Online Learning to Diversify from Implicit Feedback

Karthik Raman
Cornell University
Ithaca, NY
karthik@cs.cornell.edu

Pannaga Shivaswamy
Cornell University
Ithaca, NY
pannaga@cs.cornell.edu

Thorsten Joachims
Cornell University
Ithaca, NY
tj@cs.cornell.edu

ABSTRACT

In order to minimize redundancy and optimize coverage of multiple user interests, search engines and recommender systems aim to diversify their set of results. To date, these diversification mechanisms are largely hand-coded or relied on expensive training data provided by experts. To overcome this problem, we propose an online learning model and algorithms for learning diversified recommendations and retrieval functions from implicit feedback. In our model, the learning algorithm presents a ranking to the user at each step, and uses the set of documents from the presented ranking, which the user reads, as feedback. Even for imperfect and noisy feedback, we show that the algorithms admit theoretical guarantees for maximizing any submodular utility measure under approximately rational user behavior. In addition to the theoretical results, we find that the algorithm learns quickly, accurately, and robustly in empirical evaluations on two datasets.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Retrieval Models; I.2.6 [**Artificial Intelligence**]: Learning

General Terms

Algorithms, Experimentation, Theory

Keywords

Online Learning, Diversified Retrieval, Submodularity

1. INTRODUCTION

Modeling the dependencies between items in a ranking of results is one of the most promising directions for improving the quality of retrieval and recommendation systems. First, consider the example of a search engine and a query such as "jaguar" or "apple". For such queries, it is important to present a diverse set of results since diversity hedges against uncertainty about the users intent. Such hedging against uncertainty about the user's information need is called *extrinsic diversity* [10]. A second reason for diversity is called *intrinsic diversity* [10] where it is important to avoid redundancy and provide a set of results that cover multiple aspects of an information need. For example, of all the articles in the NY Times on a given day, a user only has time to read a small subset. Therefore, even if the user is interested in the European Debt Crisis, he may not want to read exclusively about this one topic, but rather read one article and also cover other topics. In the following, we focus on problems where such intrinsic diversity is important.

In this paper, we extend the recently proposed coactive learning model [14] to learn diversified results from implicit user feedback. In particular, we develop two algorithms for learning both relevance and the desired amount of diversity from set-valued preference data that can be derived from implicit feedback. The algorithms proposed in this paper are easy to implement and allow theoretical analysis. Furthermore, the ability to learn the desired amount of diversity based on user feedback makes the algorithms attractive for a wide range of applications where the required amount of diversity is not determined apriori. A crucial extension over the methods in [14] is that we now consider models with submodular structure, whose diminishing returns property makes it possible to avoid redundancy and increase novelty.

Coactive learning proceeds in the following online fashion. In each step, a ranking is presented to the user that (approximately) maximizes the current estimate of the submodular utility function. As feedback, the algorithm observes the (possibly diverse) set of documents the user reads in the presented ranking. After receiving this feedback, the algorithm updates its model. Even though we allow user feedback to be imperfect, noisy, and only "weakly informative" (in a specific sense), we are able to prove guarantees on the performance of the algorithm. Unlike the theorems in [14], our guarantees apply even though submodular models only allow for approximate inference. Finally, experiments demonstrate the empirical effectiveness of the proposed approach in learning both relevance and diversity.

2. RELATED WORK

Presenting a diverse set of results is an important goal in both web-search ranking as well as recommender systems research. While much prior work on diversity has focused on non-learning approaches (e.g. [2, 19, 3, 16, 4]), recently developed supervised learning methods for diversity have shown a lot of promise (e.g. [13, 18, 12, 8]). Unfortunately,

KDD'12, August 12–16, 2012, Beijing, China.

supervised learning relies on manually judged training data with multi-topic annotations, which are expensive and difficult to obtain. El-Arini and Guestrin [6] proposed an approach to discover relevant scientific literature based on a set of scientific papers. They retrieve a set of papers based on both diversity and relevance. While their approach also makes use of submodular influence measures, they assume noise-free feedback, which is unrealistic for our problem.

While some online learning methods exist that can exploit click data, those methods either cannot generalize across queries [11] and/or have a hard-coded notion of diversity that cannot be adjusted through learning [15]. Recently, Yue and Guestrin [17] proposed online learning algorithms to maximize submodular utilities and applied them to diversified retrieval. However, their model relies on observing cardinal utilities whereas in our model we only rely on implicit preference feedback. User studies for web search [7] have shown that such preference feedback can be extracted reliably from observable user behavior (e.g. clicks), while attempt to interpret click-data as cardinal utility statements were found to be biased and unreliable.

3. LEARNING PROBLEM AND MODEL

To illustrate our learning model, consider the example of a personalized news reader that users visit on a daily basis. On day t, the news reader suggests a list of articles $\mathbf{y}_t = (d_1, d_2, d_3, d_4, d_5, ...)$ and observes which of these articles are actually read by the user. We assume that the decision to read an article is influenced by two factors. First, the article must be relevant to the user's interest. Second, the decision may have dependencies with other articles in $\mathbf{y}$. For example, the user may be interested in the European debt crisis. But the user may only want to read one article related to this issue, even if $\mathbf{y}$ contains 5 relevant articles.

In this paper, we design an online learning algorithm that can model both relevance as well as interdependencies between documents. The training data we exploit are the sets of documents read by the user each day. Continuing the example from above, the system may observe that the user read articles d_3 and d_5. Obviously, we cannot conclude that $\{d_3, d_5\}$ was the optimal set of articles the user wanted to read on day t, since there may have been other articles far down the list that the user never saw. However, we can conclude that the user would have preferred the ranking $\bar{\mathbf{y}}_t = (d_3, d_5, d_1, d_2, d_4, ...)$ over the ranking $\mathbf{y}_t = (d_1, d_2, d_3, d_4, d_5, ...)$ that was presented. We refer to $\bar{\mathbf{y}}_t$ as the **user feedback ranking**.

We now define the learning problem and the user-interaction model more generally. At each round t, our algorithm presents a ranking $\mathbf{y}_t$ from a corpus $\mathbf{x}_t \in \mathcal{X}$ of candidate documents[1]. We assume that the user acts (approximately) rational according to an unknown utility function $U(\mathbf{x}_t, \mathbf{y}_t)$ that models both relevance of the documents as well as their dependencies (e.g. redundancy). In the context of such a utility function, we can interpret the user feedback as a preference between rankings. This type of preference feedback over multiple rounds t is the input for our learning model. Given the set of candidate documents $\mathbf{x}_t$, the **optimal ranking** is denoted by

$$\mathbf{y}_t^* := \arg\max_{\mathbf{y} \in \mathcal{Y}} U(\mathbf{x}_t, \mathbf{y}). \quad (1)$$

[1]In general, $\mathbf{x}_t$ can also represent a query/context.

