchi, & Fischer, 2002a), 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 (e.g., Cesa-Bianchi & Lugosi, 2006a). Online convex optimization (Zinkevich, 2003; Hazan, Agarwal, & Kale, 2007) and online convex optimization in the bandit setting (Flaxman, Kalai, & McMahan, 2005) are continuous relaxations of the expert and the bandit problems respectively. Our model, where information about two arms is revealed at each iteration (the one we presented and the one we receive as feedback from the user), sits between the expert and the bandit setting. Most closely related to Coactive Learning is the dueling bandits setting (Yue, Broder, Kleinberg, & Joachims, 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. Our model allows contextual information like in contextual bandits (Langford & Zhang, 2007), however, the arms in our problem are structured objects such as rankings. A summary of how our framework compares with other existing frameworks is shown in Table 1. Other types of feedback have also been explored in the literature. For example, in the multi-class classification problems, after the algorithm makes a prediction based on the context, the feedback received is only whether the prediction is correct or wrong as opposed to the actual label (Crammer & Gentile, 2011; Kakade, Shalev-Shwartz, & Tewari, 2008). This can be seen as observing partial feedback (as opposed to the actual cardinal feedback) in a bandit problem.

As pointed out above, Coactive Learning algorithms and conventional online learning algorithms operate in different types of environments. Coactive Learning algorithms present an object and observe another object as a feedback, while online convex learning algorithms pick a vector in each step and observe the gradient at that vector as feedback. Despite the contrast between online learning and Coactive Learning, two of the algorithms presented in this paper are closely related to those in the work of Zinkevich (2003) and Hazan et al. (2007). We show that it is possible to adapt the regret bounds of these algorithms to corresponding regrets bounds for Coactive Learning. At the heart of all our algorithms and analysis is the well-known idea (Polyak & Tsypkin, 1973) that the descent algorithms do not necessarily need to know the gradients, but that a vector with positive inner product with the gradient in expectation suffices.

While feedback in Coactive Learning takes the form of a preference, it is different from ordinal regression and ranking. Ordinal regression (e.g., Crammer & Singer, 2001) assumes training examples (x, y), where y is a rank. In the Coactive Learning model, absolute ranks are never revealed. More closely related is learning with pairs of examples (Herbrich, Graepel, & Obermayer, 2000; Freund, Iyer, Schapire, & Singer, 2003; Chu & Ghahramani, 2005), since it circumvents the need for absolute ranks; only relative orderings are required. Vari-

Framework	Algorithm	Feedback
Bandits	pull an arm	observe cardinal reward for the arm pulled
Experts	pull an arm	observe cardinal rewards for all the arms
Dueling Bandits	pull two arms	observe feedback on which one is better
Coactive Learning	pull an arm	observe another arm which is better

Table 1: A comparison of different online learning frameworks.

ants of such pairwise ranking algorithms have been applied to Natural Language Processing (Haddow, Arun, & Koehn, 2011; Zhang, Lei, Barzilay, Jaakkola, & Globerson, 2014) and image annotation (Weston, Bengio, & Usunier, 2011). However, existing approaches require an *iid* assumption and typically perform batch learning. Finally, there is a large body of work on ranking (see Liu, 2009). These approaches are different from Coactive Learning as they require training data (x, y) where y is the *optimal* ranking for query x. However, we will draw upon structured prediction approaches for ranking problems in the design of our models.

Coactive learning was first proposed by Shivaswamy and Joachims (2012); this paper serves as a journal extension of that paper, adding a complete discussion of batch updates and expected feedback, the exponentiated gradient algorithm, the $\mathcal{O}(\log(T)/T)$ algorithm for λ-strongly convex loss functions, and a substantially extended empirical evaluation. Since then, coactive learning has been applied to intrinsically diverse retrieval (Raman, Shivaswamy, & Joachims, 2012), learning ranking function from click feedback (Raman, Joachims, Shivaswamy, & Schnabel, 2013), optimizing social welfare (Raman & Joachims, 2013), personal robotics (Jain, Wojcik, Joachims, & Saxena, 2013), pattern discovery (Boley, Mampaey, Kang, Tokmakov, & Wrobel, 2013), robotic monitoring (Somers & Hollinger, 2014), and extended to allow approximate inference (Goetschalckx, Fern, & Tadepalli, 2014).

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 (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 human user returns an improved object $\bar{\mathbf{y}}_t \in \mathcal{Y}$ (e.g. reordered list of URLs), i.e.,[1]

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

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 approximately utility-maximizing action, where the user is modeled as a boundedly rational

1. The improvements should be not just strict, but by a "margin", this will be clear in Section 3.1.

agent. In particular, the user selects the feedback object $\bar{\mathbf{y}}_t$ by approximately maximizing utility over a user-defined subset $\bar{\mathcal{Y}}_t$ of all possible $\mathcal{Y}$.

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

This approximately and boundedly rational user may employ various tools (e.g., query reformulations, browsing) to construct the subset $\bar{Y}$ and to perform this search. Importantly, however, the feedback $\bar{\mathbf{y}}_t$ is typically *not* the optimal label which is defined as,

$$\mathbf{y}_t^* := \text{argmax}_{\mathbf{y}\in\mathcal{Y}} U(\mathbf{x}_t, \mathbf{y}). \tag{3}$$

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 $U(\mathbf{x}_t, \mathbf{y}_t)$ 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 exist in many application settings (e.g., Jain et al., 2013; Boley et al., 2013; Somers & Hollinger, 2014; Goetschalckx et al., 2014), especially when users can be assumed to act approximately and boundedly rational according to U.

Despite the weak preference feedback, the aim of the algorithm is nevertheless to 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 the algorithm over T steps as follows:

$$REG_T = \frac{1}{T}\sum_{t=1}^{T}\left(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t)\right). \tag{4}$$

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 the following inequality is satisfied:

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

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 $U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t)$. The term on the right hand side in the above inequality ensures that user feedback

$\bar{\mathbf{y}}_t$ is not only better than $\mathbf{y}_t$, but also better by a margin $\alpha(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t))$. Violations of the above feedback model are allowed by introducing slack variables ξ_t:[2]

$$U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t) = \alpha(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t)) - \xi_t. \tag{6}$$

Note that the ξ_t are not restricted to be positive, but can be negative as well. We refer to the above feedback model as α-*informative* feedback. Note also that it is possible to express feedback of any quality using (6) with an appropriate value of ξ_t. Our regret bounds will contain ξ_t, quantifying to what extent the 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:

$$\mathbf{E}_t[U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t)] = \alpha(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t)) - \bar{\xi}_t. \tag{7}$$

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 $\mathbf{x}_t$).

In the rest of this paper, we use a linear model for the utility function,

$$U(\mathbf{x}, \mathbf{y}) = \mathbf{w}_*^\top \phi(\mathbf{x}, \mathbf{y}), \tag{8}$$

where $\mathbf{w}_* \in \mathbf{R}^N$ is a parameter vector unknown both to the learning system and users and $\phi : \mathcal{X} \times \mathcal{Y} \to \mathbf{R}^N$ is a known joint feature map known to the system such that[3]

$$\|\phi(\mathbf{x}, \mathbf{y})\|_{\ell_2} \leq R, \tag{9}$$

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.

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, Granka, Pan, Hembrooke, Radlinski, & Gay, 2007). In this study, subjects (undergraduate students, $n = 16$) were asked to answer 10 identical 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[4]. 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 Manning, Raghavan, & Schütze,

2. 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 in the notation.
3. We make a slightly different assumption in Section 5.1.
4. This was done manually, but can be automated with high accuracy (Jones & Klinkner, 2008).

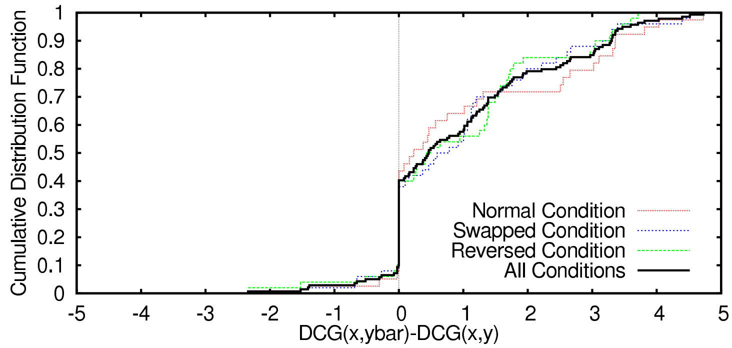

Figure 1: Cumulative distribution of utility differences between presented ranking $\mathbf{y}$ and click-feedback ranking $\bar{\mathbf{y}}$ in terms of $DCG@10$ for three experimental conditions and overall.

2008). To get ground-truth relevance assessments $r(\mathbf{x}, d)$, five independent human assessors (students) were asked to manually rank the set of results encountered during each query chain. The assessors were also given the true answers for navigational queries. 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 substantial range of retrieval quality. Intuitively, a worse retrieval system may make it harder to find good results, but it also makes an easier baseline $\mathbf{y}_t$ 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.

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

4. The Preference Perceptron for Coactive Learning

In this section, 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 $\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)$.

Although the update of 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. Further, the standard perceptron mistake bound (Novikoff, 1962) contains $R^2\|\mathbf{w}\|^2$ while our bound in the following Theorem contains $R\|\mathbf{w}\|$ where R is as defined in (9).

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:*

$$REG_T \le \frac{1}{\alpha T}\sum_{t=1}^{T} \xi_t + \frac{2R\|\mathbf{w}_*\|}{\alpha\sqrt{T}}. \tag{10}$$

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

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

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

$$\begin{aligned}
\mathbf{w}_{T+1}^\top \mathbf{w}_* &= \mathbf{w}_T^\top \mathbf{w}_* + (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))^\top \mathbf{w}_* \\
&= \sum_{t=1}^{T} (U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t)).
\end{aligned} \tag{11}$$

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

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

From the α-informative modeling of the user feedback in (6), we have

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

from which the claimed result follows. ■

The first term in the regret bound denotes the quality of feedback in terms of violation of the α-informative feedback. In particular, if the user feedback is strictly α-informative for some α, then all slack variables in (10) vanish and $REG_T = \mathcal{O}(1/\sqrt{T})$.

It is trivial to design algorithms (with even better regret) under strict α-informative assumption when the cardinality of the context set $\mathcal{X}$ is finite. One of the interesting aspects of the above bound (Theorem 1) and the subsequent results is that we can minimize the regret even when the context $\mathbf{x}_t$ is different in every step. Thus, $|\mathcal{X}|$ could be infinite and the regret bound still holds.

We note that the bound in Theorem 1 holds for any $\mathbf{w}_*$ and $\alpha \in (0, 1]$. The slacks have to be based on the corresponding α and $\mathbf{w}_*$.

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

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 (7) is applicable.

Corollary 2 *Under the expected α-informative feedback model, the expected regret (over user behavior distribution) of the preference perceptron algorithm can be upper bounded as follows:*

$$\mathbf{E}[REG_T] \leq \frac{1}{\alpha T} \sum_{t=1}^{T} \bar{\xi}_t + \frac{2R\|\mathbf{w}_*\|}{\alpha\sqrt{T}}. \tag{12}$$

The above corollary can be proved by following the argument of Theorem 1, but taking expectations over user feedback:

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

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 $\mathbf{E}[\mathbf{w}_{T+1}^\top \mathbf{w}_{T+1}] \leq 4TR^2$.

Algorithm 2 Batch Preference Perceptron.

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

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.

4.1 Lower Bound

We now show that the upper bound in Theorem 1 cannot be improved in general with respect to its scaling with T. In the following lemma, given any Coactive Learning algorithm, we construct a sequence of examples where, even with $\alpha = 1$ feedback, the algorithm suffers an average regret of $\Omega(1/\sqrt{T})$.

Lemma 3 *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 $\mathbf{w}_* = [-\mathbf{y}_1/\sqrt{T},\ -\mathbf{y}_2/\sqrt{T},\ \ldots\ , -\mathbf{y}_T/\sqrt{T}]^\top$, 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

The Preference Perceptron as stated in Algorithm 1 makes an update after every iteration. In some applications, due to high volumes of feedback, it might not be possible to do an update that frequently. 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

Algorithm 3 Generic Template for Coactive Learning Algorithms

Initialize $\mathbf{w}_1$
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 $\bar{\mathbf{y}}_t$
  Perform an update on $\mathbf{w}_t$ using the gradient of $\mathbf{w}^\top(\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t))$ to obtain $\mathbf{w}_{t+1}$.
end for

previous update until the next update. The type of updates shown in Algorithm 2 are called mini-batch updates and have been used in distributed online optimization (Dekel, Gilad-Bachrach, Shamir, & Xiao, 2012). The steps of the batch update algorithm are shown in Algorithm 2. It is easy to show the following regret bound for batch updates:

Lemma 4 *The average regret of the batch preference perceptron algorithm can be upper bounded, for any $\alpha \in (0, 1]$ and for any $\mathbf{w}_*$ as follows:*

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

While this lemma implies that mini-batches slow down learning by a factor equal to the batch size, we will see in Section 6.2.3 that empirically convergence is substantially faster.

