Scalable Exploration for Neural Online Learning to Rank with Perturbed Feedback

Yiling Jia  
Department of Computer Science  
University of Virginia  
Charlottesville, VA 22903, USA  
yj9xs@virginia.edu

Hongning Wang  
Department of Computer Science  
University of Virginia  
Charlottesville, VA 22903, USA  
hw5x@virginia.edu

Abstract

Deep neural networks (DNNs) demonstrate significant advantages in improving ranking performance in retrieval tasks. Driven by the recent technical developments in optimization and generalization of DNNs, learning a neural ranking model online from its interactions with users becomes possible. However, the required exploration for model learning has to be performed in the entire neural network parameter space, which is prohibitively expensive and limits the application of such online solutions in practice.

In this work, we propose an efficient exploration strategy for online interactive neural ranker learning based on the idea of bootstrapping. Our solution employs an ensemble of ranking models trained with perturbed user click feedback. The proposed method eliminates explicit confidence set construction and the associated computational overhead, which enables the online neural rankers’ training to be efficiently executed in practice with theoretical guarantees. Extensive comparisons with an array of state-of-the-art OL2R algorithms on two public learning to rank benchmark datasets demonstrate the effectiveness and computational efficiency of our proposed neural OL2R solution.

Keywords: Online learning to rank; perturbed exploration; regret analysis

1. Introduction

Online learning to rank (OL2R) has recently attracted great research interest because of its unique advantages in capturing users’ ranking preferences without requiring expensive relevance labeling as in classical offline learning to rank solutions (Yue and Joachims, 2009; Schuth et al., 2016; Wang et al., 2019; Jia et al., 2021; Lattimore et al., 2018; Li et al., 2018; Kveton et al., 2018). Because users’ implicit feedback is noisy and biased (Joachims et al., 2005; Agichtein et al., 2006; Joachims et al., 2007; Chapelle et al., 2012), the key in OL2R is to effectively explore the unknowns for improved relevance estimation, while serving the users with high-quality ranked results, which is known as the explore-exploit trade-off.

Most existing work in OL2R assumes a linear scoring function (Yue and Joachims, 2009; Schuth et al., 2016; Wang et al., 2019). Dueling bandit gradient descent (DBGD) and its different variants are the most popularly used OL2R solutions (Yue and Joachims, 2009), where new model variants are sampled via random perturbations in the parameter space to estimate the direction for model update. As its non-linear extension, pairwise differential gradient descent (PDGD) (Oosterhuis and de Rijke, 2018) samples the next ranked document from a Plackett-Luce model and estimates an unbiased gradient from the inferred pairwise preference. PairRank (Jia et al., 2021) learns a logistic ranker online in a pairwise manner and explores the ranking space based on the model’s estimation uncertainty about the pairwise comparisons of document rankings. Though practically effective, such a linear or generalized linear model assumption is incompetent to capture the possible complex non-linear relations between a document’s ranking features and its relevance quality. This is already proved to be crucial in the past offline learning to rank practices (Liu, 2011; Burges, 2010).
To unleash the power of representation learning, deep neural networks (DNN) have been introduced to learn the underlying scoring function for document ranking. In (Oosterhuis and de Rijke, 2018), PDGD is also experimented on a neural ranker. Though the authors reported promising empirical results, its theoretical property (e.g., convergence) is unknown. On the other hand, enabled by the substantial progress in optimization and generalization of DNNs, quantifying a neural model’s uncertainty on new data points become possible (Cao and Gu, 2020, 2019; Chen et al., 2019; Daniely, 2017; Arora et al., 2019). A recent work named olRankNet (Jia and Wang, 2022) extended PairRank with a neural network ranker, which performs exploration in the pairwise document ranking space with topological sort by using the neural tangent kernel technique (Jacot et al., 2018). Compared with PairRank, olRankNet provided encouraging performance improvement, which was reported to be the best among all state-of-the-art OL2R solutions. More importantly, olRankNet is proved to achieve a sublinear gap-dependent regret upper bound, which is defined on the total number of mis-ordered pairs over the course of interactions with users. To our best knowledge, olRankNet is the first known OL2R solution for neural rankers with theoretical guarantees.