Since the user's utility function $U(\mathbf{x}_t, \mathbf{y})$ is unknown, this optimal ranking $\mathbf{y}_t^*$ cannot be computed. The goal of the learning algorithm is to predict rankings with utility close to that of $\mathbf{y}_t^*$. Note, however, that the user feedback does not even give the optimal $\mathbf{y}_t^*$ to the algorithm (as in traditional supervised learning), but only the user feedback ranking $\bar{\mathbf{y}}_t$ is observed. To analyze the learning algorithms in the subsequent sections, we refer to any feedback that satisfies the following inequality as **strictly α-informative feedback**:

$$U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t) \geq \alpha \left(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t) \right). \quad (2)$$

The above inequality states that the utility of the user feedback ranking $\bar{\mathbf{y}}_t$ must be slightly better than the utility of the ranking $\mathbf{y}_t$ that was presented as a fraction of the difference between the utility of $\mathbf{y}_t^*$ and the utility of $\mathbf{y}_t$. As demonstrated in the example above, such a slightly improved rankings $\bar{\mathbf{y}}_t$ can be constructed as a reordering of $\mathbf{y}_t$ based on user clicks. The amount of improvement is quantified by $\alpha \in (0, 1]$, which is (an unknown) parameter in the above inequality that controls by what fraction the utility of the feedback ranking $\bar{\mathbf{y}}_t$ is higher than that of the predicted ranking $\mathbf{y}_t$ as compared to the maximum possible utility gain. To allow noisy feedback, we introduce slack variables ξ_t which allow violations of the above condition. This gives the following user feedback, referred to as **α-informative feedback**:

$$U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t) = \alpha \left(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t) \right) - \xi_t. \quad (3)$$

The above feedback model can be further relaxed, requiring that it merely holds in expectation over feedback. We refer to this as **expected α-informative feedback**

$$\mathbf{E}[U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t)] = \alpha \left(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t) \right) - \bar{\xi}_t, \quad (4)$$

and many of our results can be generalized to his weaker form of feedback as well. Note that the above expectation is over user's choice of $\bar{\mathbf{y}}_t$ given $\mathbf{y}_t$ for corpus $\mathbf{x}_t$ (i.e., distribution $\mathbf{P}_{\mathbf{x}_t}[\bar{\mathbf{y}}_t | \mathbf{y}_t]$). Moreover, $\bar{\xi}_t$ denotes the corresponding slack variable.

To measure the performance of our method we define a notion of **regret** based on the utility of the ranking we present with respect to the utility of the best possible ranking $\mathbf{y}_t^*$ that could have been presented in each step:

$$REG_T := \frac{1}{T} \sum_{t=0}^{T-1} \left(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t) \right). \quad (5)$$

Note that regret is measured with respect to the user's true utility function $U(\mathbf{x}_t, \mathbf{y}_t)$ and the optimal ranking $\mathbf{y}_t^*$, even though neither is ever explicitly revealed to the algorithm. Thus a decreasing regret indicates the utility of the predicted ranking improves over time.

4. MODELING RELEVANCE AND DIVERSITY

A key step in designing a learning algorithm that models both relevance and diversity lies in the design of an appropriate hypothesis space for modeling $U(\mathbf{x}, \mathbf{y})$. In short, the learning algorithm needs to learn an accurate model of how the user values a ranking $\mathbf{y}$ for a given $\mathbf{x}$. Since this relates to metrics for evaluating retrieval systems, we start our design of $U(\mathbf{x}, \mathbf{y})$ based on existing retrieval measures.

While traditional IR metrics are oblivious to diversity (e.g. NDCG, Precision), more recent additions account for diversity in some form (e.g. [16, 11, 18, 1, 5]). We define our hypothesis space based on the family of performance measures proposed in [12], since it subsumes many existing measures. These measures exhibit a *diminishing returns* property (i.e. submodularity), which means that the marginal utility of a document is lower if the intents the document is relevant to are already represented in the ranking.

While [12] focuses on the case of extrinsic diversity, the same model structure also applies to problems with need for intrinsic diversity. In particular, we model $U(\mathbf{x}, \mathbf{y})$ as a function that is linear in its parameters $\mathbf{w} \in \mathbf{R}^m$ with $\mathbf{w} \geq 0$,[2] but submodular (and non-linear) in a feature map $\phi(\mathbf{x}, \mathbf{y}) \in \mathbf{R}^m$ with $\phi(\mathbf{x}, \mathbf{y}) \geq 0$:

$$U(\mathbf{x}, \mathbf{y}) := \mathbf{w}^\top \phi(\mathbf{x}, \mathbf{y}). \quad (6)$$

The parameters $\mathbf{w}$ will be learned by the learning algorithm. The feature vector $\phi(\mathbf{x}, \mathbf{y})$ describes the ranking, but for simplicity of exposition we will consider $\mathbf{y}$ to be the set consisting of the top k results that were viewed by the user, not the full ranking[3]. The function $\phi(\mathbf{x}, \mathbf{y})$ generates a feature vector describing the set $\mathbf{y} = \{d_{i_1}, d_{i_2}, ..., d_{i_k}\}$ under context $\mathbf{x} = \{d_1, d_2, ..., d_{|\mathbf{x}|}\}$ in the following manner: We assume that each document d itself is described by a feature vector $\phi(d)$. These feature vectors are aggregated into the feature vector $\phi(\mathbf{x}, \mathbf{y})$ of $\mathbf{y}$ using an aggregation function F. Let $\phi^j(\mathbf{x}, \mathbf{y})$ be the j-th feature of $\phi(\mathbf{x}, \mathbf{y})$ and $\phi^j(d)$ the j-th feature of $\phi(d)$, then

$$\phi^j(\mathbf{x}, \mathbf{y}) = F(\{\phi^j(d_{i_1}), \phi^j(d_{i_2}), ..., \phi^j(d_{i_k})\}). \quad (7)$$

Examples of the per-feature aggregation function F are:

Name	$F(A)$	Subsumes
LIN	$F(A) = \sum_{a \in A} a$	Precision, DCG
MAX	$F(A) = \max_{a \in A} a$	Coverage

The MAX variant, but not LIN, encourages diversity in the following way. As example, consider a boolean bag-of-words representation of documents $\phi(d)$. The first document to contain a term t will increase the feature value of t in $\phi(\mathbf{x}, \mathbf{y})$ by 1. The second document to contain t, however, will not cause any increase. This models the redundancy of multiple occurrences of t, as it does not give any benefit to all but the first occurrence of t. Note that multiple aggregation functions F can be stacked into $\phi(\mathbf{x}, \mathbf{y})$, which allows the linear model to select a desired diminishing-returns profile. Note also that our model is not restricted to the F listed above, but rather any F can be used that is monotone and submodular [12], including less stringent functions which allows for some redundancy (like square root).