5. Deriving Algorithms for Coactive Learning

The Preference Perceptron and the regret it minimizes, as defined in Eqn. (4), is only one point in the design space of different regret definitions and learning algorithms for coactive learning. In this section, we will outline a general strategy for deriving coactive learning algorithms from existing algorithms for online optimization. Furthermore, we will demonstrate that more general notions of regret are meaningful and feasible in coactive learning, and derive coactive learning algorithms for general convex and λ-strongly convex losses.

All coactive learning algorithms that we derive in this section follow the general template outlined in Algorithm 3. After initializing $\mathbf{w}_1$, in each iteration the context $\mathbf{x}_t$ is observed and the algorithm presents $\mathbf{y}_t$ by maximizing its current utility estimate represented by $\mathbf{w}_t$. Once the feedback $\bar{\mathbf{y}}_t$ is observed, the algorithm simply takes the gradient of $\mathbf{w}^\top(\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t))$ and uses an update from a standard online convex optimization algorithm to obtain $\mathbf{w}_{t+1}$ from $\mathbf{w}_t$.

In each case, an upper bound on the regret of the proposed algorithm is derived by using the following strategy. First, we start with a notion of regret that is suited for coactive learning. We then upper bound this regret by first reducing it to a form such that results from a standard online convex opimization regret bound can be applied. This gives a regret bound for the original coactive algorithm in turn. In each case, we use this template algorithm to derive a specific algorithm. However, we still provide a self-contained proof (in the appendix) clearly pointing out where we have used the regret bound from a corresponding online convex optimization algorithm.

Algorithm 4 Exponentiated Preference Perceptron

Intialize $\mathbf{w}_1^i \leftarrow \frac{1}{N}$
$\eta \leftarrow \frac{1}{2S\sqrt{T}}$
for $t = 1$ **to** T **do**
  Observe $\mathbf{x}_t$
  Present $\mathbf{y}_t \leftarrow \mathrm{argmax}_{\mathbf{y}\in\mathcal{Y}} \mathbf{w}_t^\top \phi(\mathbf{x}_t, \mathbf{y})$
  Obtain feedback $\bar{\mathbf{y}}_t$
  $\mathbf{w}_{t+1}^i \leftarrow \mathbf{w}_t^i \exp(-\eta(\phi^i(\mathbf{x}_t, \mathbf{y}_t) - \phi^i(\mathbf{x}_t, \bar{\mathbf{y}}_t)))/Z_t$, where Z_t is such that the weights add to one.
end for

5.1 Exponentiated Preference Perceptron

To illustrate the generic strategy for deriving coactive learning algorithms, we first derive an exponentiated gradient algorithm for coactive learning that can be used as an alternative to the Preference Perceptron. The exponentiated gradient algorithm inherits the ability to learn quickly for sparse weight vectors.

Unlike the additive updates of the Preference Perceptron, the exponentiated gradient algorithm summarized in Algorithm 4 performs multiplicative updates. This exponentiated algorithm is closely related to the exponentiated algorithms for classification (Kivinen & Warmuth, 1997). At the start, it initializes all weights uniformly. Each subsequent update step has a rate η associated with it. The rate depends on an upper bound on the ℓ_∞ norm of the features (i.e., $\|\phi(\cdot,\cdot)\|_{\ell_\infty} \leq S$) and the time horizon T. After each multiplicative update, the weights are normalized to sum to one, and the steps of the algorithm repeat. Since the updates are multiplicative and the weights are initially positive, $\mathbf{w}_t$ is guaranteed to remain in the positive orthant for this algorithm. We note that Algoithm 4 is assumed to know both T and S. There are standard techniques (see Cesa-Biachi & Lugosi, 2006b) to convert such an algorithm to not have dependence on T, however, such extensions are not the focus of this paper.

We now provide a regret bound for Algorithm 4. While the regret bounds for Algorithm 1 and Algorithm 2 depended on the ℓ_2 norm of the features, the bound for the exponentiated algorithm depends on the ℓ_∞ norm of the features.

Theorem 5 *For any $\mathbf{w}_* \in \mathbf{R}^N$ such that $\|\mathbf{w}_*\|_{\ell_1} = 1$, $\mathbf{w}_* \geq 0$, under (expected) α-informative feedback the average (expected) regret of the Exponentiated Preference Perceptron can be upper bounded as:*

$$REG_T \leq \frac{1}{\alpha T}\sum_{t=1}^{T}\xi_t + \frac{2\log(m)S}{\alpha\sqrt{T}} + \frac{S}{2\alpha\sqrt{T}},$$

$$\mathbf{E}[REG_T] \leq \frac{1}{\alpha T}\sum_{t=1}^{T}\bar{\xi}_t + \frac{2\log(m)S}{\alpha\sqrt{T}} + \frac{S}{2\alpha\sqrt{T}},$$

where $\|\phi(\mathbf{x}, \mathbf{y})\|_{\ell_\infty} \leq S$.

Proof We start with the regret of the coactive learning algorithm as defined in (4):

$$\begin{aligned}
REG_T &= \frac{1}{T}\sum_{t=1}^{T}(U(\mathbf{x}_t,\mathbf{y}_t^*)-U(\mathbf{x}_t,\mathbf{y}_t)) \\
&= \frac{1}{\alpha T}\sum_{t=1}^{T}(U(\mathbf{x}_t,\bar{\mathbf{y}}_t)-U(\mathbf{x}_t,\mathbf{y}_t))+\frac{1}{\alpha T}\sum_{t=1}^{T}\xi_t \\
&= \frac{1}{\alpha T}\sum_{t=1}^{T}\left(\mathbf{w}_*^\top\phi(\mathbf{x}_t,\bar{\mathbf{y}}_t)-\mathbf{w}_*^\top\phi(\mathbf{x}_t,\mathbf{y}_t)\right)+\frac{1}{\alpha T}\sum_{t=1}^{T}\xi_t \qquad (13)
\end{aligned}$$

In the above equation, we have used the definition of α-informative feedback as defined in Eqn. (6). By viewing Algorithm 4 as an exponentiated online gradient descent algorithm, it is easy to derive the following regret bound using techniques initially introduced by Kivinen and Warmuth (1997),

$$\sum_{t=1}^{T}(\mathbf{w}_t^\top(\phi(\mathbf{x}_t,\mathbf{y}_t)-\phi(\mathbf{x}_t,\bar{\mathbf{y}}_t)))\leq\sum_{t=1}^{T}(U(\mathbf{x}_t,\mathbf{y}_t)-U(\mathbf{x}_t,\bar{\mathbf{y}}_t))+2\log(N)S\sqrt{T}+\frac{S\sqrt{T}}{2}.$$

Since we could not find this specific bound in the literature, a self-contained proof is provided in Appendix A. In the proof, REG_T is first upper bounded in terms of the difference between $KL(\mathbf{w}||\mathbf{w}_{t+1})$ and $KL(\mathbf{w}||\mathbf{w}_t)$. A telescoping argument is then used to get the above result.

Observing that $\mathbf{w}_t^\top(\phi(\mathbf{x}_t,\mathbf{y}_t)-\phi(\mathbf{x}_t,\bar{\mathbf{y}}_t))\geq 0$, we get,

$$\sum_{t=1}^{T}(U(\mathbf{x}_t,\bar{\mathbf{y}}_t)-U(\mathbf{x}_t,\mathbf{y}_t))\leq 2\log(N)S\sqrt{T}+\frac{S\sqrt{T}}{2}. \qquad (14)$$

Combining (13) and (14), we obtain the average regret bound. The proof of the expected regret bound is analogous to that of the Preference Perceptron. ■

Like the result in Theorem 1, the above result (Theorem 5) also bounds the regret in terms of the noise in the feedback (first term) and additional terms which converge to zero at the rate $\mathcal{O}(1/\sqrt{T})$. The key difference to Theorem 1 is that the regret bound of the exponentiated algorithm scales logarithmically with the number of features, but with the ℓ_1-norm of $\mathbf{w}$, which can be advantageous if the optimal $\mathbf{w}$ is sparse.

5.2 Convex Preference Perceptron

Generalizing the definition of regret from Eqn. (4), we now allow 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 based on the difference in utility between $\mathbf{y}_t$ and the optimal $\mathbf{y}_t^*$. The functions c_t are assumed to be non-increasing. Non-increasing assumption on c_t is based on the intuition that the loss should be higher as $U(\mathbf{x}_t,\mathbf{y}_t)$ is farther from $U(\mathbf{x}_t,\mathbf{y}_t^*)$. Further, sub-derivatives of the c_t's are assumed to be bounded. Formally, $c_t'(\theta)\in[-G,0]$ for all t and for all $\theta\in\mathbf{R}$ where $c_t'(\theta)$ denotes the sub-derivative of $c_t(\cdot)$ at θ. The vector $\mathbf{w}_*$ which determines the utility of $\mathbf{y}_t$ under context $\mathbf{x}_t$ is assumed from a closed and

Algorithm 5 Convex Preference Perceptron.

Initialize $\mathbf{w}_1 \leftarrow \mathbf{0}$
for $t = 1$ **to** T **do**
  Set $\eta_t \leftarrow \frac{1}{\sqrt{t}}$
  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: $\bar{\mathbf{w}}_{t+1} \leftarrow \mathbf{w}_t + \eta_t(\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t))$
  Project: $\mathbf{w}_{t+1} \leftarrow \arg\min_{\mathbf{u}\in\mathcal{B}} \|\mathbf{u} - \bar{\mathbf{w}}_{t+1}\|^2$
end for

bounded convex set $\mathcal{B}$ whose diameter is denoted as $|\mathcal{B}|$. In the case of convex losses, we consider the following notion of regret:

$$CREG_T := \frac{1}{T}\sum_{t=1}^{T} c_t(U(\mathbf{x}_t, \mathbf{y}_t) - U(\mathbf{x}_t, \mathbf{y}_t^*)) - \frac{1}{T}\sum_{t=1}^{T} c_t\,(0) \tag{15}$$

In the bound (16), $c_t(0)$ 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 $\mathbf{y}_t^*$. Hence the above regret compares the loss of our algorithm with the best loss that could be achieved with a convex loss. Note that, for the case $c_t(\theta) = -\theta$, the above definition of regret reduces to our earlier definition of regret in the linear case (Eqn. (4)).

Algorithm 5 minimizes the average convex loss. There are two differences between this algorithm and Algorithm 1. First, there is a rate η_t associated with the update at time t. Second, after every update, the resulting vector $\bar{\mathbf{w}}_{t+1}$ is projected back to the set $\mathcal{B}$. Algorithm 5 is also closely related to the online convex optimization algorithm propsed by Zinkevich (2003). However, the online convex optimization algorithm assumes that the gradient of the loss ($c_t(\cdot)$) is observed after each iteration. Our algorithm doesn't observe the gradient directly, but only observes an improved object $\bar{\mathbf{y}}_t$ after presenting an object $\mathbf{y}_t$.

Our earlier regret bounds were expressed in terms of slack variables ξ_t. However, here and in the following section, our bounds will be expressed in terms of the clipped version of the slack variables defined as $\xi_t^+ := \max(0, \xi_t)$.

Theorem 6 *For the Convex Preference Perceptron under α-informative feedback, for non-increasing convex losses $c_t(\cdot)$ with bounded sub-derivative, we have, for any $\alpha \in (0, 1]$ and any $\mathbf{w}_* \in \mathcal{B}$,*

$$CREG_T \leq \frac{2G}{\alpha T}\sum_{t=1}^{T}\xi_t^+ + \frac{G}{\alpha}\left(\frac{|\mathcal{B}|}{2\sqrt{T}} + \frac{|\mathcal{B}|}{T} + \frac{4R^2}{\sqrt{T}}\right). \tag{16}$$

Similarly, under expected α-informative feedback, we have,

$$\mathbf{E}[CREG_T] \leq \frac{2G}{\alpha T}\sum_{t=1}^{T}\bar{\xi}_t^+ + \frac{G}{\alpha}\left(\frac{|\mathcal{B}|}{2\sqrt{T}} + \frac{|\mathcal{B}|}{T} + \frac{4R^2}{\sqrt{T}}\right). \tag{17}$$

The proof for the above Theorem is provided in the Appendix B. The idea of the proof is to first divide the time steps into two types depending on the nature of the feedback. This allows us to upper bound $CREG_T$ in terms of $\sum_{t=1}^{T}(\mathbf{w}_t - \mathbf{w}_*)^\top(\phi(\mathbf{x}_t, \mathbf{y}_t) - \phi(\mathbf{x}_t, \bar{\mathbf{y}}_t))$. This term can further be upper bounded by following the argument from Zinkevich (2003) even in the Coactive Learning framework.

From the definition of $CREG_T$ in Eqn. (15), the above theorem upper bounds the average convex loss via the minimum achievable loss and the quality of the 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.

In the case of the linear utility bounds in Theorem 1 and Theorem 5, it was sufficient to have the average of the slack variables be zero to achieve zero regret. However, in the case of convex losses, our upper bound on regret approaches zero only when the average of the clipped slack variables is zero.

5.3 Second-Order Preference Perceptron

For a particular class of convex functions, it turns out that we can give much stronger regret bounds than for general convex losses. The improvement for this special class of losses parallels improvements in online convex optimization from general convex losses (Zinkevich, 2003) to λ-strongly convex losses (Hazan et al., 2007).