Despite being theoretically sound and empirically effective, olRankNet’s limitation is also remarkably serious: its computational cost for performing the required exploration is prohibitively high (almost cubic to the number of neural network’s parameters). More specifically, to quantify the uncertainty of its estimated pairwise preferences among candidate documents, it has to maintain a high-probability confidence set for the current ranker’s parameter estimation over time. However, the construction of the confidence set depends on the dimensionality of the neural network’s parameters: as required by the neural tangent kernel, an inverse of the covariance matrix computed based on the gradient of the entire neural network is needed whenever the network is updated. For example, for a simple two layer feed-forward neural network with input dimension $d$ and $m$ neurons in each layer, the size of the covariance matrix is $(md + m^2 + m)^2$. The best known time complexity for computing the inverse of this covariance matrix is $O((md + m^2 + m)^{2.373})$, by the optimized Coppersmith–Winograd algorithm (Williams, 2012). This computational complexity quickly outpaces the limit of any modern computational machinery, given $m$ or $d$ are usually very large in practice (e.g., $m$ is often in the hundreds and $d$ in tens of thousands) and such matrix inverse operation is needed in every round when the neural network is updated. Due to this limitation, olRankNet has to employ the diagonal approximation of the covariance matrix in its actual implementations (Jia and Wang, 2022). But such an approximation loses its all theoretical guarantees, which unfortunately leads to a gap between the theoretical and empirical performance of olRankNet. And even how this gap would depend on the dimensionality of the network and affect olRankNet’s performance is completely unknown. This inevitably limits the application of the neural OL2R solutions, especially the olRankNet-type algorithms, in practice.

In this work, we develop an efficient and scalable exploration strategy for olRankNet by eliminating its explicit confidence set construction. The basic idea is to use bootstrapping technique to measure the uncertainty of a neural ranker’s output via a set of sample estimates. In particular, we maintain $N$ rankers in parallel. And in each round, after receiving the user’s click feedback, each of the rankers is updated with the observed clicks and independently generated pseudo noise from a zero-mean Gaussian distribution. The overall model’s estimation uncertainty on a pair of documents is then determined by an ensemble of the estimates from all $N$ rankers. For example, for a document pair $(i, j)$, if all $N$ rankers predict $i \succ j$, $(i, j)$ is considered as in a certain rank order, otherwise it is considered as in an uncertain rank order, where exploration is needed. Besides regular neural network updates, no additional computation is needed, which greatly reduces the computational overhead as required in olRankNet. We name our new solution as Perturbed Pairwise Neural Rank (or P$^2$NeurRank in short). We rigorously prove that with a high probability P$^2$NeurRank obtains the same regret as olRankNet, but the computational complexity is way much lower. In addition, as no approximation is needed in P$^2$NeurRank, its theoretical analysis directly suggests its empirical performance. Our extensive empirical evaluations demonstrate the strong advantage in
both efficiency and effectiveness of $P^2$NeurRank against olRankNet and a rich set of state-of-the-art solutions over two OL2R benchmark datasets on standard retrieval metrics.

2. Related Work

Online learning to rank with neural rankers. Most of the existing parametric OL2R solutions are limited to linear ranking models (Yue and Joachims, 2009; Li et al., 2018). In particular, DBGD and its extensions (Yue and Joachims, 2009; Schuth et al., 2014; Wang et al., 2019, 2018a), as the most popularly referred OL2R solutions, are inherently designed for linear models as they rely on random perturbations in linear model weights for parameter estimation. But such a linear assumption is incompetent to capture any non-linear relations about documents' relevance quality under given queries, which shields such OL2R algorithms away from the successful practices in offline learning to rank models that are empowered by DNNs (Burges, 2010; Pasumarthi et al., 2019).

