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

We propose an online learning model and algorithm for learning rankings that balance relevance and diversity. In each step, the algorithm presents a ranking to the user. As feedback, the algorithm observes the set of documents the user reads in the presented ranking. We propose a simple algorithm exploiting such feedback to maximize any submodular utility measure. Even for imperfect and noisy feedback, we show that the algorithm admits strong theoretical guarantees. In addition to the theoretical results, we find that the algorithm learns quickly, accurately, and robustly in an empirical evaluation.

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 movie recommendation system that wants to recommend at least one movie the user wants to watch on a given day. The system is well-advised to present a diverse set of movies, since diversity hedges against uncertainty about the users mood on that day. Such hedging against uncertainty about the user's information need is called *extrinsic diversity* [8]. A second reason for diversity is called *intrinsic diversity* [8] 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 this paper, we focus on problems where such intrinsic diversity is important.

While much prior work on diversity has focused on non-learning approaches (e.g. [2, 16, 3, 14, 4]), recently developed supervised learning methods for diversity have shown a lot of promise (e.g. [15, 10, 6]). Unfortunately, supervised learning relies on manually judged training data with multi-topic annotations, which are expensive and difficult to obtain. While some online learning methods exist that can exploit click data, those methods either cannot generalize across queries [9] and/or have a hard-coded notion of diversity that cannot be adjusted through learning [13].

We overcome these problems by extending a recently proposed online learning model [12] for learning from implicit user feedback. In particular, we develop an algorithm for learning both relevance and the desired amount of diversity from set-valued preference data that can be derived from implicit feedback. The algorithm proposed in this paper is extremely easy to understand and implement. Furthermore, the ability to learn the desired amount of diversity based on user feedback makes the algorithm attractive for a wide range of applications where the required amount of diversity is not determined apriori. A crucial extension over the methods in [12] is that we now consider models with submodular structure. Their diminishing returns property makes it possible to avoid redundancy and increase novelty.

The learning process 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 models in an online fashion. 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 [12], our guarantees apply even though submodular models allow only approximate inference. Finally, experiments show the empirical effectiveness of the proposed approach in learning both relevance and diversity.

2. 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 such interdependencies. The training data we exploit are the sets of documents read on 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. $\bar{\mathbf{y}}_t$ is referred to 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

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

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 nevertheless ensure meaningful learning under this weak form of feedback, we make the following assumption on user feedback:

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

It 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. In particular, $\alpha \in (0, 1]$ 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 $\xi_t \geq 0$ which allow violations of the above condition. This gives the following user feedback, referred to as **α-informative feedback**:

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

The above feedback model can be further relaxed, requiring that it merely holds in expectation over feedback. This gives,

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

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:

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

Note that regret is measured with respect to the user's true utility function $U(\mathbf{x}_t, \mathbf{y}_t)$, even though this function is never explicitly revealed to the algorithm.

3. 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. [14, 9, 15, 1, 5]). We define our hypothesis space based on the family of performance measures proposed in [10], since it subsumed many existing measures. These measure 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 [10] 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 $\mathbf{w}$, but submodular (and non-linear) in a feature map $\phi(\mathbf{x}, \mathbf{y})$.

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

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[2]. 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

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

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

[2]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($\mathbf{w}, \mathbf{x}$)

```
$\mathbf{y} \leftarrow 0$
for $i = 1$ to $k$ do
  $bestU \leftarrow -\infty$
  for all $d \in \mathbf{x}/\ \mathbf{y}$ do
    if $\mathbf{w}^\top(\mathbf{x}, \mathbf{y} \oplus d) > bestU$ then
      $bestU \leftarrow \mathbf{w}^\top \phi(\mathbf{x}, \mathbf{y} \oplus d)$
      $best \leftarrow d$
  $\mathbf{y} \leftarrow \mathbf{y} \oplus best$
return $\mathbf{y}$
```

Examples of the per-feature aggregation function F are the following:

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

The variants SQRT and MAX, but not LIN, encourage diversity in the following way. As example, consider a boolean bag-of-words representation of documents $\phi(d)$ and the SQRT aggregation. The first document to contain a term t will increase the feature value of t in $\phi(\mathbf{x}, \mathbf{y})$ by $\sqrt{1} = 1$. The second document to contain t, however, will only lead to a diminished increase of $\sqrt{2} - \sqrt{1} = 0.41$, and a third one to an even smaller one (i.e. $\sqrt{3} - \sqrt{2} = 0.32$). This models the partial redundance of multiple occurrences of t. The most extreme is MAX, which 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 in $\mathbf{y}$ [10].