Definition 7 *A convex function $f : \mathcal{D} \to R$ is λ-strongly convex if for all points x and y in $\mathcal{D}$, the following condition is satisfied for a fixed $\lambda > 0$:*

$$f(x) \leq f(y) + \nabla f(x)^\top (x - y) - \frac{\lambda}{2}||y - x||^2, \tag{18}$$

where $\nabla f(x)$ denotes a sub-derivative at x.

Algorithm 6 shows the Second-order Preference Perceptron for λ-strongly convex losses. Like our previous algorithms, the Second-order Preference Perceptron also maintains a weight vector $\mathbf{w}_t$, and the step of presenting $\mathbf{y}_t$ based on a context $\mathbf{x}_t$ is still the same as in our previous algorithms. However, in addition to the weight vector, it also maintains an additional matrix A_t which is constructed from the outer product of the vector $\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)$. The update step and the projection steps now involve both A_t as shown in Algorithm 6. Algorithm 6 is closely related to the online convex optimization algorithm proposed by Hazan et al. (2007). However, as we pointed out in the case of Algorithm 5, our algorithm only observes a user preference feedback after each step unlike online convex optimization algorithms which observe gradients. It is still possible to prove a regret bound for the λ-strongly convex case, and we have the following result.

Theorem 8 *For the second order preference learning algorithm, for (expected) λ-strongly convex, non-increasing functions c_t, with bounded sub-derivatives, we have,*

$$CREG_T \leq \frac{\gamma}{2T\alpha^2}\sum_{t=1}^{T}\xi_t^{+2} + \frac{2G}{T\alpha}\sum_{t=1}^{T}\xi_t^{+} + \frac{G\epsilon|\mathcal{B}|}{T\alpha} + \frac{GN}{2T\gamma\alpha}\log\left(\frac{4R^2T\gamma}{\epsilon} + 1\right), \tag{19}$$

$$\mathbf{E}[CREG_T] \leq \frac{\gamma}{2T\alpha^2}\sum_{t=1}^{T}\bar{\xi}_t^{+2} + \frac{2G}{T\alpha}\sum_{t=1}^{T}\bar{\xi}_t^{+} + \frac{G\epsilon|\mathcal{B}|}{T\alpha} + \frac{GN}{2T\gamma\alpha}\log\left(\frac{4R^2T\gamma}{\epsilon} + 1\right), \tag{20}$$

Algorithm 6 Second-order Preference Perceptron.

Intialize $\mathbf{w}_1 \leftarrow \mathbf{0}$
$A_0 \leftarrow \epsilon \mathbf{I}$
$\gamma \leftarrow \lambda / G$.
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 $\bar{\mathbf{y}}_t$
  $A_t \leftarrow A_{t-1} + \gamma[\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)][\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)]^\top$
  Update: $\bar{\mathbf{w}}_{t+1} \leftarrow \mathbf{w}_t + A_t^{-1}[\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)]$
  Project: $\mathbf{w}_{t+1} = \arg\min_{\mathbf{w} \in \mathcal{B}} (\bar{\mathbf{w}}_{t+1} - \mathbf{w})^\top A_t (\bar{\mathbf{w}}_{t+1} - \mathbf{w})$
end for

where, $\epsilon > 0$ is an initialization parameter, as shown in Algorithm 6.

We prove the above Theorem in the Appendix C. Like in the proof of Theorem 6, we divide time steps into two types. Starting with this, it is possible to upper bound $CREG_T$ to such a form that the resulting terms can be upper bounded using similar arguments as that for online strongly convex optimization (Hazan et al., 2007).

When user feedback is strictly α-informative for some α and some $\mathbf{w}^* \in \mathcal{B}$, the first two terms of the regret bound (19) result in an $\mathcal{O}(\frac{\log T}{T})$ scaling with T. However, there is a linear dependence on the dimensionality of the joint feature map in the regret bound for the Second-order Preference Perceptron algorithm.

Even though it appears like we need to invert the matrix A_t in the Second-order Preference Perceptron, this can be avoided since the updates on A_t are of rank one. By the Woodbury matrix inversion lemma, we have:

$$\begin{aligned} A_t^{-1} &= (A_{t-1} + \gamma[\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)][\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)]^\top)^\top)^{-1} \\ &= A_{t-1}^{-1} - \frac{A_{t-1}^{-1}[\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)][\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)]^\top A_{t-1}^{-1}}{1/(\gamma) + [\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)]^\top A_{t-1}^{-1}[\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)]}. \end{aligned}$$

Thus, in practice, the Second-order Preference Perceptron can update both A_t and B_t in each iteration. Nevertheless, the projection step to obtain $\mathbf{w}_{t+1}$ involves solving a quadratically-constrained quadratic program where $\mathcal{B}$ is a ball of fixed radius, which still takes $\mathcal{O}(N^3)$ time. Hence, the Second-order Preference Perceptron is computationally more demanding than the Convex Preference Perceptron. As we show in the experiments, the Second-order Preference Perceptron might be still quite useful for low-noise data.

6. Experiments

We empirically evaluate our Coactive Learning algorithms on two real-world datasets. The two datasets differ in the nature of prediction and feedback. On the first dataset, the algorithms operate on structured objects (rankings) whereas on the second dataset, atomic items (movies) were presented and received as feedback.

6.1 Datasets And User Feedback Models

First, we provide a detailed description of the two datasets that were used in our experiments. Along with this, we provide the details of the strategies that we used on each dataset for generating user feedback.

6.1.1 Structured Feedback: Web-Search

Our first dataset is a publicly available dataset from Yahoo! (Chapelle & Chang, 2011) for learning to rank in web-search. 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:

$$\mathbf{w}^\top \phi(q, \mathbf{y}) = \sum_{i=1}^{5} \frac{\mathbf{w}^\top \mathbf{x}_{\mathbf{y}_i}^q}{\log(i+1)}. \tag{21}$$

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 graded relevance. Note that the above utility function defined via the feature-map is analogous to DCG@5 (see, 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 Coactive Learning algorithms present 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. (4), based on the definition of utility $\mathbf{w}_*^\top \phi(q, \mathbf{y})$ 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 we consider to be the best least squares fit to the relevance labels from the features using the entire dataset. We obtain $\mathbf{y}^{q_t*}$ from Eqn. 3, that is, $\mathbf{y}^{q_t*} = \mathrm{argmax}_{\mathbf{y}\in\bar{\mathcal{Y}}} \mathbf{w}_*^\top \phi(q_t, \mathbf{y})$. In all our experiments, query ordering was randomly permuted twenty times and we report average and standard error of the results.

We used the following two user models for generating simulated user feedback in our experiments. The first feedback model is an idealized version of feedback whereas the second feedback is based directly on relevance labels that are available in the dataset:

- **Strict α-informative feedback:** In this model, the user is assumed to provide strictly α-informative feedback at a given α value (i.e., slacks zero). 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 $\mathbf{w}_*$. This model assumes that the user has access to $\mathbf{w}_*$ hence it is an idealized feedback.

- **Noisy feedback at depth k:** In this feedback model, given a ranking for a query, the user would go down the list inspecting the top k URLs (or all the URLs if the list is shorter) for a specified k value. 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.

6.1.2 Item Feedback: Movie Recommendation

In contrast to the structured prediction problem in the previous dataset, we considered a second dataset with atomic predictions, namely movie recommendation. In each iteration, a movie is presented to the user, and the feedback consists of a single movie as well. We used the MovieLens dataset from grouplense.org which consists of a million ratings over 3900 movies as rated by 6040 users. The movie ratings range from one to five.

We randomly divided users into two equally sized sets. The first set was used to obtain a feature vector $\mathbf{x}_j$ 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 $\mathbf{x}_j$. 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 $\mathbf{w}_{i*}^T\mathbf{x}_j$ to the user's utility functions on the available ratings. This enabled us to impute utility values for movies that were not explicitly rated by this user. Furthermore, it allowed us to measure regret for each user as $\frac{1}{T}\sum_{t=1}^{T}\mathbf{w}_{i*}^{\top}(\mathbf{x}_{t*}-\mathbf{x}_t)$, which is the average difference in utility between the recommended movie $\mathbf{x}_t$ and the best available movie $\mathbf{x}_{t*}$. We denote the best available movie at time t by $\mathbf{x}_{t*}$ which is obtained from Eqn. 3. 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. In all the experiments we report (average) regret values averaged over all 3020 users in the test set.

To simulate user behavior, we considered the following two feedback models on this dataset:

- **Strict α-informative feedback:** As in the previous dataset, in this model, the user is assumed to provide strictly α-informative feedback at a given α value (i.e., slacks zero). Given the predicted movie $\mathbf{y}_t$, the user is assumed to watch the movie if it already has the highest rating in the remaining corpus of movies. If not, the user picks another movie from the corpus with lowest utility that still satisfies strict α-informative assumption. This model again assumes that the user has access to $\mathbf{w}_*$, hence it is an idealized feedback.

- **Noisy feedback:** In this feedback model, given a movie $\mathbf{y}$, the user is assumed to have access to either the actual rating of the movies (when available) or is assumed to round the imputed rating to the nearest legal rating value. We used two sub-strategies by which the user provides feedback. In **better** feedback, the user provides $\bar{\mathbf{y}}$ such that it has the smallest rating (actual rating or rounded rating) but strictly better rating than that of $\mathbf{y}$. In **best** feedback, the user provides $\bar{\mathbf{y}}$ such that it has the highest rating (actual rating or rounded rating) in the remaining corpus. There could be multiple movies satisfying the above criteria, and ties were broken uniformly at random among such movies. Note that this feedback model results in a noisy feedback due to rounding of movie ratings to discrete values.

6.2 Preference Perceptron

In the first set of experiments, we analyze the empirical performance and scaling behavior of the basic Preference Perceptron Algorithm and its variants.

6.2.1 Strong Versus Weak Feedback

The goal of the first experiment to explore how the regret of the algorithm changed with feedback quality. To get feedback at different quality levels α, we used **strict α-informative feedback** for various α values.

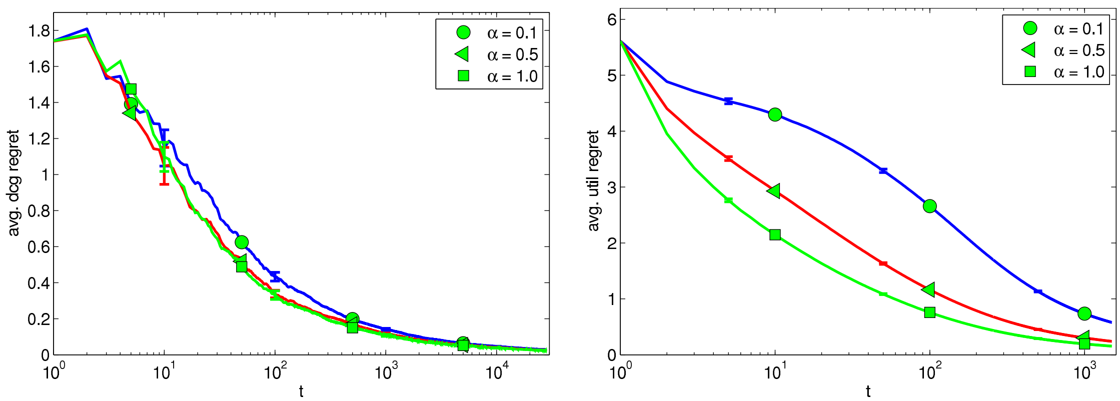

Figure 2: Regret based on **strict α-informative feedback** for various α values for web-search (left) and movie recommendation (right).

Figure 2 shows the results of this experiment for three different α values. Overall, regret is typically substantially reduced after only tens or hundreds of iterations. As expected, the regret for $\alpha = 1.0$ is lower compared to the regret for lower α values. Note, however, that the difference between the two curves is much smaller than a factor of ten. Also note that the differences are less prominent in the case of web-search. This is because strictly α-informative feedback is also strictly β-informative feedback for any $\beta \leq \alpha$. So, in our user feedback model, we could be providing much stronger feedback than that required by a particular α value. As expected from the theoretical bounds, since the user feedback is based on a linear model with no noise, utility regret approaches zero in all the cases. Note that we show standard error in the plots, giving an indication of statistical significance. In the left plots in Figure 2, the standard errors are high at lower iterations but become lower with more iterations. In some plots in the rest of the paper, the error bars are small and may be difficult to visually identify.

In the rest of this paper, for **strict α-informative feedback**, we consider $\alpha = 0.5$ unless we explicitly mention otherwise.

6.2.2 Noisy Feedback

In the previous experiment, user feedback was based on actual utility values computed from the optimal $\mathbf{w}_*$. We next study how regret changes with noisy feedback where user behavior does not follow a linear utility model. For the web-search dataset, we use **noisy feedback** at depths 10 and 25, and for the movie dataset we use **noisy feedback** with both the **better** and the **best** variant of it.