Such a limitation motivates some preliminary attempts in OL2R. In (Oosterhuis and de Rijke, 2018), pairwise differentiable gradient descent (PDGD) is proposed to sample the next ranked document from a Plackett-Luce model and estimate an unbiased gradient from the inferred pairwise ranking preference. Although improved empirical performance is reported for PDGD with a neural ranker, there is no theoretical guarantee on its online performance. Taking a completely different perspective, PairRank (Jia et al., 2021) directly learns a pairwise logistic regression ranker online and explores the pairwise ranking space via a divide-and-conquer strategy based on the model's uncertainty about the documents' rankings. The authors claimed logistic regression can be treated as a one-layer feed-forward neural network, but it is unclear how PairRank can be extended to more general neural ranking architectures.

Recently, substantial progress in optimization and generalization of DNNs enables theoretical analysis about the neural models (Liang and Srikant, 2016; Telgarsky, 2015, 2016; Yarotsky, 2017, 2018; Lu and Kawaguchi, 2017; Hanin and Sellke, 2017; Zou et al., 2019; Zou and Gu, 2019). For example, with the neural tangent kernel technique (Jacot et al., 2018), the uncertainty of a neural model's estimation on new data points can be quantified (Zhou et al., 2019; Zhang et al., 2020; Jia et al., 2022). Most recently, olRankNet (Jia and Wang, 2022) is proposed to extend PairRank with a multi-layer neural ranker. The authors proved that the good theoretical properties of PairRank (i.e., sublinear regret) are inherited in olRankNet, and empirically improved performance over PairRank was also reported in olRankNet. However, one serious issue of olRankNet is its cumbersome computational complexity: to quantify the confidence interval of the neural ranker's output for the exploration purpose, one has to compute the inverse of a covariance matrix which is derived by the gradient of entire neural network. The complexity is almost cubic to the number of parameters in the neural network, which is prohibitively expensive for OL2R, as this matrix inverse is needed every time the ranker is updated. The authors in (Jia and Wang, 2022) suggested using diagonal approximation for the covariance matrix, but no guarantee is provided about the impact of such an approximation.

Randomized exploration in online learning. Efficient exploration is critical for online algorithms, as the model learns by actively acquiring feedback from the environment (Lattimore and Szepesvári, 2019). Distinct from the deterministic exploration strategies, such as upper confidence bound (Auer, 2002; Abbasi-Yadkori et al., 2011), randomization-based exploration enjoys advantages in its light computational overhead and thus has received increasing attention in online learning community. The most straightforward randomization-based exploration strategy is $\epsilon$-greedy (Auer, 2002), which takes the currently estimated best action with probability $1 - \epsilon$, otherwise randomly take an action. It has been applied in OL2R in (Hofmann et al., 2013). Almost no additional computation is needed in $\epsilon$-greedy for exploration, but the exploration is also independent from the current model estimation and therefore can hardly be optimal in practice. More advanced randomization-based exploration strategies are built on the bootstrapping technique in statistics. Giro (Kveton et al., 2019b) explores by updating a model with a bootstrapped sample of its history with pseudo
reward. In (Kveton et al., 2019a, 2020), random noise is added to the observed feedback for the model training to achieve the purpose of exploration in model’s output. Such a strategy is proved to be effective in both linear and generalized linear model training. Most recently, Jia et al. (Jia et al., 2022) proved randomization can also be used for online neural network learning. The most closely related work to our study is (Ash et al., 2021; Ishfaq et al., 2021), where an ensemble of models are trained to approximate the confidence interval for the purpose of exploration in online model update.

3. Method

In this section, we provide a brief introduction of the general problem setting in OL2R, and then present our proposed solution for scalable exploration in neural OL2R.