To compute the ranking that maximizes a utility function, i.e. $\mathbf{y} := \arg\max_{\mathbf{y} \in \mathcal{Y}} \mathbf{w}^\top \phi(\mathbf{x}, \mathbf{y})$, one can use the simple and efficient Greedy Algorithm 1. At each step, the algorithm greedily chooses the document with the highest marginal utility to be added to the ranking. Note that $\mathbf{y} \oplus d$ is used to refer to the operator that appends document d to ranking $\mathbf{y}$. Also note that Algorithm 1 computes the exact utility optimizer $\mathbf{y}_t$ for the modular measure LIN, whereas it finds a $1 - 1/e$ approximate $\mathbf{y}_t$ for any submodular and monotone function F.

[2]Denotes component-wise non-negativity.

[3]A ranking can be viewed as a nested structure of top-k sets, and the greedy algorithm we will later use to compute rankings uniformly optimizes the utility of the sets at any cutoff in the ranking.

Algorithm 1 GreedyRanking($\mathbf{w}, \mathbf{x}$)

```
y ← 0
for i = 1 to k do
  bestU ← −∞
  for all d ∈ x/ y do
    if w^⊤(x, y ⊕ d) > bestU then
      bestU ← w^⊤ φ(x, y ⊕ d)
      best ← d
  y ← y ⊕ best
return y
```

5. ONLINE LEARNING ALGORITHMS

In this section, we present our coactive learning algorithms. In section 5.1, we present a perceptron style algorithm and then a clipped version of it. In section 5.2 we present an exponentiated gradient algorithm. We prove regret bounds for all the proposed algorithms.

5.1 Diversified Perceptron

We now describe our first learning algorithm for minimizing regret (5) for utility functions of the form (6). Algorithm 2, which we call the **Diversifying Perceptron (DP)**, maintains a weight vector $\mathbf{w}_t$ which is initialized to $\mathbf{0}$. At each time step t, DP presents a ranking $\mathbf{y}_t$ from the corpus $\mathbf{x}_t$ using Algorithm 1 with the current estimate $\mathbf{w}_t$. DP then uses the **user feedback ranking** $\bar{\mathbf{y}}_t$ (obtained as outlined in Section 3) to update the weight vector $\mathbf{w}_t$ in the direction of $\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)$. Note that the α used in modeling user feedback (in Eqns. (2) and (3)) is unknown to the algorithm; it only plays a role in the analysis.

The following theorem describes the generalization performance of the Diversified Perceptron. Note that bound on the worst-case regret is independent of the dimensionality of the feature space, that the regret converges to its asymptote at the rate of $1/\sqrt{T}$ (where T is equal to the number of examples), and that the informativeness α of the feedback enters the bound only linearly. The first term of the bound captures the noise in the feedback.

THEOREM 1. *The average regret of the diversified perceptron algorithm can be upper bounded, for any $\mathbf{w} \in \mathbf{R}^m$ with $\mathbf{w} \geq 0$ that defines the utility in Eq. (6), as follows:*

$$REG_T \leq \frac{1}{\alpha T} \sum_{t=0}^{T-1} \xi_t + \frac{\beta R \|\mathbf{w}\|}{\alpha} + \frac{\sqrt{2}\sqrt{4 - \beta^2} R \|\mathbf{w}\|}{\alpha \sqrt{T}}. \quad (8)$$

Here $\frac{1}{\beta+1}$ is the approximation factor of the greedy algorithm with $\beta \leq 2$ and $\|\phi(\mathbf{x}, \mathbf{y})\|_{\ell_2} \leq R$.

PROOF. Consider the ℓ_2 norm of $\mathbf{w}_T$:

$$\begin{aligned} \|\mathbf{w}_T\|^2 &= \|\mathbf{w}_{T-1}\|^2 + 2\mathbf{w}_{T-1}^\top(\phi(\mathbf{x}_{T-1}, \bar{\mathbf{y}}_{T-1}) - \phi(\mathbf{x}_{T-1}, \mathbf{y}_{T-1})) \\ &\quad + \|\phi(\mathbf{x}_{T-1}, \bar{\mathbf{y}}_{T-1}) - \phi(\mathbf{x}_{T-1}, \mathbf{y}_{T-1})\|^2 \\ &\leq \|\mathbf{w}_{T-1}\|^2 + 2\beta\ \mathbf{w}_{T-1}^\top \phi(\mathbf{x}_{T-1}, \mathbf{y}_{T-1}) + 4R^2 \\ &\leq \|\mathbf{w}_{T-1}\|^2 + 2\beta \|\mathbf{w}_{T-1}\| R + 4R^2 \quad (9) \end{aligned}$$

The first line comes from the update rule in Algorithm 2. The second line is from the fact: $\mathbf{w}_{T-1}^\top \phi(\mathbf{x}_{T-1}, \bar{\mathbf{y}}_{T-1}) \leq (\beta +$

Algorithm 2 Diversifying Perceptron.

Initialize $\mathbf{w}_0 \leftarrow \mathbf{0}$
for $t = 0$ **to** $T-1$ **do**
  Observe $\mathbf{x}_t$
  Present $\mathbf{y}_t \leftarrow GreedyRanking(\mathbf{w}_t, \mathbf{x}_t)$
  Obtain feedback $\bar{\mathbf{y}}_t$
  Update: $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)$

$1)\mathbf{w}_{T-1}^\top \phi(\mathbf{x}_{T-1}, \mathbf{y}_{T-1})$ since the greedy algorithm produces an $\frac{1}{\beta+1}$ approximation and that $\|\phi(\cdot,\cdot)\| \leq R$. The third line comes by using the Cauchy-Schwarz inequality.

Let us inductively assume that $\|\mathbf{w}_t\| \leq c_1 R(t+c_2)$ for $t = \{0, ...T-1\}$ where the values $c_1, c_2 \geq 0$ will be determined later. The base case is trivially shown as $\|\mathbf{w}_0\| = 0$. Thus to complete the induction step, we have:

$$\begin{aligned}
\|\mathbf{w}_T\|^2 &\leq \|\mathbf{w}_{T-1}\|^2 + 2\beta\|\mathbf{w}_{T-1}\|R + 4R^2 \\
&\leq \|\mathbf{w}_{T-2}\|^2 + 2\beta R(\|\mathbf{w}_{T-1}\| + \|\mathbf{w}_{T-2}\|) + 8R^2 \\
&\leq \|\mathbf{w}_0\|^2 + 2\beta R \sum_{t=0}^{T-1} \|\mathbf{w}_t\| + 4R^2 T \\
&\leq \beta R^2 c_1 (T^2 - T) + 2\beta R^2 T c_1 c_2 + 4R^2 T \\
&\leq R^2 \Big(\beta c_1 T^2 + T(-\beta c_1 + 2\beta c_1 c_2 + 4) \Big)
\end{aligned}$$