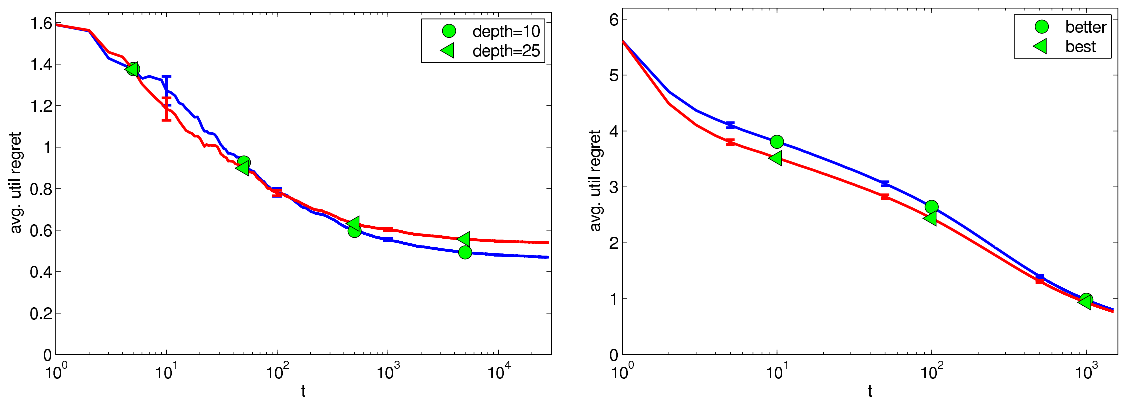

Figure 3: Regret based on **noisy feedback** for web-search (left) and movie recommendation (right).

The results for this experiment are shown in Figure 3. The first observation to make is that in the case of web-search, the regret values now do not converge to zero. Similarly, in the case of movie recommendation the regret values are higher than those in the previous experiment. These results are in line with our theory which shows regret converging to average slack variables when the user does not provide strict α informative feedback for any α. Interestingly, in the case of web-search the average regret is slightly higher when the user goes to greater depth in providing feedback. This is due to the fact that the relevance labels in this dataset are noisy – when the user maximizes (noisy) utility over a larger set of URLs, the selection of the (true) utility maximizers becomes less reliable, which degrades user feedback quality.

In the rest of this paper, for web-search we consider **noisy feedback** with depth=10. In the case of movie recommendation, we consider the **better** version of the **noisy feedback** unless we explicitly mention otherwise.

6.2.3 Batch Updates

In this section, we consider the Batch Preference Perceptron algorithm (Algorithm 2). Its regret bound from Section 4.2 scales by a factor $\sqrt{k}$ under strict α-informative feedback, if the update is made only every k iterations of the algorithm. We now verify whether empirical performance scales as suggested by the bound. For both web-search and movies, we considered both **strict α-informative feedback** and **noisy feedback**. For both types of feedback, we use the Batch Perceptron Algorithm with various values of k and report the resulting average regret.

The results of these experiments are shown in Figure 4 and Figure 5. As expected, as the value of k becomes smaller, regret converges faster. However, we observe that the

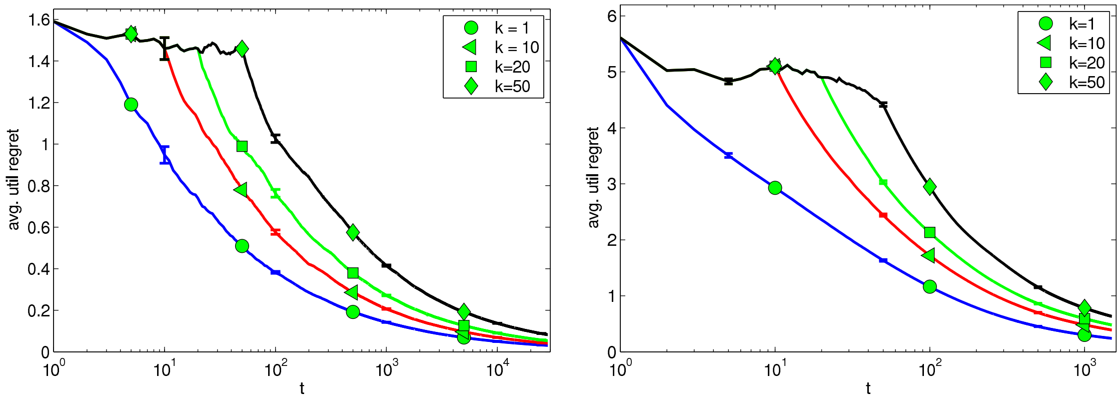

Figure 4: Regret versus time based on batch updates with **strict α-informative feedback** for web-search (left) and movie recommendation (right).

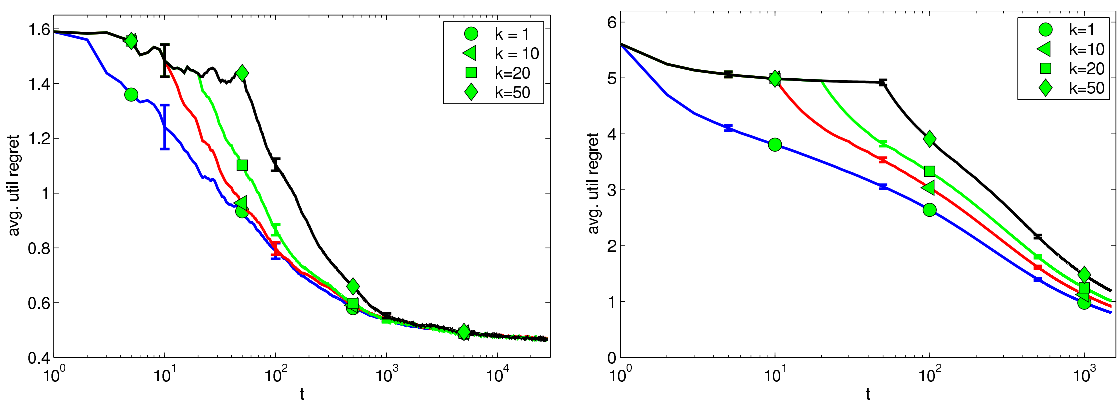

Figure 5: Regret versus time based on batch updates with **noisy feedback** for web-search (left) and movie recommendation (right).

empirical scaling with k is substantially better than the $\sqrt{k}$ factor suggested by Lemma 4. These results show the feasibility of using Coactive Learning algorithms in systems where it might be impractical to do an update after every iteration.

6.2.4 Expected User Feedback

The user feedback was deterministic in our experiments so far. In this sub-section, we consider probabilistic feedback and study the behavior of the Preference Perceptron algorithm. Recall that we provided an upper bound on the expected regret for expected user feedback in Corollary 2.

To provide α-informative feedback under expectation, we consider the following strategy. Given an object $\mathbf{y}_t$ on context $\mathbf{x}_t$, the user would first generate deterministic feedback $\mathbf{y}_t$

following a **strict α-informative feedback** model ($\alpha = 0.5$ for web-search and $\alpha = 1.0$ for movie recommendation).[5] In addition, we consider five randomly generated objects as feedback. We then put uniform probability mass over the randomly generated objects and remaining mass over the deterministic feedback such that the user feedback is still α-informative at $\alpha = 0.5$ in expectation.

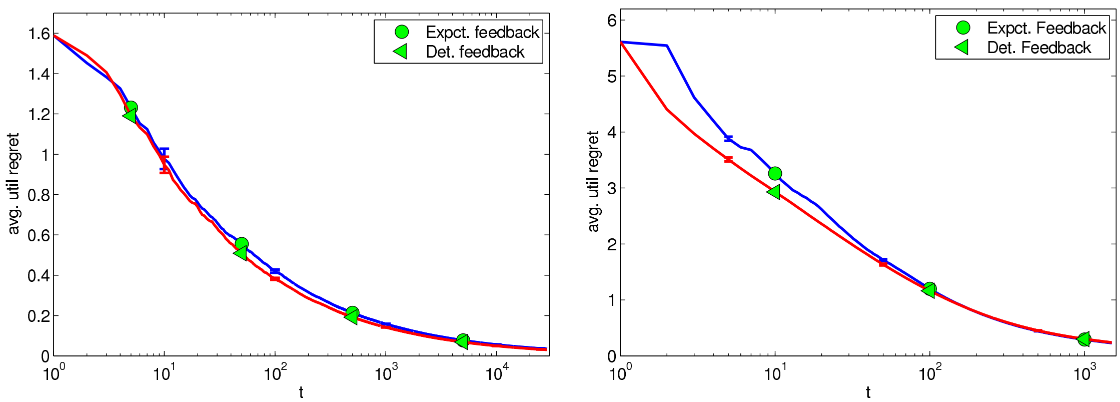

Figure 6: Expected feedback versus deterministic feedback on web-search (left) and movie recommendation (right).

The results for this experiment are shown in Figure 6. As a reference, we also plot the regret curve with deterministic α-informative feedback with $\alpha = 0.5$. It can be seen that there is not much difference between deterministic and expected feedback at higher numbers of iterations. It can also be seen that the regret converges to zero even with α-informative feedback in expectation as suggested by Corollary 2.

6.2.5 Comparison with Ranking SVM

We now compare our algorithms against several baselines, starting with a conventional Ranking SVM (Joachims, 2002) that is repeatedly trained. At each iteration, the previous SVM model is used to present a ranking to the user ($\mathbf{y}_{svm}^{q_t}$). The user returns a ranking ($\bar{\mathbf{y}}_{svm}^{q_t}$) based on **strict α-informative feedback** in one experiment and based on **noisy feedback** in the other. The pairs of examples $(q_t, \mathbf{y}_{svm}^{q_t})$ and $(q_t, \bar{\mathbf{y}}_{svm}^{q_t})$ are used as training pairs for the ranking SVM. Note that training a ranking SVM after each iteration would be prohibitive expensive, 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-

5. Note that, in the case of web-search, our user model can provide strictly β-informative where β larger than 0.5.

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.

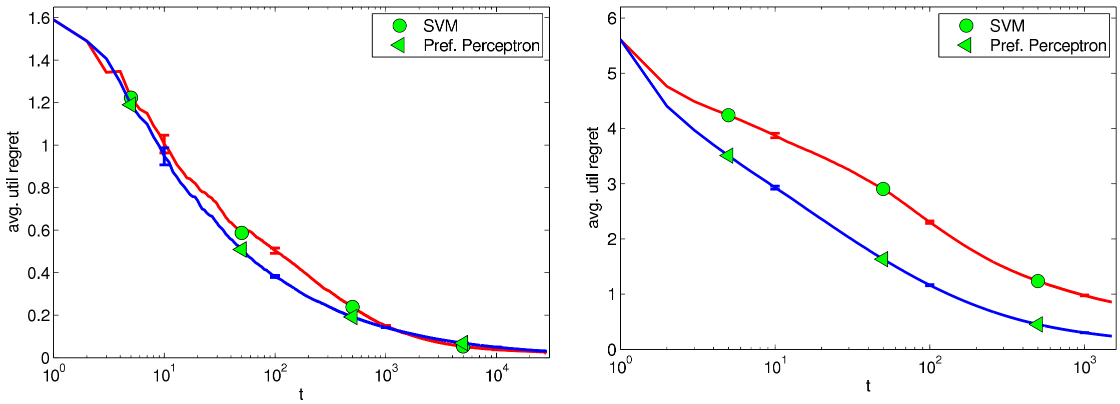

Figure 7: Preference Perceptron versus Ranking SVM with **strict α-informative feedback** on web-search (left) and movie recommendation (right).

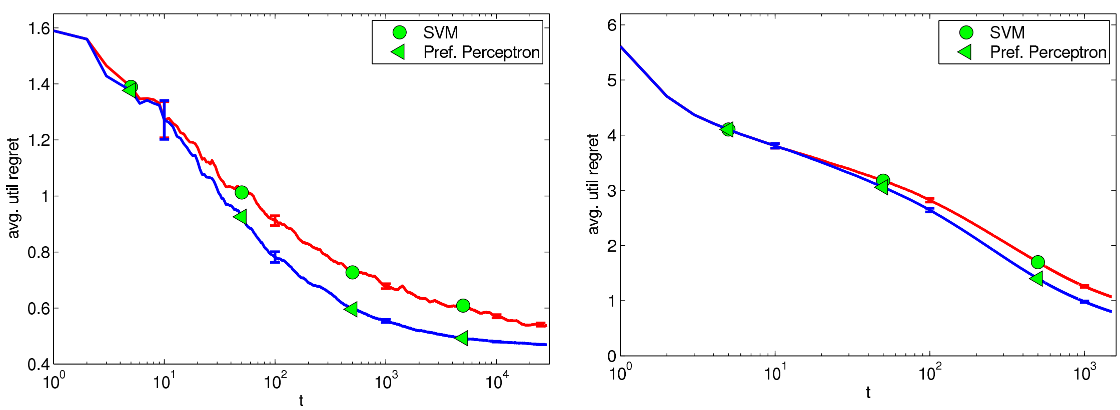

Figure 8: Preference Perceptron versus Ranking SVM with **noisy feedback** on web-search (left) and movie recommendation (right).

The results of this experiment are shown in Figure 7 and Figure 8. In the case of **strict α-informative feedback**, the Preference Perceptron performed much better than the SVM for the movie recommendation, and comparably for web search. In the case of **noisy feedback**, the Preference Perceptron performs significantly better than the SVM over most of the range on both the datasets. While it took around 20 minutes to run the Preference Perceptron experiment, it took about 20 hours to run the SVM experiment on web-dataset for each permutation of the dataset. Similary, on the movie recommendation task it took around 125 seconds to run the preference perceptron for each user while it took around 260 seconds to run the SVM for each user. These results show that the preference perceptron can perform on par or better than SVMs on these tasks at a fraction of the computational cost.