To compute the ranking that maximizes a utility function, i.e. $\mathbf{y} := \arg\max_{\mathbf{y} \in \mathcal{Y}} \left[\mathbf{w}^\top \phi(\mathbf{x}, \mathbf{y})\right]$, 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 measure (e.g. SQRT, MAX) [10].

4. ONLINE LEARNING ALGORITHM

We now present the learning algorithm for minimizing regret (5). 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 2) to update the weight vector $\mathbf{w}_t$ in the direction of $\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)$.

THEOREM 1. *For feedback that is α-informative, the average regret of the diversified perceptron algorithm can be upper bounded as follows:*

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

Algorithm 2 Diversifying Perceptron.

```
Initialize $\mathbf{w}_1 \leftarrow \mathbf{0}$
for $t = 1$ to $T$ do
  Observe $\mathbf{x}_t$
  Present $\mathbf{y}_t \leftarrow GreedyRanking(\mathbf{w}_t, \mathbf{x}_t)$
  Obtain feedback $\bar{\mathbf{y}}_t$
  Update: $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)$
```

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

PROOF. Consider the norm of $\mathbf{w}_{T+1}$:

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

The first line comes from the update rule in Algorithm 2. The second line is from the fact that $\mathbf{w}_T^\top\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) \le (\beta + 1)\phi_t(\mathbf{x}_T, \mathbf{y}_T)$ since the greedy algorithm produces an $\frac{1}{\beta+1}$ approximation and that $\|\phi(\cdot,\cdot)\| \le R$. The third line comes by using the Cauchy-Schwarz inequality. Fourth line is by using the fact that $\beta \le 2$. We obtain the last line inductively. We now prove a stronger bound using the above fact.

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

The last line comes from the earlier result which implies $\|\mathbf{w}_{T+1}\| \le 2RT$ and the fact that $\sum_{k=1}^{T}(k-1) = T(T-1)/2$. Further, from the update rule in algorithm 2, we have,

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

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

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

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

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

from which the claimed result follows. □

For the case of modular utility (LIN), $\beta = 0$ and the above bound reduces to the one in [12]. 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

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

5. EXPERIMENTS

In this section we empirically study different aspects of our proposed algorithm. In particular, we show the benefit of using the submodular utility to 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. Finally, we compare our method against a supervised method.

5.1 Experiment Setup

Since there is no large publicly available corpus containing intrinsic diversity judgments[3] (which is the focus of this work), we created an artificial dataset from the RCV-1 [7] text corpus. This corpus contains over 800k documents each of which is annotated as belonging to one or more of 100+ *topics*. To simulate users with multiple different interests, we formed *super-users* with 5 different interests corresponding to 5 different RCV-1 topics. Thus if a document is relevant to any of these topics they are relevant to that super-user, else they are not. While the original RCV-1 topics are arranged hierarchically, to make the problem non-trivial, we considered only topics from the second level. We assume that 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 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 3.

5.2 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 3. 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 clarity, we assume $\alpha = 1$ informative

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

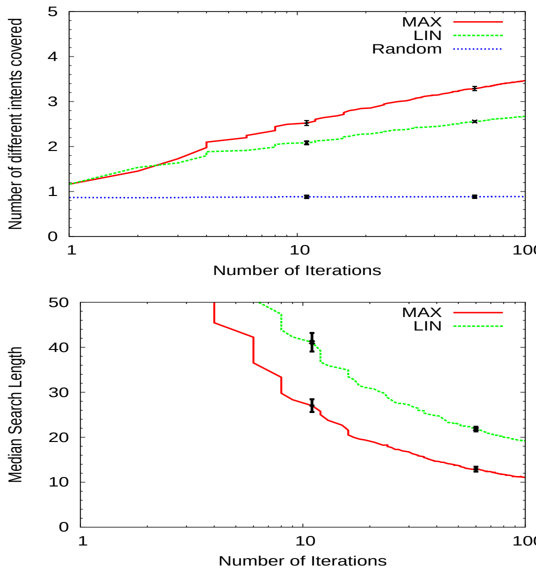

Figure 1: Comparison between the submodular (MAX) and independent (LIN) model for users that are purely seeking diversity.

feedback. We also compared 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. The upper pane 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. In particular we can see how starting from nearly a single intent covered in the top 5, the MAX method covers more than 3 intents on average after about 30 iterations. In comparison LIN still covers less than 3 intents even after 100 iterations. The bottom pane further illustrates this effect, as it shows how the median search length (required to find at least one document for each intent) starts from more than 100, but quickly drops to less than 20 after 20 iterations. LIN needs about 100 iterations to reach this performance. Both learning methods clearly outperform the Random baseline, the value of which is too large to show. In both plots, the standard errors are quite small implying statistical significance.