We now choose c_1 and c_2 such that the induction step holds. This is done by ensuring that the coefficients of T^2 and T in the above expression are smaller than the corresponding terms in $c_1^2 T^2 + 2c_1^2 c_2 T + c_1^2 c_2^2$. First, set $c_1 = \beta + \epsilon$, which will ensure the inequality for T^2. Next, we can ensure $-\beta c_1 + 2\beta c_1 c_2 + 4 \leq 2c_1^2 c_2$, by setting $c_2 = \frac{4-\beta(\beta+\epsilon)}{2\epsilon(\epsilon+\beta)}$. We therefore have $\|\mathbf{w}_T\| \leq (\epsilon+\beta)TR + \frac{(4-\beta^2)R}{2\epsilon} - \frac{\beta R}{2}$. Minimizing the above bound over ϵ, we get $\epsilon = \sqrt{\frac{4-\beta^2}{2T}}$. Substituting this in the upper bound for $\|\mathbf{w}_T\|$, we get $\|\mathbf{w}_T\| \leq (\beta T + \sqrt{4-\beta^2}\sqrt{2T})R$.

Thus using the update rule of Algorithm 2, we have,

$$\begin{aligned}
\mathbf{w}_T^\top \mathbf{w} &= \mathbf{w}_{T-1}^\top \mathbf{w} + U(\mathbf{x}_{T-1}, \bar{\mathbf{y}}_{T-1}) - U(\mathbf{x}_{T-1}, \mathbf{y}_{T-1}) \\
&= \sum_{t=0}^{T-1} U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t).
\end{aligned}$$

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

$$\sum_{t=0}^{T-1} U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t) \leq (\beta T + \sqrt{4-\beta^2}\sqrt{2T})R\|\mathbf{w}\|.$$

The above inequality, along with the condition of *α-informative feedback gives:*

$$\alpha REG_T - \frac{1}{T}\sum_{t=0}^{T-1} \xi_t \leq \left(\beta + \sqrt{4-\beta^2}\sqrt{\frac{2}{T}}\right) R\|\mathbf{w}\|$$

from which the claimed result follows. □

For the case of modular utility (LIN), $\beta = 0$ and the above bound resembles the bound in [14]. For submodular utilities, $\beta = 1/(e+1)$ in the worst case, but is typically much smaller in practice. When users provide "clean" feedback according to (2), the first term in the bound (8) vanishes. We can also show a result similar to the one above in the case of expected

Algorithm 3 Clipped Diversifying Perceptron.

Initialize $\mathbf{w}_0 \leftarrow \mathbf{0}$
for $t = 0$ **to** $T-1$ **do**
  Observe $\mathbf{x}_t$
  Present $\mathbf{y}_t \leftarrow GreedyRanking(\mathbf{w}_t, \mathbf{x}_t)$
  Obtain feedback $\bar{\mathbf{y}}_t$
  Update: $\bar{\mathbf{w}}_{t+1} \leftarrow \mathbf{w}_t + \phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)$
  Clip: $\mathbf{w}_{t+1}^j \leftarrow \max(\bar{\mathbf{w}}_{t+1}^j, 0) \quad \forall 1 \leq j \leq m.$

α-informative feedback (4). We do not provide a proof for this case due to space limitations.

While the above theorem holds whenever there is a $\frac{1}{1+\beta}$-approximation for finding $\mathbf{y}_t$, there is a caveat. In the case of submodular utility, to ensure that the approximation guarantee holds, all the weights in $\mathbf{w}_t$ must be positive. This can be done by an additional clipping step that modifies each weight of $\mathbf{w}_t$ by clipping it at zero if it is negative. The clipped version of the algorithm is shown in Algorithm 3.

For Algorithm 3, assuming that the utility is also defined using a vector $\mathbf{w}$ which has only non-negative components, we can still give a regret bound similar to Theorem 1. Start by observing that, for any t,

$$\|\mathbf{w}_t\|^2 \leq \|\bar{\mathbf{w}}_t\|^2 \quad \text{and} \quad \mathbf{w}^\top \mathbf{w}_t \geq \mathbf{w}^\top \bar{\mathbf{w}}_t \tag{10}$$

The first inequality holds because the product of any clipped value with itself is positive. Since all the components of $\mathbf{w}$ are positive and since only negative values in $\bar{\mathbf{w}}_T$ are set to zero in the clipping step, the second inequality holds. With these two steps, the remaining steps in the proof of Theorem 1 follow and we have the following corollary.

COROLLARY 2. *The average regret of the diversified perceptron algorithm can be upper bounded, for any $\mathbf{w} \in \mathbf{R}^m$ with $\mathbf{w} \geq \mathbf{0}$ that defines the utility, as follows:*

$$REG_T \leq \frac{1}{\alpha T}\sum_{t=0}^{T-1} \xi_t + \frac{\beta R\|\mathbf{w}\|}{\alpha} + \frac{\sqrt{2}\sqrt{4-\beta^2}R\|\mathbf{w}\|}{\alpha\sqrt{T}}, \tag{11}$$

where $\frac{1}{\beta+1}$ is the approximation factor of the greedy algorithm with $\beta \leq 2$ and $\|\phi(\mathbf{x}, \mathbf{y})\| \leq R$.

We obtained the clipped version of the algorithm to avoid non-negative weights. In the next sub-section, we provide an elegant exponentiated algorithm that naturally maintains non-negative weights.

5.2 Exponentiated Algorithm

Our exponentiated algorithm for learning to diversify from implicit feedback is shown in Algorithm 4. In this algorithm, the weights are initialized uniformly at the start. There is a rate θ associated with each step. The rate depends on the maximum ℓ_∞ norm of the feature vectors (i.e., $\|\phi(\cdot,\cdot)\|_{\ell_\infty} \leq S$) and time horizon T.

At each step, a context $\mathbf{x}_t$ is observed and an object $\mathbf{y}_t$ is presented just like in the earlier algorithms. However, once the feedback $\bar{\mathbf{y}}_t$ is obtained, the update rules are multiplicative as shown in Algorithm 4. The weights are normalized 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 positive in this algorithm.

We now prove the regret bound for Algorithm 4. While the regret bounds for Algorithms 2 and 3 depended on the ℓ_2 norm of the features, and the ℓ_2 norm of $\mathbf{w}$, the bound

Algorithm 4 Exponentiated Diversifying Algorithm.

Initialize $\mathbf{w}_0^i \leftarrow \frac{1}{m} \ \ \forall 1 \leq i \leq m$.
$\theta \leftarrow \frac{1}{2S\sqrt{T}}$
for $t = 0$ **to** $T-1$ **do**
  Observe $\mathbf{x}_t$
  Present $\mathbf{y}_t \leftarrow GreedyRanking(\mathbf{w}_t, \mathbf{x}_t)$
  Obtain feedback $\bar{\mathbf{y}}_t$
  Update: $\mathbf{w}_{t+1}^i \leftarrow \mathbf{w}_t^i \exp(\theta(\phi^i(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi^i(\mathbf{x}_t, \mathbf{y}_t)))/Z_t$
  where Z_t is such that the weights add to one.

for the exponentiated algorithm depends on the ℓ_∞ norm of the feature vectors and the ℓ_1 norm of $\mathbf{w}$.