6.2.6 Comparison with Dueling Bandit

As a second baseline, we compare the Preference Perceptron algorithm with the dueling bandit approach of Yue and Joachims (2009). In each step, the dueling bandit algorithm makes a comparison between a vector $\mathbf{w}$ and a perturbed version of it $\mathbf{w}_1$ (in a random direction $\mathbf{u}$ such that $\mathbf{w}_1 = \mathbf{w} + \gamma\mathbf{u}$). The results produced by these two weight vectors are assessed by the user through techniques such as interleaving (Radlinski, Kurup, & Joachims, 2008), providing a preference between $\mathbf{w}$ and $\mathbf{w}_1$. The preference feedback determines the update that the dueling bandits algorithm makes to $\mathbf{w}$. If $\mathbf{w}$ is preferred, it is retained for the next round. If $\mathbf{w}_1$ is preferred, a small step of length δ is taken in the direction of perturbation $\mathbf{u}$.

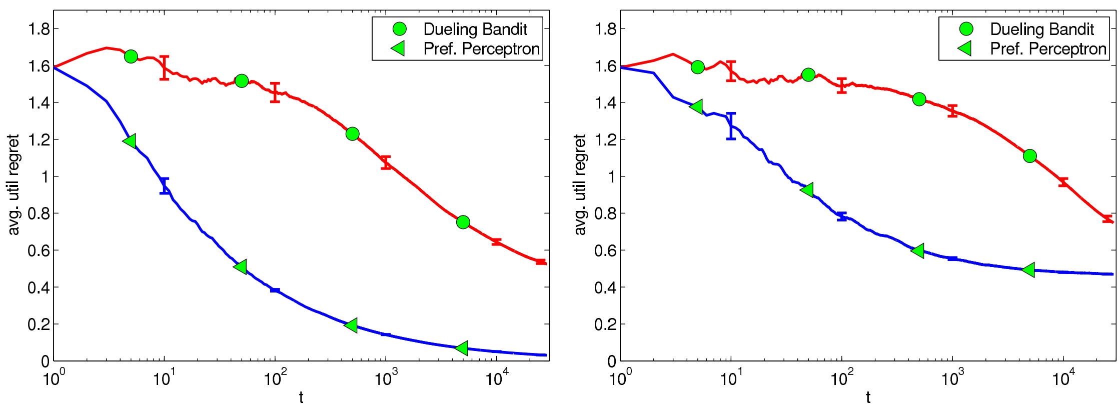

Figure 9: Preference Perceptron versus Dueling Bandit on web-search. The left plot is based on **strict** α**-informative feedback**, the right plot shows **noisy feedback**.

In our first experiment on web-search, in each step, we first obtained two ranked lists based on $\mathbf{w}$ and $\mathbf{w}_1$. The features used to obtain these ranked lists were identical to those used for Preference Perceptron. The two rankings were then interleaved. The interleaved ranking was presented to a user. In the first experiment, the user provided **strict** α**-informative feedback** on the interleaved ranking. In the second experiment, the user provided **noisy feedback**. Depending on the feedback, we inferred which of the two rankings was preferred using the Team-Game method proposed by Radlinski et al. (2008). When $\mathbf{w}$ was preferred or when there was a tie, no step was taken. When $\mathbf{w}_1$ was preferred, a step of length δ was taken in the direction $\mathbf{u}$. The regret of the dueling bandit algorithm was measured by considering the utility of the interleaved ranking. Unlike the Preference Perceptron algorithm, the dueling bandit algorithm has two parameters (γ and δ) that need to be tuned. We considered 25 values for these parameters (5x5 grid) and simply chose the best parameter values of the dueling bandits algorithm in hindsight.

The results for this experiment are shown in Figure 9. Despite the advantage of setting the parameters to best possible values, it can be seen that dueling bandit algorithm performs significantly worse compared to the preference perceptron algorithm by orders of magnitude. For example, the performance of the dueling bandit at around 28,000 iterations is matched by preference perceptron at less than 100 iterations with both types of feedback. This is not surprising, since the dueling bandit algorithm basically relies on random vectors to determine the direction in which a step needs to be taken. In the Coactive Learning model, the user feedback provides a (better than random) direction to guide the algorithm.

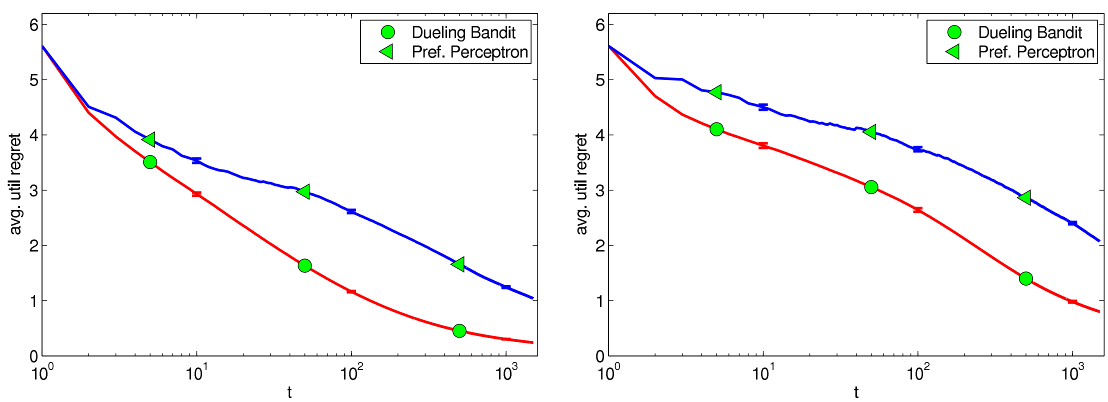

Figure 10: Preference Perceptron versus Dueling Bandit on movie recommendation. The left plot is based on utility values whereas the right plot shows results with rounded values.

Similarly, we also conducted a comparison with the dueling bandit algorithm on the movie recommendation dataset. However, unlike the web-search experiment, the dueling bandit model is somewhat unnatural on this dataset in our experimental setup, since interleaving two rankings is natural whereas interleaving two items is not. We therefore consider a different setup. Two movies were obtained based on $\mathbf{w}$ and $\mathbf{w}_1$ for the dueling bandit algorithm. User feedback was to merely indicate which of these two movies has a higher rating. In the noisy case, user feedback was based on the actual rating or the rounded rating. In the noise-free case, user feedback was based on the utility values. In either case, the utility of dueling bandit was considered to be the average utility of the two movies selected for comparison.

The performance of the dueling bandit algorithm in this experiment is shown in Figure 10. For the Preference Perceptron algorithm, regret curves for strict α-informative feedback ($\alpha = 0.5$) and **better** noisy feedback are also shows as reference. It can be seen that the dueling bandit algorithm again performs substantially worse compared to the Preference Perceptron algorithm.

6.3 Exponentiated versus Additive Updates

In this experiment, we compare the exponentiated algorithm (Algorithm 4) with the additive Preference Perceptron algorithm. For the exponentiated algorithm, all the components of $\mathbf{w}_*^e$ must be non-negative.[6] We obtained a non-negative $\mathbf{w}_*^e$ as follows:

$$[\mathbf{w}_*^e]_i = \begin{cases} \min(0, [\mathbf{w}_*]_i) & 1 \leq i \leq m, \\ \max(0, -[\mathbf{w}_*]_{i-m}) & m+1 \leq i \leq 2m. \end{cases} \tag{22}$$

In the above equation, $[\mathbf{w}_*^e]_i$ denotes the i^{th} component of $\mathbf{w}_*^e$. Moreover, we also modified the joint feature map for the exponentiated algorithm as follows:

$$[\phi^e(\mathbf{x}, \mathbf{y})]_i = \begin{cases} +[\phi(\mathbf{x}, \mathbf{y})]_i & 1 \leq i \leq m \\ -[\phi(\mathbf{x}, \mathbf{y})]_{i-m} & m+1 \leq i \leq 2m \end{cases} \tag{23}$$

With the above modifications, $\mathbf{w}_*^e$ will have only non-negative components and moreover, it is easy to verify that $\mathbf{w}_*^{e\top}\phi^e(\mathbf{x}, \mathbf{y}) = \mathbf{w}^\top\phi(\mathbf{x}, \mathbf{y})$. This makes the regret of the exponentiated algorithm directly comparable with the regret of the additive algorithm.

The exponentiated algorithm has a fixed rate parameter η that inversely depends on the time horizon T. When T is large, η is small. In this situation, consider the update in Algorithm 4:

$$\mathbf{w}_{t+1}^i \leftarrow \mathbf{w}_t^i \exp(\eta(\phi^i(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi^i(\mathbf{x}_t, \mathbf{y}_t)))/Z_t.$$

Since, η is small, we can approximate the exponential term in the above equation with a first order approximation:

$$\exp(\eta(\phi^i(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi^i(\mathbf{x}_t, \mathbf{y}_t))) \approx 1 + \eta(\phi^i(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi^i(\mathbf{x}_t, \mathbf{y}_t)).$$

Thus the exponentiated updates resemble the updates of the additive algorithm up to a normalization factor. Despite the normalization factor, we empirically observed the behavior of the two algorithms to be nearly identical (though not exact). We thus empirically evaluated the exponentiated algorithm with a variable rate parameter $\eta_t = \frac{1}{2S\sqrt{t}}$ at time t. Note that this is an empirical result without formal theoretical guarantees for this variable rate.

Results of this experiment are shown in Figure 11 and Figure 12 for **strict α-informative feedback** and **noisy feedback** respectively. It can be seen that the exponentiated algorithm tends to performs slightly better than the additive algorithm for small number of iterations. As the time horizon becomes large, the two algorithms seem to have comparable performance in most cases.

6.4 Minimizing Convex Losses

In this section, we empirically evaluate the Convex Preference Perceptron (Algorithm 5) and the Second-Order Preference Perceptron (Algorithm 6).

6. We put a superscript 'e' to distinguish the joint feature map and $\mathbf{w}$ that we used in our experiments for the exponentiated algorithm.

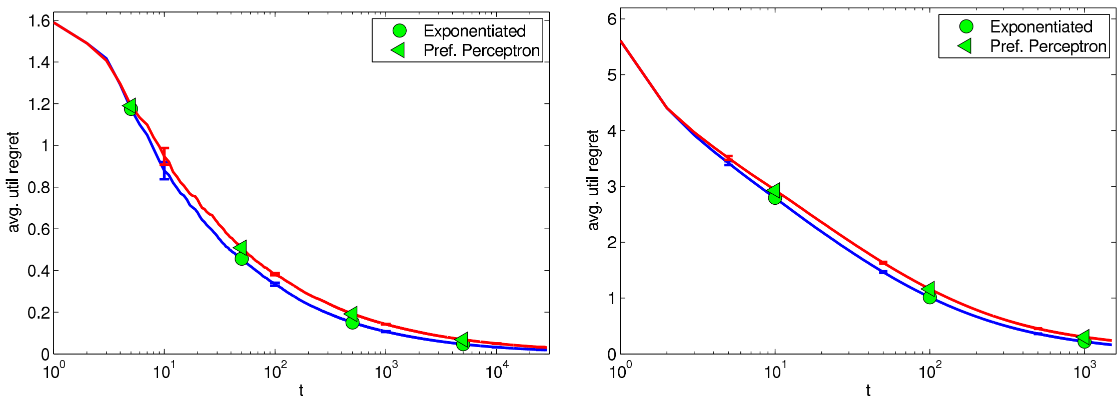

Figure 11: Exponentiated versus Additive with **strict α-informative feedback** on web-search (left) and movie recommendation (right).

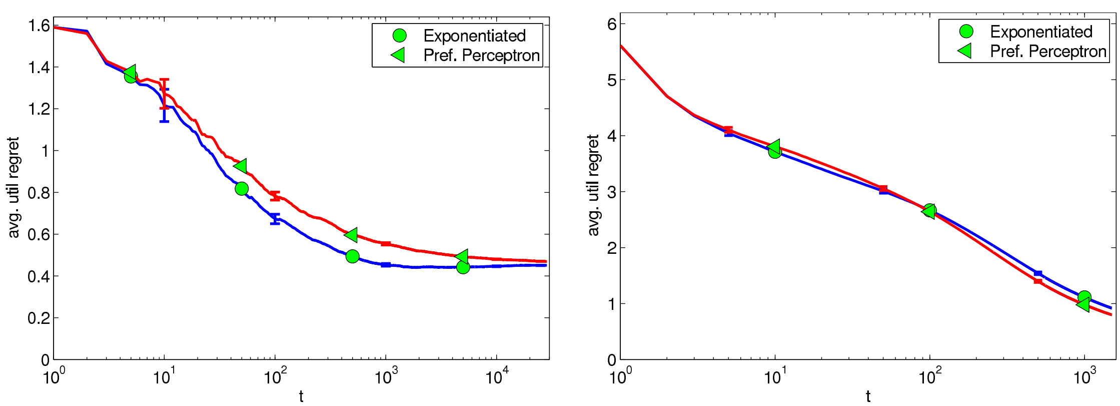

Figure 12: Exponentiated versus Additive with **noisy feedback** on web-search (left) and movie recommendation (right).