5.3 What is the effect of feedback quality (α)?

We next study the effect of the quality of feedback (as governed by α) on the performance our method. As real-life 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. To study this effect, 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 on 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 performance is very close to that with perfect feedback ($\alpha = 1.0$).

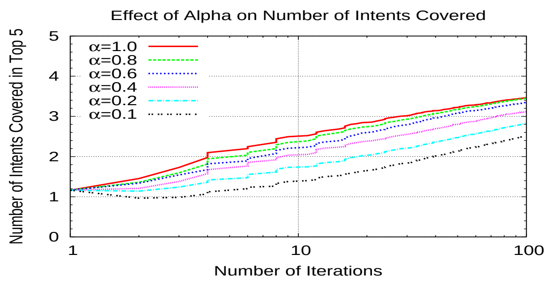

Figure 2: Effect of α on performance of the algorithm for users that are purely seeking diversity.

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.

5.4 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:

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

Figure 3 shows the effect of varying the noise factor η. As seen in the figure, the algorithm is quite robust to this kind of 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 $\alpha = 0.4$.

5.5 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 effect, 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

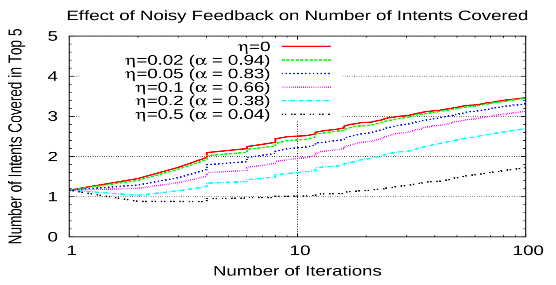

Figure 3: Effect of η on performance of the algorithm for users that are purely seeking diversity (number in bracket indicates the average α).

		User-Utility	
		LIN	MAX
	$RANDOM$	.862(±.007)	.756(±.016)
Algo-Util	LIN	.137(±.019)	.447(±.005)
	MAX	.169(±.02)	.274(±.011)
	$LIN+MAX$	.158(±.021)	.31(±.0095)

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

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 [15]), 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. First, consider the cases where the algorithm *is given the user's true diversity 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 tradeoff 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, similar to recent work on extrinsic diversity [11].

5.6 Comparison to supervised learning

To the best of our knowledge, ours is the first online learning method that can provide a diverse ranking 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 earlier in [10], which is a generalization of [15].

In this setup, for each maximum diversity-seeking user we obtain the complete document-intent relevance labels for the

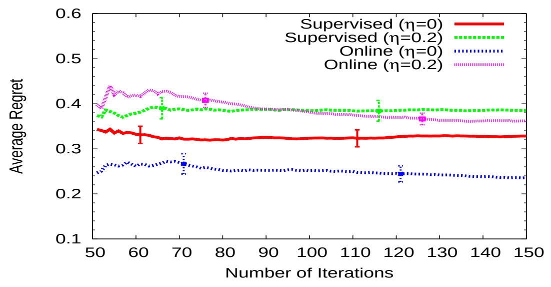

Figure 4: Comparison with supervised learning.

first 50 iterations, which is then used in training the SVM-struct based supervised learning method of [10] to obtain the $\mathbf{w}_t$. We train model using the labels from 40 iterations, while utilizing the remaining 10 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 iterations. We also run the online algorithm over these 150 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 iterations, to ensure a fair evaluation, we report the average regret for the next 100 iterations *i.e.* :

$$REG_T := \frac{1}{100} \sum_{t=51}^{150} \left(U(\mathbf{x}_t, \mathbf{y}_t^*) - U(\mathbf{x}_t, \mathbf{y}_t) \right). \tag{9}$$

We also run both methods with noise introduced using the technique mentioned in subsection 5.4.

As seen in Figure 4, the DP algorithm performs significantly better than the supervised learning method, achieving nearly 25% lower regret when there is no noise. 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.

6. CONCLUSIONS

We proposed an online-learning algorithm for learning diversity in rankings. The proposed DP algorithm balances 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 algorithm is able learn rankings that optimize the user's utility. In addition to theoretically characterizing the performance of the algorithm and its robustness to noise, we showed that the algorithm performs well in empirical studies. Future directions for research are the deployment of the algorithm in a real system and the validation of the feedback model in user studies.

7. ACKNOWLEDGMENT

This work was funded in part by NSF Awards IIS-0905467 and IIS-1142251.

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

[7] 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.

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

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

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

[11] R. L. Santos, C. Macdonald, and I. Ounis. Selectively diversifying web search results. In *CIKM*, pages 1179–1188, 2010.

[12] P. Shivaswamy and T. Joachims. Online learning with preference feedback. In *NIPS workshop on Choice Models and Preference Learning*, 2011.

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

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

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

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