THEOREM 3. *For any $\mathbf{w} \in \mathbf{R}^m$ such that $\|\mathbf{w}\|_{\ell_1} = 1$, $\mathbf{w} \geq 0$, the average regret of the exponentiated algorithm can be upper bounded as follows:*

$$REG_T \leq \frac{1}{\alpha T}\sum_{t=0}^{T-1}\xi_t + \frac{S\beta}{\alpha} + \frac{2\log(m)S}{\alpha\sqrt{T}} + \frac{S}{2\alpha\sqrt{T}}, \quad (12)$$

where $\frac{1}{\beta+1}$ is the approximation factor of the greedy algorithm with $\beta \leq 2$ and $\|\phi(\mathbf{x}, \mathbf{y})\|_{\ell_\infty} \leq S$.

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

$$\begin{aligned} &KL(\mathbf{w}||\mathbf{w}_t) - KL(\mathbf{w}||\mathbf{w}_{t+1}) = \sum_{i=1}^m \mathbf{w}^i \log(\mathbf{w}_{t+1}^i/\mathbf{w}_t^i) \\ &= \sum_{i=1}^m \mathbf{w}^i(\theta(\phi^i(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi^i(\mathbf{x}_t, \mathbf{y}_t))) - \log(Z_t) \\ &= \theta\mathbf{w}^\top(\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)) - \log(Z_t). \quad (13) \end{aligned}$$

On the second line, we pulled out $\log(Z_t)$ from the sum since $\sum_{i=1}^m \mathbf{w}^i = 1$. 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,

$$\begin{aligned} \log(Z_t) &= \log\left(\sum_{i=1}^m \mathbf{w}_t^i \exp(\theta\Delta^i\phi_t)\right) \\ &\leq \log\left(\sum_{i=1}^m \mathbf{w}_t^i(1 + \theta\Delta^i\phi_t + \theta^2{\Delta^i\phi_t}^2)\right) \\ &\leq \log\left(1 + \theta\mathbf{w}_t^\top\Delta\phi_t + \theta^2S^2\right) \\ &\leq \theta\mathbf{w}_t^\top\Delta\phi_t + \theta^2S^2. \quad (14) \end{aligned}$$

On the second line we used the fact that $\exp(x) \leq 1 + x + x^2$ for $x \leq 1$. The rate θ ensures that $\theta(\Delta^i\phi) \leq 1$. On the last line, we used the fact that $\log(1+x) \leq x$. Combing (13) and (14), we get,

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

Adding the above inequalities, we get:

$$\begin{aligned} &\sum_{t=0}^{T-1}(\mathbf{w} - \mathbf{w}_t)^\top(\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)) \\ &\leq \sum_{t=0}^{T-1}\frac{KL(\mathbf{w}||\mathbf{w}_t) - KL(\mathbf{w}||\mathbf{w}_{t+1})}{\theta} + \sum_{t=0}^{T-1}\theta S^2. \\ &\leq \frac{KL(\mathbf{w}||\mathbf{w}_0)}{\theta} + \theta S^2 T. \quad (15) \end{aligned}$$

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

$$\begin{aligned} &\sum_{t=0}^{T-1}(U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t)) \\ &\leq \sum_{t=0}^{T-1}\mathbf{w}_t^\top(\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)) + 2\log(m)S\sqrt{T} + \frac{S\sqrt{T}}{2} \\ &\leq \sum_{t=0}^{T-1}\beta\mathbf{w}_t^\top\phi(\mathbf{x}_t, \mathbf{y}_t) + 2\log(m)S\sqrt{T} + \frac{S\sqrt{T}}{2} \\ &\leq \beta ST + 2\log(m)S\sqrt{T} + \frac{S\sqrt{T}}{2}. \quad (16) \end{aligned}$$

In the above, we also used the fact that $KL(\mathbf{w}||\mathbf{w}_0) \leq \log(m)$ since $\mathbf{w}_0$ is initialized uniformly. On line three, we used the fact that the greedy algorithm finds a $\frac{1}{1+\beta}$ approximation. Moreover, from a generalized version of Cauchy-Schwarz inequality, we obtained

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

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

Like the result in Theorem 1, Theorem 3 also bounds the regret in terms of the noise in the feedback (first term), the approximation factor of the inference algorithm (second 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, and with the ℓ_1-norm of $\mathbf{w}$, which can be advantageous if the optimal $\mathbf{w}$ is sparse.

6. EXPERIMENTS

In this section we empirically study different aspects of our proposed algorithms. In particular, we show how using the submodular utility helps achieve diversity. Furthermore, we explore the robustness of our learning method under degraded feedback quality and noise. We also explore learning the *amount* of diversity a user wants and also compare our method against a supervised method. Finally, we compare the three algorithms that we proposed in this paper against each other.

6.1 Experiment Setup

Since there is no large publicly available real-world corpus containing intrinsic diversity judgments[4], we created two artificial datasets from the RCV-1 [9] text corpus and from the 20 newsgroups dataset (abbreviated 20NG).

The RCV-1 corpus contains over 800k documents, each of which is annotated as belonging to one or more of 100+ *topics*. While the original RCV-1 topics are arranged hierarchically, to make the problem non-trivial, we considered only topics from the second level. The 20NG dataset contains about 19k documents (with duplicates removed) with a single class label for each document. We simulate users with multiple different interests, by forming *super-users* with 5 different interests corresponding to 5 different topics/classes. Thus, if a document is relevant to any of these topics it is relevant to that super-user, else it is not. We assume that

[4]Corpora like the TREC WEB corpus are small and contain relevance judgments only for extrinsic diversity.