6.4.1 Convex Perceptron Versus Second-Order Algorithm

The regret bounds from Section 5 show that one can get lower regret for λ-strongly convex functions using a second-order algorithm, while the first-order Convex Perceptron applies to general convex functions. In this section, we evaluate the relative performance of the first-order and the second-order algorithms empirically. For this purpose, we considered the quadratic loss $c(\theta) = (\theta - M)^2$ where M is the largest utility value on any possible $\phi(\mathbf{x}, \mathbf{y})$ with any $\mathbf{w}$ in a convex ball of radius $\|\mathbf{w}_*\|$. It can be verified this loss function is λ-strongly convex. $\mathcal{B}$ was set to 100 for both the algorithms for both the datasets.

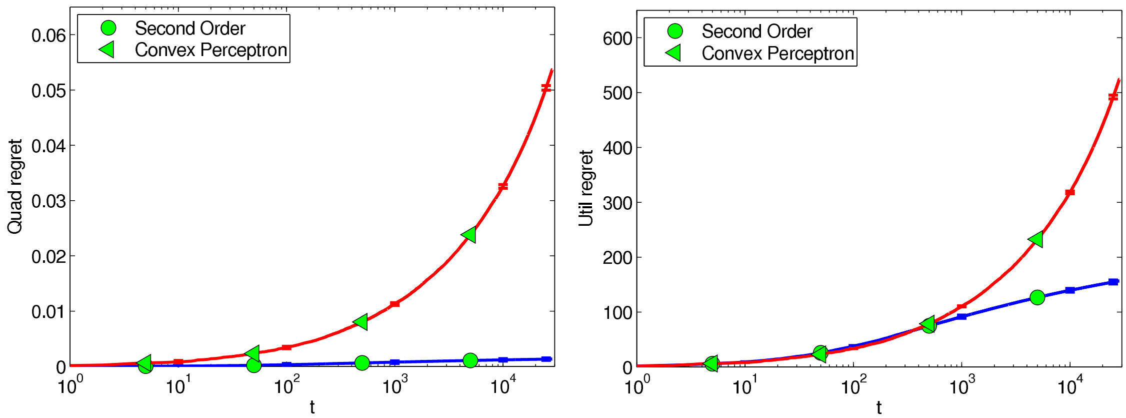

Figure 13: Cumulative regret of the convex perceptron and the second order convex perceptron for web-search.

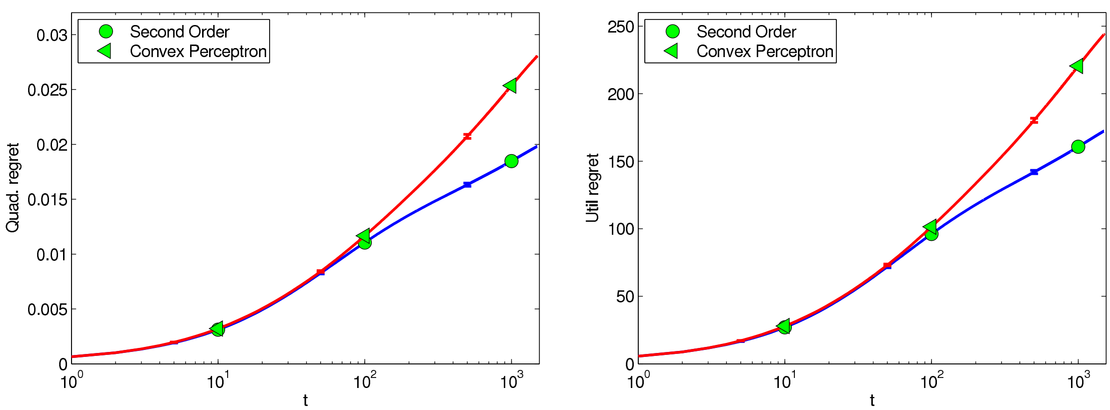

Figure 14: Cumulative regret of the convex perceptron and the second order convex perceptron for movie recommendation.

In the first set of experiments, we considered **strict α-informative feedback**. We ran both the second-order algorithm as well as the Convex Preference Perceptron algorithm 5. The γ value in the second order perceptron was simply set to one. We recorded the REG_T and $CREG_T$ values for both the methods. Note that REG_T corresponds to the utility regret as defined in 4.

Results of this experiment are shown in Figure 13 and Figure 14. To demonstrate the qualitative difference between the two algorithms, we plot cumulative regret (i.e. $T \times REG_T$ and $T \times CREG_T$) in these figures. The cumulative regret of the second-order algorithm is linear on a log-scale. This shows that the convergence of the regret is indeed logarithmic, compared to the much slower convergence of the Convex Preference Perceptron. Interestingly, even the cumulative regret based on raw utility values empirically shows a similar behavior. This is purely an empirical result, since theoretically, $\mathcal{O}(\log(T)/T)$ average regret holds only for strongly convex losses and the linear loss is not strongly convex.

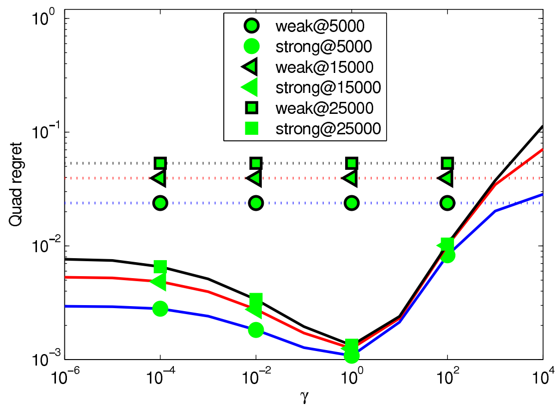

Figure 15: Sensitivity of the second order preference perceptron algorithm to the parameter value γ.

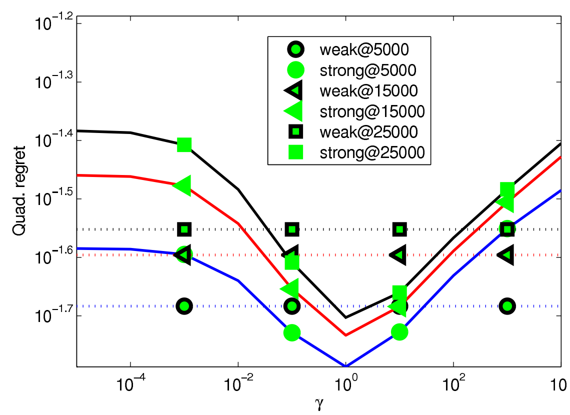

Figure 16: Sensitivity of the second order preference perceptron algorithm to the parameter value γ on movie recommendation.

In the previous experiment, we fixed the γ value in the second-order algorithm to one. We now study the sensitivity of the second-order algorithm to the value of this parameter. Figures 15 and 16 show regret values after a given number of iterations when γ is swept over a range of values. The dotted lines show the performance of the Convex Preference Perceptron for comparison. In the case of web-search, there is a wide range of parameter values where the performance of the algorithm is good. As the parameter γ takes an extreme value on either side, the performance of the algorithm deteriorates. The range of suitable γ values is much broader for the web-search dataset than for movie recommendation. It is interesting to note that both the algorithms performed empirically best at $\gamma = 1$ among the values that were tried.

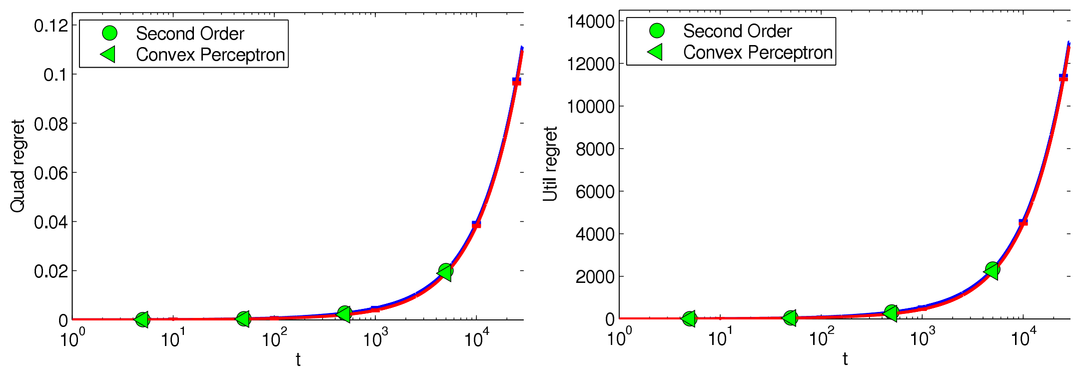

Figure 17: Strong convex versus weak convex with noisy feedback on web-seach.

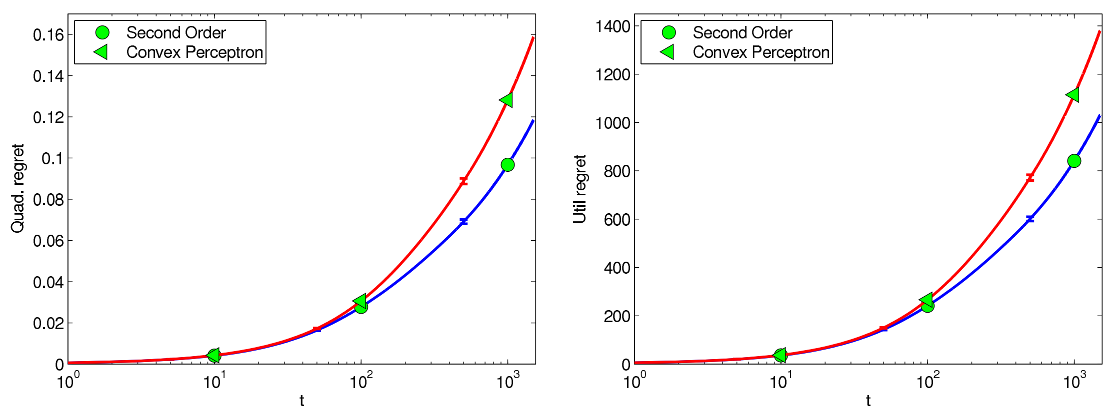

Figure 18: Strong convex versus weak convex with noisy feedback on movie recommendation.

We also tested the convex algorithms under **noisy feedback**. Both regret bounds contain the slack terms on the right hand side. Thus, when user feedback is not α-informative for any α, the regret bounds for the second-order algorithm and the first-order algorithm are both dominated by the slack variables. The empirical performance of the two algorithms under **noisy feedback** are shown in Figures 17 and 18. In the case of web-search, the results for the second-order algorithm and the first-order algorithm are nearly identical. However, in the case of movie recommendation, there is still some advantage to the second-order algorithm.

In summary, the second-order algorithm performs substantially superior under no-noise circumstances. In the presence of noise in the feedback, the two algorithms do not show drastically different behaviors.

7. Conclusions

We proposed Coactive Learning as a new model of online learning with preferences that is especially suited for implicit user feedback. Unlike most supervised learning approaches, Coactive Learning algorithms do not require optimal labels, but merely (noisy) feedback that improves over the prediction. Our model, where no cardinal utilities are observed, sits between the experts and the bandits settings, and we argue that Coactive Learning is applicable to a wide range of systems that aim to optimize themselves based on observable user actions.

We provide several algorithms that provably optimize regret in the Coactive Learning framework, and we empirically validate the effectiveness of the proposed framework on web-search ranking and movie recommendation datasets with simulations of both noisy and noise-free feedback. A recurring theme in this paper is that a wide variety of conventional online learning algorithms can be converted into Coactive Learning algorithms, despite the differences in the learning model itself, in the nature of feedback and in the notion of regret. We conjecture that many other online learning algorithms could similarly be converted to practically useful Coactive Learning algorithms.

The Coactive Learning model, the algorithms we proposed, and the ability to use weak feedback from observable user behavior offer a wide range of opportunities for new learning approaches to application problems ranging from natural language processing and information retrieval to robotics. There are also several opportunities for further developing algorithms for the Coactive Learning model. For example, our algorithm for convex loss minimization assume only that the gradient of the convex losses are bounded. However, in most practical situations, the convex loss to be minimized is known apriori. It is an interesting research direction to study whether there are algorithms that can utilize the gradient of the loss to perform better either theoretically or empirically. Another question is whether better algorithms exist for special cases of the linear utility model. Our lower bound is based on an argument where the dimensionally of the joint feature maps grow with the given horizon T. When the dimensionality of the joint feature map is fixed, an interesting research question is: are there algorithms with better regret than our proposed algorithms?

Acknowledgments

This work was funded in part under NSF awards IIS-0905467, IIS-1247637, and IIS-1142251. This was work was done when Pannaga Shivaswamy was a postdoctoral associate at Cornell University. We thank Peter Frazier, Bobby Kleinberg, Karthik Raman, Tobias Schnabel and

Yisong Yue for helpful discussions. We also thank anonymous reviewers for their thoughtful comments on an earlier version of this paper.