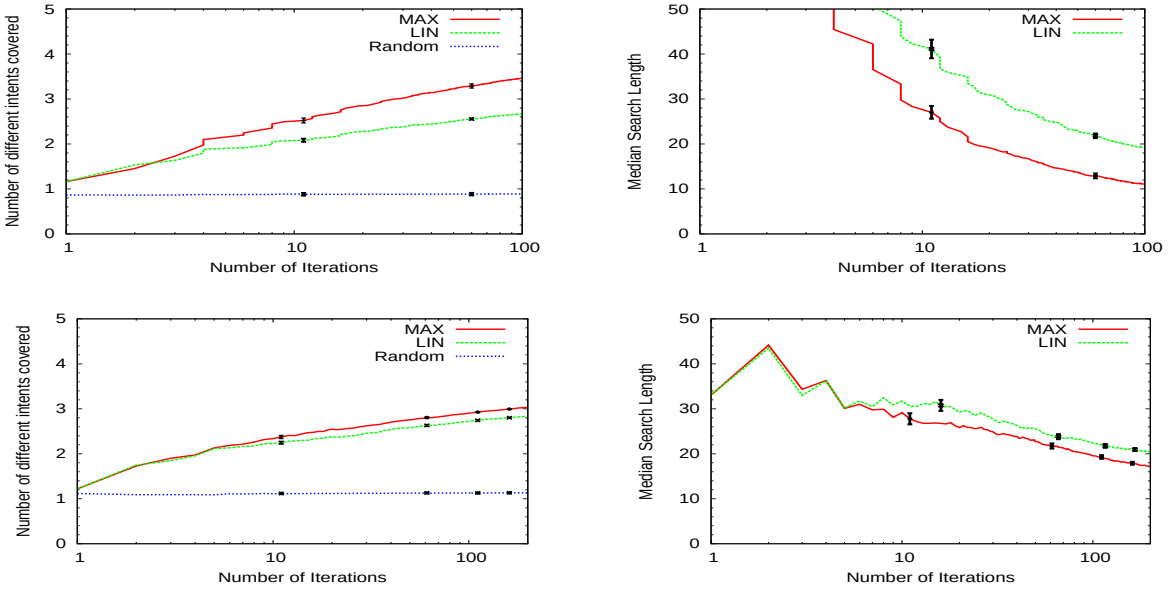

Figure 1: Comparison between the submodular (MAX) and independent (LIN) model for users that are purely seeking diversity; top: RCV-1, bottom: 20NG.

all topics are equally important unless otherwise mentioned. In addition, for a given *super-user* we removed documents relevant to multiple interests. In this manner, producing a diverse set of results would require being able to truly learn each of the interests separately.

We ran the Diversifying Perceptron algorithm with a fresh set of 1000 documents for RCV1 (100 for *20NG*) in each step as the corpus $\mathbf{x}$ and presented a ranking $\mathbf{y}$ from the current corpus. In particular we focus on the top 5 results for all evaluation measures for brevity, though the trends reported in the following hold true for other ranking lengths as well. All results we report are averaged over 50 runs of the algorithm, each for a different *super-user*. Documents are represented as TFIDF vectors. The joint feature map $\phi(\mathbf{x}, \mathbf{y})$ is an aggregation of the document vectors using one (or multiple) of the aggregation functions F described in Section 4.

6.2 Can the algorithm learn to diversify?

We first evaluate if the proposed DP algorithm is really able to learn a function that combines relevance and diversity. In particular, we generated users with 5 different and disjoint interests, and each user wants to read exactly one document relevant to each interest in every iteration. Note that users of this type are seeking maximum diversity in their rankings. To illustrate the performance of the algorithm, we report two quantities. First, we computed how many interests are covered in the top 5 documents of the presented ranking in each iteration. Second, we considered the *median* depth the user needs to search down the ranking to find one document for each of his interests.

We ran the DP algorithm with the **MAX** feature map as defined in Section 4. This is compared against another instance of our algorithm that uses the conventional model **LIN**, which focuses purely on relevance but cannot model diversity directly. For simplicity, we assume $\alpha = 1$ informative feedback. We also compare against a **Random** baseline, which is the performance of a random ranking.

Figure 1 shows the average and standard error of the results for this experiment on the two datasets. The left column shows the number of intents covered in the top 5 positions over time. While the LIN method is far better than the Random method and continues to improve over time, it is outperformed by the MAX method, which is able to learn better.

The right pane further illustrates this result, as it shows how the median search length (required to find at least one document for each intent) starts at high values, but quickly drops to small values after a few iterations. Both learning methods clearly outperform the Random baseline, the value of which is too large to show. In all the plots, the standard errors are small implying statistical significance.

It can be observed that the difference between the MAX and the LIN is much higher in the case of RCV-1 compared to 20NG dataset. This is due to the fact that 20NG has only 20 categories, whereas RCV-1 has more than 100 and is thus much harder to learn for LIN.

6.3 What is the effect of feedback quality?

We next study the effect of the quality of feedback (as described by α) on the performance our method. As real-world users are unlikely to provide perfect feedback, we would like our algorithm to learn even in scenarios where the user-feedback is far from ideal. We varied the quality of the feedback by changing the value of α. A change in α is achieved through the following mechanism: for any intent not covered in the presented ranking, but covered in the optimal ranking, with probability $1 - \alpha$, documents covering that intent are absent in the feedback ranking. This leads to having α-informative feedback in expectation.

Figure 2 shows the results for this experiment. Most notably, the performance is nearly unchanged for larger values of α. In particular, we find that for $\alpha \geq 0.6$ the perfor-

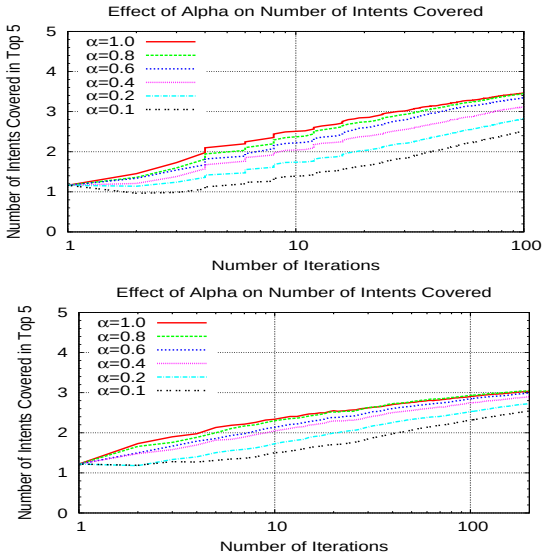

Figure 2: Effect of α on performance of the algorithm for users that are purely seeking diversity; top: RCV-1, bottom: 20NG.

mance is very close to that with perfect feedback ($\alpha = 1.0$). At low values of α such as 0.2 or 0.1, the method still makes reasonable progress over time, albeit at a slower rate. We see that for $\alpha = 0.2$ within 100 iterations the number of intents covered more than doubles. These results indicate that the proposed method is still able to learn even when the informativeness of the user feedback is poor.

6.4 What is the robustness to noise?

While the experiments in the previous section showed robustness to imperfect feedback, we now test the robustness of our algorithm to noisy feedback. One key difference between the two is that with noisy feedback, the user may return a feedback ranking that is worse than the one he was presented. Such a degradation in the quality of the ranking will be captured by the slack variable seen in Eq. (3). We would particularly like the noise introduced to be reflective of that expected in the real-world, where users may sometimes be unsure of the relevance of some documents. Thus we modify the user clicking mechanism that produces the feedback in the following manner:

- Each irrelevant document encountered in the ranking may be considered as relevant with probability η.
- Documents relevant to one of the user's topics may be confused for a different topic with probability $\eta/5$.