Appendix A. Proof of Theorem 5

Proof We look at how the KL divergence between $\mathbf{w}$ and $\mathbf{w}_t$ evolves,

$$\begin{aligned} KL(\mathbf{w}||\mathbf{w}_t) - KL(\mathbf{w}||\mathbf{w}_{t+1}) &= \sum_{i=1}^{N} \mathbf{w}^i \log(\mathbf{w}^i_{t+1}/\mathbf{w}^i_t) \\ &= \sum_{i=1}^{N} \mathbf{w}^i(\eta(\phi^i(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi^i(\mathbf{x}_t, \mathbf{y}_t))) - \log(Z_t) \\ &= \eta \mathbf{w}^\top(\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)) - \log(Z_t). \end{aligned} \tag{24}$$

On the second line, we pulled out $\log(Z_t)$ from the sum since $\sum_{i=1}^N \mathbf{w}^i = 1$. Now, consider the last term in the above equation. Denoting $\phi^i(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi^i(\mathbf{x}_t, \mathbf{y}_t)$ by $\Delta^i \phi_t$ for brevity, we have, by definition,

$$\begin{aligned} \log(Z_t) &= \log\left(\sum_{i=1}^{N} \mathbf{w}^i_t \exp(\eta \Delta^i \phi_t)\right) \\ &\le \log\left(\sum_{i=1}^{N} \mathbf{w}^i_t (1 + \eta\Delta^i\phi_t + \eta^2 \Delta^i {\phi_t}^2)\right) \\ &\le \log\left(1 + \eta \mathbf{w}_t^\top \Delta\phi_t + \eta^2 S^2\right) \\ &\le \eta \mathbf{w}_t^\top \Delta\phi_t + \eta^2 S^2. \end{aligned} \tag{25}$$

On the second line we used the fact that $\exp(x) \le 1 + x + x^2$ for $x \le 1$. The rate η ensures that $\eta(\Delta^i\phi) \le 1$. On the last line, we used the fact that $\log(1+x) \le x$. Combing (24) and (25), we get,

$$(\mathbf{w} - \mathbf{w}_t)^\top \Delta\phi_t \le \frac{KL(\mathbf{w}||\mathbf{w}_t) - KL(\mathbf{w}||\mathbf{w}_{t+1})}{\eta} + \eta S^2.$$

Adding the above inequalities, we get:

$$\begin{aligned} \sum_{t=1}^{T} (\mathbf{w} - \mathbf{w}_t)^\top(\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)) &\le \sum_{t=1}^{T} \frac{KL(\mathbf{w}||\mathbf{w}_t) - KL(\mathbf{w}||\mathbf{w}_{t+1})}{\eta} + \sum_{t=1}^{T-1} \eta S^2. \\ &\le \frac{KL(\mathbf{w}||\mathbf{w}_0)}{\eta} + \eta S^2 T. \end{aligned}$$

Rearranging the above inequality, and substituting the value of η from Algorithm 4, we get:

$$\begin{aligned} \sum_{t=1}^{T} (U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t)) &\le \sum_{t=1}^{T} \mathbf{w}_t^\top(\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)) + 2\log(N) S\sqrt{T} + \frac{S\sqrt{T}}{2} \\ &\le 2\log(N) S\sqrt{T} + \frac{S\sqrt{T}}{2}. \end{aligned}$$

In the above, we also used the fact that $KL(\mathbf{w}||\mathbf{w}_1) \le \log(N)$ since $\mathbf{w}_1$ is initialized uniformly. Moreover, from **Hölder's** inequality, we obtained

$$\mathbf{w}_t^\top \phi(\mathbf{x}_t, \mathbf{y}_t) \le \|\mathbf{w}_t\|_{\ell_1} \|\phi(\mathbf{x}_t, \mathbf{y}_t)\|_{\ell_\infty} \le S.$$

The above inequality along with α-informative feedback gives the claimed result. ■

Appendix B. Proof of Theorem 6

Proof First, we divide the set of time steps into two different sets based on the nature of the feedback:

$$\begin{aligned} I &:= \{t : U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t)) \ge 0; 1 \le t \le T\}, \\ J &:= \{t : U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t)) < 0; 1 \le t \le T\}. \end{aligned}$$

For brevity we denote $\phi(\mathbf{x}_t, \mathbf{a}) - \phi(\mathbf{x}_t, \mathbf{b})$ by[7] $\Delta(\mathbf{a}, \mathbf{b})$ in the rest of this proof. We start by considering the following term for a single time step t:

$$\begin{aligned} & c_t(U(\mathbf{x}_t, \mathbf{y}_t) - U(\mathbf{x}_t, \mathbf{y}_t^*)) - c_t(0) \\ \le & c_t(U(\mathbf{x}_t, \mathbf{y}_t) - U(\mathbf{x}_t, \mathbf{y}_t^*)) - c_t\left(\frac{\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha}\right) \\ = & c_t\left(\frac{\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} - \frac{\xi_t}{\alpha}\right) - c_t\left(\frac{\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha}\right) \\ \le & \left(\frac{(\mathbf{w}_* - \mathbf{w}_t)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} - \frac{\xi_t}{\alpha}\right) c_t'\left(\frac{\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} - \frac{\xi_t}{\alpha}\right) \\ \le & \begin{cases} (\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) + \xi_t^+ - \mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t))G/\alpha & t \in I \\ (\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) + \xi_t^+)G/\alpha & t \in J. \end{cases} \end{aligned}$$

In the above inequalities, the second line follows from the fact that $\frac{\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} \ge 0$ and $c_t(\cdot)$ is non-increasing. The third line follows from α-informative feedback (Eqn. (6)). The fourth line follows since the function c_t is convex.[8] We obtain the first term in the next inequality (in either case) since $c_t'(\cdot) \in [-G, 0]$ and $\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) \ge 0$ from the choice of $\mathbf{y}_t$ in the algorithm. The second terms (in either case) is obtained by the fact that $-\xi_t c_t'(\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t))$ is upper bounded by $\xi_t^+ G$. This is the step in which the clipped version of the slack variables are needed in the proof. Finally, $\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)$ is either positive or negative depending on the feedback which leads to two different cases depending on whether $t \in I$ or $t \in J$.

7. Since the context $\mathbf{x}_t$ will always be clear, we suppress this in our notation for brevity.
8. For any convex function f, $f(y) - f(x) \le (y - x) f'(y)$ where $f'(y)$ denotes a sub-derivative of f at y.

Summing the above inequalities from 1 to T , we get:

$$
\begin{aligned}
&\sum_{t=1}^{T} c_t(\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \mathbf{y}_t^*)) - \sum_{t=1}^{T} c_t\,(0) \\
\leq& \frac{G}{\alpha}\sum_{t=1}^{T} \mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) + \frac{G}{\alpha}\sum_{t=1}^{T} \xi_t^+ - \frac{G}{\alpha}\sum_{t\in I} \mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) \\
\leq& \frac{G}{\alpha}\sum_{t=1}^{T} (\mathbf{w}_t - \mathbf{w}_*)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) + \frac{G}{\alpha}\sum_{t=1}^{T} \xi_t^+ + \frac{G}{\alpha}\sum_{t\in J} \mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t).
\end{aligned} \tag{26}
$$

We obtained the last line above simply by adding and and subtracting $G\sum_{t\in J}\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)/\alpha$ on the right side of the previous inequality. From this point, we mostly follow the proof techniques from online convex optimization (Zinkevich, 2003).

We now bound the first term on the right hand side of (26). For this purpose, consider the following:

$$
\begin{aligned}
\|\bar{\mathbf{w}}_{t+1} - \mathbf{w}_*\|^2 &= \|\mathbf{w}_t + \eta_t \Delta(\bar{\mathbf{y}}_t, \mathbf{y}_t) - \mathbf{w}_*\|^2 \\
&= \|\mathbf{w}_t - \mathbf{w}_*\|^2 + \eta_t^2 \|\Delta(\bar{\mathbf{y}}_t, \mathbf{y}_t)\|^2 + 2\eta_t(\mathbf{w}_t - \mathbf{w}_*)^\top \Delta(\bar{\mathbf{y}}_t, \mathbf{y}_t).
\end{aligned} \tag{27}
$$

Rearranging terms in the above equation, we get:

$$
\begin{aligned}
(\mathbf{w}_t - \mathbf{w}_*)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) =& \frac{1}{2\eta_t}\|\mathbf{w}_t - \mathbf{w}_*\|^2 - \frac{1}{2\eta_t}\|\bar{\mathbf{w}}_{t+1} - \mathbf{w}_*\|^2 + \frac{\eta_t}{2}\|\Delta(\bar{\mathbf{y}}_t, \mathbf{y}_t)\|^2 \\
\leq& \frac{1}{2\eta_t}\|\mathbf{w}_t - \mathbf{w}_*\|^2 - \frac{1}{2\eta_t}\|\mathbf{w}_{t+1} - \mathbf{w}_*\|^2 + 2\eta_t R^2
\end{aligned}
$$

where, on the last line, we used the fact that $\|\mathbf{w}_{t+1} - \mathbf{w}_*\|^2 \leq \|\bar{\mathbf{w}}_{t+1} - \mathbf{w}_*\|^2$ since the $\mathbf{w}_{t+1}$ is just the projection of $\bar{\mathbf{w}}_{t+1}$ to the convex set $\mathcal{B}$ (which contains the vector $\mathbf{w}_*$). We can bound the first term in (26) using the following telescoping argument.

$$
\begin{aligned}
&\sum_{t=1}^{T}\left(\frac{1}{2\eta_t}\|\mathbf{w}_t - \mathbf{w}_*\|^2 - \frac{1}{2\eta_t}\|\mathbf{w}_{t+1} - \mathbf{w}_*\|^2 + 2\eta_t R^2\right) \\
\leq& \frac{1}{2\eta_1}\|\mathbf{w}_1 - \mathbf{w}_*\|^2 + \sum_{t=2}^{T}\left(\frac{1}{2\eta_t} - \frac{1}{2\eta_{t-1}}\right)\|\mathbf{w}_t - \mathbf{w}_*\|^2 + 2R^2\sum_{t=1}^{T}\eta_t \\
\leq& \frac{1}{2\eta_1}|\mathcal{B}| + \sum_{t=2}^{T}\left(\frac{1}{2\eta_t} - \frac{1}{2\eta_{t-1}}\right)|\mathcal{B}| + 2R^2(2\sqrt{T} - 1) \\
\leq& \frac{\sqrt{T}+1}{2}|\mathcal{B}| + 4R^2\sqrt{T}.
\end{aligned}
$$

In the above, we obtained the second line by simply rearranging the terms in the expression above. On the third line, we used the boundedness property of the set $\mathcal{B}$, as well as the fact $\sum_{t=1}^{T}\eta_t \leq 2\sqrt{T} - 1$. The final line follows from cancelling out terms and the fact that $\eta_T = 1/\sqrt{T}$.

Now, consider the third term on the right hand side of (26):

$$\frac{\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} \leq \mathbf{w}_*^\top \Delta(\mathbf{y}_t, \mathbf{y}_t^*) + \frac{\xi_t^+}{\alpha} \leq \frac{\xi_t^+}{\alpha}.$$

The first inequality above follows from α-informative feedback. Whereas the second inequality follows from the fact $\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \mathbf{y}_t^*) \leq 0$ from the definition of $\mathbf{y}_t^*$. Finally, the bound (16) follows from the trivial fact $0 \leq \frac{G}{\alpha} \sum_{t \in I} \xi_t^+$.

To obtain the bound on the expected regret, consider the convex loss at step t conditioned on user behavior so far:

$$\begin{aligned}
& c_t(\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \mathbf{y}_t^*)) - \mathbf{E}_t c_t \left(\frac{\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} \right) \\
\leq & c_t \left(\frac{\mathbf{E}_t[\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) - \bar{\xi}_t]}{\alpha} \right) - \mathbf{E}_t c_t \left(\frac{\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} \right) \\
\leq & \mathbf{E}_t c_t \left(\frac{\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) - \bar{\xi}_t}{\alpha} \right) - \mathbf{E}_t c_t \left(\frac{\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} \right) \\
\leq & \begin{cases} G\mathbf{E}_t[\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) + \bar{\xi}_t^+ - \mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)]/\alpha & t \in I \\ G\mathbf{E}_t[\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) + \bar{\xi}_t^+]/\alpha & t \in J \end{cases}
\end{aligned}$$

where the second line follows from the definition of expected α-informative feedback and the third line follows from Jensen's inequality. We obtain the last line following an argument similar to that in the proof of Theorem 6. The bound follows from an expected version of (27). ■

Appendix C. Proof of Theorem 8

Proof First, we divide time steps into two different sets based on the nature of feedback:

$$\begin{aligned}
I &:= \{t : U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t)) \geq 0; 1 \leq t \leq T\}, \\
J &:= \{t : U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t)) < 0; 1 \leq t \leq T\}.
\end{aligned}$$

We start by considering a single time step t, we have:

$$
\begin{aligned}
& c_t(\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \mathbf{y}_t^*)) - c_t(0) \\
\leq & c_t(\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \mathbf{y}_t^*)) - c_t\left(\frac{\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha}\right) \\
\leq & c_t\left(\frac{\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} - \frac{\xi_t^+}{\alpha}\right) - c_t\left(\frac{\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha}\right) \\
\leq & \left(\frac{(\mathbf{w}_* - \mathbf{w}_t)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} - \frac{\xi_t^+}{\alpha}\right) c_t'\left(\frac{\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} - \frac{\xi_t^+}{\alpha}\right) - \frac{\lambda}{2}\left(\frac{(\mathbf{w}_* - \mathbf{w}_t)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} - \frac{\xi_t^+}{\alpha}\right)^2 \\
\leq & \begin{cases} G\left(\left(\frac{\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} + \frac{\xi_t^+}{\alpha} - \frac{\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha}\right) - \frac{\gamma}{2}\left(\frac{(\mathbf{w}_* - \mathbf{w}_t)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} - \frac{\xi_t^+}{\alpha}\right)^2\right) & t \in I \\
G\left(\left(\frac{\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} + \frac{\xi_t^+}{\alpha}\right) - \frac{\gamma}{2}\left(\frac{(\mathbf{w}_* - \mathbf{w}_t)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} - \frac{\xi_t^+}{\alpha}\right)^2\right) & t \in J. \end{cases}
\end{aligned} \tag{28}
$$

In the above inequalities, the second line follows from the fact that $\frac{\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} \geq 0$ and $c_t(\cdot)$ is non-increasing. The third line follows from the fact that the function $c_t(\cdot)$ is non-increasing and the following inequality which follows from the definition of ξ_t^+:

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

The fourth line follows by strong convexity. The last line follows from a same line of reasoning as in the proof of Theorem 6.