Like α, η affects only the quality of the user feedback and not the learning algorithm itself. Figure 3 shows the effect of varying the noise factor η. As seen in the figure, the algorithm is quite robust to noise. For high values of η, such as 0.2, we find that the algorithm is still able to learn quite well. The figures also indicate the expected α of the feedback received after adding noise. However, note that in this scenario, unlike the experiments varying α, the feedback ranking can be significantly worse than the predicted ranking. Thus we see that for $\eta = 0.2$, although $\alpha \sim 0.4$ in expectation, the performance is noticeably worse than for the case of $\alpha = 0.4$.

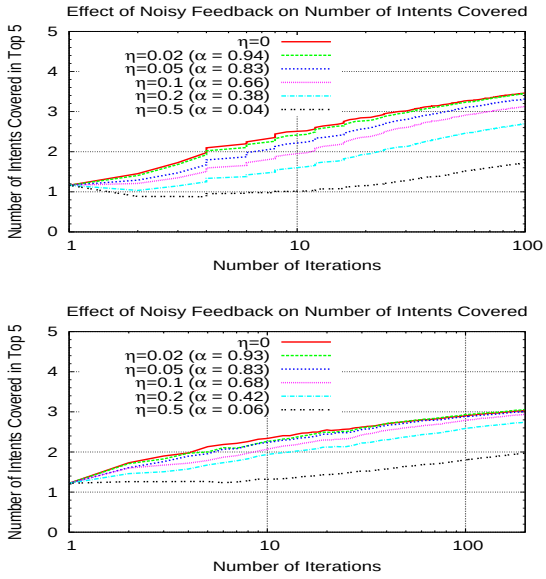

Figure 3: Effect of η on performance of the algorithm for users that are purely seeking diversity (number in bracket indicates the effective α of the feedback); top: RCV-1, bottom: 20NG.

		User-Utility	
		LIN	MAX
	$RANDOM$	.862(±.007)	.756(±.016)
Algo-Util	LIN	.137(±.019)	.447(±.005)
	MAX	.169(±.020)	.274(±.011)
	$LIN + MAX$	.158(±.021)	.310(±.010)

Table 1: Average Regret for different user utilities and algorithm utility functions.

6.5 Learn the desired amount of diversity?

We next explore whether the algorithm can learn how much diversity the user wants. Furthermore, it is interesting to know how the algorithm performs in settings where the utility that the user optimizes (to provide feedback) is different from the one the algorithm uses.

To study this question, we experimented with the MAX and LIN utility functions mentioned earlier. We varied the user's inherent utility as well as the algorithm's utility to either of these two values. We also experimented with a *combination* method for the DP algorithm, which simply takes the joint feature vector representations used in the MAX and LIN functions and appends them to form a single vector. We refer to this method as $MAX + LIN$. To ensure difference in feedback between the two user utility functions, we weight the different intents (as done in [18]), which results in the utility being higher if a more *popular* topic is covered instead of a less popular one. We ran the DP algorithm for 100 iterations, where at each iteration the feedback provided by the user is as per the utility they optimize. We report performance in terms of the the average regret over these 100 iterations of the user's utility measure (since that is what the true $\mathbf{w}$ captures), thus *lower the better*.

Table 1 shows the results for RCV1[5]. First, consider the cases where the algorithm *is given the user's true diversity*

[5]We observe similar results for 20NG but omitted it due to space limitations

profile. As expected, the algorithm performs very well, as seen in the case of the LIN-maximizing algorithm performing best for purely-relevance seeking users (and similarly for the MAX-maximizing algorithm and diversity-seeking users). However, an important result of the experiment is that even when the amount of diversity the user requires is unknown, the *combination* algorithm is able to learn the amount of diversity the user wants. It performs nearly as well as the case where the user's diversity needs are known, as can be seen in the last row of the table. This shows that the combination algorithm is able to learn the trade-off between relevance and diversity that the user is looking for. This is very encouraging as it allows for the method to be used in scenarios where there is no a priori information about the desired amount of diversity. While related to recent work on extrinsic diversity [13], our method is an online learning technique and utilizes much weaker feedback than methods in [13] do.

6.6 Exponentiated algorithm

Compared to the other two algorithms, the exponentiated algorithm has a rate θ associated with it. This rate needs to be set appropriately. In practice, we observed that the performance of the exponentiated algorithm is sensitive to the value of the rate. In particular, we multiplied the rate θ by a numerical value and studied how the algorithm behaved. Note that this effectively changes the radius of the data, but seemed to significantly affect the behavior of the exponentiated algorithm. The results of this experiment is shown in Figure 4. The performance of the algorithm first improves and then deteriorates as the rate factor increases.

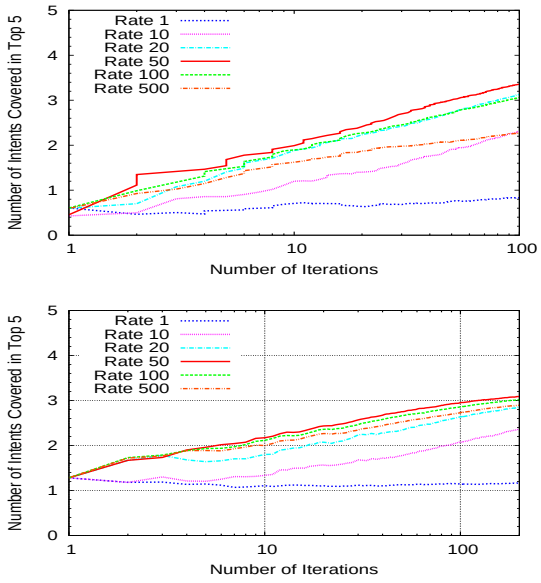

Figure 4: Exponentiated algorithm with different rates; top: RCV-1, bottom: 20NG.

6.7 How do the three algorithms compare?

We proposed three algorithms to learn diversity from implicit feedback. In this section, we study whether there is a difference in performance of these three algorithms. The clipped DP (Algorithm 3) was proposed mainly due to theoretical considerations. To compare the three algorithms, we followed the same setup as in Section 6.2. For the exponentiated algorithm, we considered the best rate parameter from the previous experiment. The results for this experiment are shown in Figure 5. It can be seen that there is not much of a difference between the clipped and the non-clipped algorithms in the case of RCV-1. In the case of 20NG, there is hardly any difference between the three algorithms. Even though restricting weights to positive values is required for theoretical purposes, in practice it does not seem to make much of a difference on these two datasets.