Now consider the last term in both the cases:

$$
\begin{aligned}
& -\frac{\gamma}{2}\left(\frac{(\mathbf{w}_* - \mathbf{w}_t)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} - \frac{\xi_t^+}{\alpha}\right)^2 \\
= & -\frac{\gamma}{2}\left(\frac{(\mathbf{w}_* - \mathbf{w}_t)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha}\right)^2 - \frac{\gamma \xi_t^{+2}}{2\alpha^2} + \frac{\gamma \xi_t^+ (\mathbf{w}_* - \mathbf{w}_t)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha^2} \\
= & -\frac{\gamma}{2}\left(\frac{(\mathbf{w}_* - \mathbf{w}_t)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha}\right)^2 + \frac{\gamma \xi_t^+ \mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha^2} - \frac{\gamma \xi_t^+ \mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha^2} - \frac{\gamma \xi_t^{+2}}{2\alpha^2} \\
\leq & -\frac{\gamma}{2}\left(\frac{(\mathbf{w}_* - \mathbf{w}_t)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha}\right)^2 + \frac{\gamma \xi_t^+}{\alpha}\left(\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \mathbf{y}_t^*) + \frac{\xi_t^+}{\alpha}\right) - \frac{\gamma \xi_t^{+2}}{2\alpha^2} \\
\leq & -\frac{\gamma}{2}\left(\frac{(\mathbf{w}_* - \mathbf{w}_t)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha}\right)^2 + \gamma \frac{\xi_t^{+2}}{2\alpha^2}.
\end{aligned} \tag{29}
$$

In the above equations, the second and the third lines follow from simple algebraic expansion of the expression on the first line. The fourth line follows from the definition of α-informative feedback and the fact that $\mathbf{w}_t^\top \Delta(\bar{\mathbf{y}}_t, \mathbf{y}_t) \leq 0$. The last line follows from the fact that $\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \mathbf{y}_t^*) \leq 0$ from the definition of $\mathbf{y}_t^*$.

Now, summing the terms in (28) and then substituting the above bound, we get,

$$
\begin{aligned}
&\sum_{t=1}^{T} c_t(\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \mathbf{y}_t^*)) - \sum_{t=1}^{T} c_t\,(0) \\
&\leq G \sum_{t=1}^{T} \left(\frac{\mathbf{w}_t^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} + \frac{\xi_t^+}{\alpha} - \frac{\gamma}{2} \left(\frac{(\mathbf{w}_* - \mathbf{w}_t)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} \right)^2 \right) - G \sum_{t \in I} \frac{\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} + \frac{\gamma \sum_{t=1}^{T} \xi_t^{+2}}{2\alpha^2} \\
&\leq G \sum_{t=1}^{T} \left(\frac{(\mathbf{w}_t - \mathbf{w}_*)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} - \frac{\gamma}{2} \left(\frac{(\mathbf{w}_* - \mathbf{w}_t)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha} \right)^2 \right) \\
&\quad + \frac{G}{\alpha} \sum_{t \in J} \mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) + \frac{\gamma \sum_{t=1}^{T} \xi_t^{+2}}{2\alpha^2} + \frac{G \sum_{t=1}^{T} \xi_t^+}{\alpha} \\
&\leq \frac{G}{\alpha} \sum_{t=1}^{T} \left((\mathbf{w}_t - \mathbf{w}_*)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) - \frac{\gamma}{2} \left((\mathbf{w}_t - \mathbf{w}_*)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) \right)^2 \right) + \frac{\gamma \sum_{t=1}^{T} \xi_t^{+2}}{2\alpha^2} + \frac{2G \sum_{t=1}^{T} \xi_t^+}{\alpha}.
\end{aligned}
$$

In the above, we obtained the third inequality by adding and subtracting $G \sum_{t \in J} \frac{\mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)}{\alpha}$. To obtain the last line, we used the fact that $-1/\alpha^2 \leq -1/\alpha$ since $\alpha \in (0, 1]$). Finally, we used an argument similar to that in the proof of theorem 6 to bound $\frac{G}{\alpha} \sum_{t \in J} \mathbf{w}_*^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)$ and obtained a factor of two with the sum of slacks term. From this point, we use arguments similar to those from online convex optimization with strongly convex losses (Hazan et al., 2007).

Next, we consider $(\bar{\mathbf{w}}_{t+1} - \mathbf{w}_*)^\top A_t (\bar{\mathbf{w}}_{t+1} - \mathbf{w}_*)$ and express it interms of $\mathbf{w}_t$ and A_{t-1}:

$$
\begin{aligned}
&(\bar{\mathbf{w}}_{t+1} - \mathbf{w}_*)^\top A_t (\bar{\mathbf{w}}_{t+1} - \mathbf{w}_*) \\
=&(\mathbf{w}_t - A_t^{-1} \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) - \mathbf{w}_*)^\top A_t (\mathbf{w}_t - A_t^{-1} \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) - \mathbf{w}_*) \\
=&(\mathbf{w}_t - \mathbf{w}_*)^\top A_t (\mathbf{w}_t - \mathbf{w}_*) + \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)^\top A_t^{-1} \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) - 2(\mathbf{w}_t - \mathbf{w}_*)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) \\
=&\gamma (\mathbf{w}_t - \mathbf{w}_*)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)^\top (\mathbf{w}_t - \mathbf{w}_*) + (\mathbf{w}_t - \mathbf{w}_*)^\top A_{t-1} (\mathbf{w}_t - \mathbf{w}_*) \\
&\quad + \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)^\top A_t^{-1} \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) + 2(\mathbf{w}_* - \mathbf{w}_t)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)
\end{aligned}
$$

Rearranging terms in the above equation, we get:

$$
\begin{aligned}
&2(\mathbf{w}_t - \mathbf{w}_*)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) - \frac{\gamma}{2} (\mathbf{w}_t - \mathbf{w}_*)^\top \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t) \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)^\top (\mathbf{w}_t - \mathbf{w}_*) \\
\leq&(\mathbf{w}_t - \mathbf{w}_*)^\top A_{t-1} (\mathbf{w}_t - \mathbf{w}_*) - (\mathbf{w}_{t+1} - \mathbf{w}_*)^\top A_t (\mathbf{w}_{t+1} - \mathbf{w}_*) + \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t)^\top A_t^{-1} \Delta(\mathbf{y}_t, \bar{\mathbf{y}}_t).
\end{aligned}
$$

We now identify that the term on the left hand side in the inequality occurs in the expression that we would like to bound in (29). We therefore have,

$$\begin{aligned}
&2\sum_{t=1}^{T}\Big((\mathbf{w}_t-\mathbf{w}_*)^\top\Delta(\mathbf{y}_t,\bar{\mathbf{y}}_t)-\gamma((\mathbf{w}_t-\mathbf{w}_*)^\top\Delta(\mathbf{y}_t,\bar{\mathbf{y}}_t))^2\Big)\\
\leq&\sum_{t=1}^{T}\Big((\mathbf{w}_t-\mathbf{w}_*)^\top A_{t-1}(\mathbf{w}_t-\mathbf{w}_*)-(\mathbf{w}_{t+1}-\mathbf{w}_*)^\top A_t(\mathbf{w}_{t+1}-\mathbf{w}_*)+\Delta(\mathbf{y}_t,\bar{\mathbf{y}}_t)^\top A_t^{-1}\Delta(\ by_t,\bar{\mathbf{y}}_t)\Big)\\
\leq&(\mathbf{w}_1-\mathbf{w}_*)^\top A_0(\mathbf{w}_1-\mathbf{w}_*)+\sum_{t=1}^{T}\Delta(\mathbf{y}_t,\bar{\mathbf{y}}_t)^\top A_t^{-1}\Delta(\mathbf{y}_t,\bar{\mathbf{y}}_t)\\
\leq&\epsilon|\mathcal{B}|+\frac{N}{\gamma}\log\left(\frac{4R^2T\gamma}{\epsilon}+1\right).
\end{aligned}$$

In the above, we have used the fact that $\sum_{t=1}^{T}\Delta(\mathbf{y}_t,\bar{\mathbf{y}}_t)^\top A_t^{-1}\Delta(\mathbf{y}_t,\bar{\mathbf{y}}_t)\leq N\log(4R^2T/\epsilon+1)$, where N is the dimens ionality of $\phi(\mathbf{x},\mathbf{y})$ and R is an upper bound on the norm of the joint feature maps (i.e. $\|\phi(\mathbf{x},\mathbf{y})\|_{\ell_2}\leq R$. A proof of this fact can be found in Hazan et al. (2007). ■

References

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

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

Bakir, G. H., Hofmann, T., Schölkopf, B., Smola, A., Taskar, B., & Vishwanathan, S. (Eds.). (2007). *Predicting Structured Data*. The MIT Press.

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

Boley, M., Mampaey, M., Kang, B., Tokmakov, P., & Wrobel, S. (2013). One click mining: Interactive local pattern discovery through implicit preference and performance learning. In *Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics*, pp. 27–35.

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

Cesa-Bianchi, N., & Lugosi, G. (2006b). *Prediction, Learning, and Games*. Cambridge University Press, Cambridge, UK.

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

Chu, W., & Ghahramani, Z. (2005). Preference learning with gaussian processes. In *ICML*.

Crammer, K., & Singer, Y. (2001). Pranking with ranking. In *NIPS*.

Crammer, K., & Gentile, C. (2011). Multiclass classification with bandit feedback using adaptive regularization. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*.

Dekel, O., Gilad-Bachrach, R., Shamir, O., & Xiao, L. (2012). Optimal distributed online prediction using mini-batches. *JMLR*, *13*, 165202.

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

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

Goetschalckx, R., Fern, A., & Tadepalli, P. (2014). Coactive learning for locally optimal problem solving.. In *Conference of the American Association for Artificial Intelligence (AAAI)*, pp. 1824–1830.

Haddow, B., Arun, A., & Koehn, P. (2011). Samplerank training for phrase-based machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pp. 261–271, Edinburgh, Scotland. Association for Computational Linguistics.

Hazan, E., Agarwal, A., & Kale, S. (2007). Logarithmic regret algorithms for online convex optimization. *Machine Learning*, *69*(2-3), 169–192.

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

Jain, A., Wojcik, B., Joachims, T., & Saxena, A. (2013). Learning trajectory preferences for manipulators via iterative improvement. In *Neural Information Processing Systems (NIPS)*, pp. 575–583.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 133–142.

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

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

Kakade, S. M., Shalev-Shwartz, S., & Tewari, A. (2008). Efficient bandit algorithms for online multiclass prediction. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*.

Kivinen, J., & Warmuth, M. (1997). Exponentiated gradient versus gradient gradient descent for linear predictors. *Journal of Information and Computation*, *132*(1), 1–64.

Langford, J., & Zhang, T. (2007). The epoch-greedy algorithm for multi-armed bandits with side information. In *NIPS*.

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

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

Novikoff, A. (1962). On convergence proofs on perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata*, Vol. XII, pp. 615–622.

Polyak, B., & Tsypkin, Y. (1973). Pseudogradient adaptation and training algorithms. *Automatic Remote Control*, *12*, 83–94.

Radlinski, F., Kurup, M., & Joachims, T. (2008). How does clickthrough data reflect retrieval quality?. In *Conference on Information and Knowledge Management (CIKM)*.

Raman, K., & Joachims, T. (2013). Learning socially optimal information systems from egoistic users. In *European Conference on Machine Learning (ECML)*, pp. 128–144.

Raman, K., Joachims, T., Shivaswamy, P., & Schnabel, T. (2013). Stable coactive learning via perturbation. In *International Conference on Machine Learning (ICML)*, pp. 837–845.

Raman, K., Shivaswamy, P., & Joachims, T. (2012). Online learning to diversify from implicit feedback. In *KDD*.

Shivaswamy, P., & Joachims, T. (2012). Online structured prediction via coactive learning. In *ICML*.

Somers, T., & Hollinger, G. (2014). Coactive learning with a human expert for robotic monitoring. In *RSS Workshop on Robotic Monitoring*.

Weston, J., Bengio, S., & Usunier, N. (2011). Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

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

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

Zhang, Y., Lei, T., Barzilay, R., Jaakkola, T., & Globerson, A. (2014). Steps to excellence: Simple inference with refined scoring of dependency trees. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 197–207, Baltimore, Maryland. Association for Computational Linguistics.

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