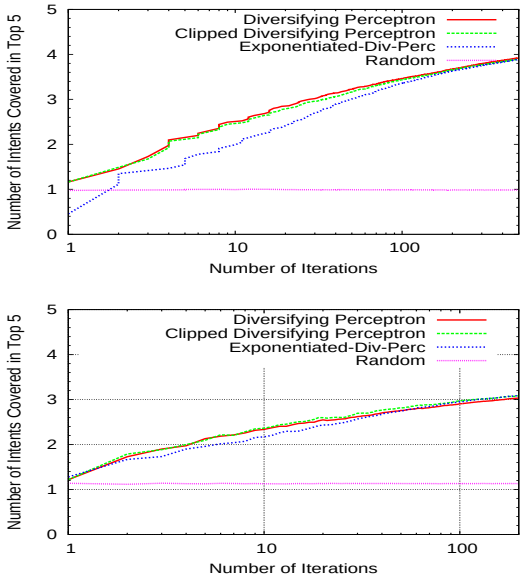

Figure 5: Comparison of the three algorithms; top: RCV-1, bottom: 20NG.

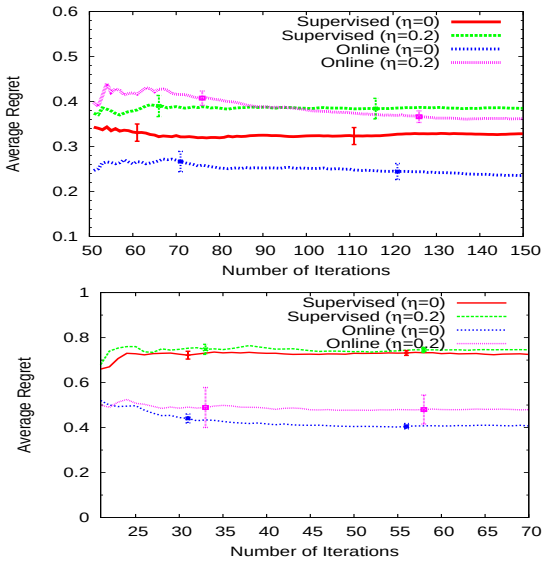

Figure 6: Comparison with supervised learning.

6.8 Comparison to supervised learning

To the best of our knowledge, ours is the first online learning method that can provide a ranking, of required diver-

sity, from a different corpus (i.e. context) in every iteration. Hence there is no suitable online learning baseline to compare against. We thus compare our method against a batch learning method. In particular, we compare against the *one-level* version of the method proposed in [12], which is a generalization of [18].

In this setup, for each maximum diversity-seeking user we obtain the complete document-intent relevance labels for the first 50 (20) iterations for RCV1 (20NG), which is then used in training the SVM-struct based supervised learning method of [12] to obtain the $\mathbf{w}_t$. We train models using the labels from 40 (16) iterations, while utilizing the remaining 10 (4) to select the best value of the C parameter, which is varied from 10^{-2} to 10. The best model is then used to make predictions over the next 100 (50) iterations. We also run the online algorithm over these 150 (70) iterations to compare the two methods. Note that both the online method and the supervised learning method use exactly the same MAX model of user utility and exactly the same document features.

Since the supervised method does not predict rankings for the first 50 (20) iterations, to ensure a fair evaluation, we report the average regret for the next 100 (50) iterations *i.e.*:

$$REG_T := \frac{1}{100} \sum_{t=51}^{150} \left(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t) \right). \qquad (17)$$

We also run both methods with noise introduced using the technique mentioned in subsection 6.4.

As seen in Figure 6, the DP algorithm performs significantly better than the supervised learning method, achieving nearly 25% lower regret when there is no noise for RCV1. This is particularly encouraging given that the amount of feedback the supervised algorithm receives is vastly superior in informativeness to that of the online learning method: While the supervised algorithm receives the relevance labels of each document for each of the user's intent, the DP algorithm only receives a single preference (which has atmost 5 documents) in each iteration. Even for the $\eta = 0.2$ case, the DP algorithm is able to achieve lower regret eventually, indicating that the trend holds even under noisy conditions.

Finally, note that the (per-iteration) training times of the supervised batch method are vastly larger than those of the DP algorithm ($\sim$ 1000s vs. 0.1s). This is because the supervised method solves a more complex optimization problem (the structural SVM objective), while training the Diversifying Perceptron involves just a single update step. Consequently, this makes the DP algorithm especially useful in problem settings where we would like to continuously improve the learned model over time, something that would be prohibitively expensive with the supervised learning method.

7. CONCLUSIONS

We proposed online-learning algorithms for learning diversity in rankings. The proposed algorithms balance diversity and relevance by modeling the utility of the ranking as a submodular function. Using plausible user feedback in the form of preferences between rankings, the algorithms are able to learn rankings that optimize the user's utility. In addition to theoretically characterizing the performance of the algorithms and their robustness to noise, we showed that the algorithms perform well in empirical studies. Future research directions are the deployment of the algorithm in a real system and validation of the feedback model in user studies.

This research was funded in part by NSF Award IIS-0905467.

References

[1] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying search results. In *WSDM*, 2009.

[2] J. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, 1998.

[3] H. Chen and D. R. Karger. Less is more: probabilistic models for retrieving fewer relevant documents. In *SIGIR*, 2006.

[4] C. Clarke, M. Kolla, and O. Vechtomova. An effectiveness measure for ambiguous and underspecified queries. In *Advances in Information Retrieval Theory*, Lecture Notes in Computer Science, 2009.

[5] C. L. Clarke, N. Craswell, and I. Soboroff. Overview of the trec 2009 web track. Technical report, 2010.

[6] K. El-Arini and C. Guestrin. Beyond keyword search: discovering relevant scientific literature. In *KDD*, 2011.

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

[8] A. Kulesza and B. Taskar. Learning determinantal point processes. In *UAI*, pages 419–427, 2011.

[9] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *JMLR*, 5:361–397, 2004.

[10] F. Radlinski, P. N. Bennett, B. Carterette, and T. Joachims. Redundancy, diversity and interdependent document relevance. *SIGIR Forum*, 43(2):46–52, 2009.

[11] F. Radlinski, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. In *ICML*, 2008.

[12] K. Raman, T. Joachims, and P. Shivaswamy. Structured learning of two-level dynamic rankings. In *CIKM*, 2011.

[13] R. L. Santos, C. Macdonald, and I. Ounis. Selectively diversifying web search results. In *CIKM*, 2010.

[14] P. Shivaswamy and T. Joachims. Online structured prediction via coactive learning. In *ICML*, 2012.

[15] A. Slivkins, F. Radlinski, and S. Gollapudi. Learning optimally diverse rankings over large document collections. In *ICML*, pages 983–990, 2010.

[16] A. Swaminthan, C. Metthew, and D. Kirovski. Essential pages. In *Technical Report, MSR-TR-2008-15*, Microsoft Research, 2008.

[17] Y. Yue and C. Guestrin. Linear submodular bandits and their application to diversified retrieval. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[18] Y. Yue and T. Joachims. Predicting diverse subsets using structural svms. In *ICML*, 2008.

[19] C. X. Zhai, W. W. Cohen, and J. Lafferty. Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *SIGIR*, 2003.