3.1 Problem Formulation

In OL2R, a ranker directly learns from the interactions with users for \( T \) rounds. At each round \( t = 1, ..., T \), the ranker receives a query \( q_t \) and its associated \( V_t \) candidate documents represented as a set of \( d \)-dimensional query-document feature vectors, \( x_{t}^{i} \in \mathbb{R}^{d} \), and we assume that \( \|x_{t}^{i}\| \leq u \). Once the query is received, the ranker determines the ranking of the \( V_t \) documents based on its knowledge about the documents’ relevance so far. We denote \( \pi_{t} = (\pi_{t}(1), ..., \pi_{t}(V_t)) \in \Pi([V_t]) \) as the ranking of the \( L_t \) documents, where \( \pi_{t}(i) \) represents the rank position of document \( i \) given query \( q_t \), and \( \Pi([V_t]) \) is the set of all permutations of the \( V_t \) documents. After the ranked list is returned to the user, the user examines the list and provides her click feedback \( C_t = \{c_{t1}, c_{t2}, ..., c_{tV_t}\} \), where \( c_{ti} = 1 \) if the user clicks on document \( i \) at round \( t \); otherwise \( c_{ti} = 0 \). The ranker updates itself according to the feedback and proceeds to the next query.

Existing studies have repeatedly demonstrated that \( C_t \) is biased and noisy (Joachims et al., 2005; Agichtein et al., 2006; Joachims et al., 2007; Chapelle et al., 2012). Users tend to click more on the top-ranked documents, which is known as the position bias; and as users can only interact with the documents shown to them, the ranker only has partial observations about relevance feedback from user clicks, which is known as the presentation bias. Therefore, a good OL2R solution needs to carefully deal with the biased implicit feedback and effectively explore the unknowns for improved relevance estimation on the one hand, and serve users with the currently best estimated ranked result on the other hand.

In this work, we follow the standard practice and treat clicks as relative preference feedback (Joachims et al., 2005). More specifically, the clicked documents are assumed to be preferred over those examined but unclicked documents. Besides, we consider every document that precedes a clicked document and the first \( l \) subsequent unclicked document as examined. Such an assumption is widely adopted and proved to be effective in both offline and online learning to rank (Wang et al., 2019; Agichtein et al., 2006; Oosterhuis and de Rijke, 2018; Jia et al., 2021). In particular, we denote \( o_t \) as the index of the last examined position in the ranked list \( \pi_t \) at round \( t \).

Different from offline learning to rank, OL2R needs to serve the users while learning from its presented rankings. Therefore cumulative regret is an important metric for evaluating OL2R. In this work, we follow the regret defined as the number of mis-ordered pairs from the presented ranking to the ideal one (Jia et al., 2021; Lattimore et al., 2018; Jia and Wang, 2022),

\[
R_T = \mathbb{E}\left[\sum_{t=1}^{T} r_t\right] = \mathbb{E}\left[\sum_{t=1}^{T} K(\pi_t, \pi_t^*)\right]
\]

where \( \pi_t \) is the ranked list generated by the current ranker, \( \pi_t^* \) is the optimal ranking for the current query, and \( K(\pi_t, \pi_t^*) = |\{(i, j): i < j, (\pi_t(i) < \pi_t^*(i) \land \pi_t(j) > \pi_t^*(j)) \lor (\pi_t(i) > \pi_t(j) \land \pi_t^*(i) < \pi_t^*(j))\}| \). Such a pairwise regret definition directly connects an OL2R algorithm’s online
performance with classical ranking evaluations as most ranking metrics, such as ARP and NDCG can be decomposed into pairwise comparisons (Wang et al., 2018b).

3.2 Exploration in olRankNet

In the recent decade, DNNs have demonstrated powerful representation learning capacity and significantly boosted the performance for a wide variety of machine learning tasks (Goodfellow et al., 2016; LeCun et al., 2015), including learning to rank (Huang et al., 2013; Dehghani et al., 2017; Guo et al., 2020; Severyn and Moschitti, 2015). OL2R, on the other hand, has received limited benefit from the advances in DNNs. While DNNs are generally more accurate at predicting a document’s relevance under a given query (i.e., exploitation), creating practical strategies to balance exploration and exploitation for neural ranking models in sophisticated online learning scenarios is challenging.

Built on the advances in renewed understandings about the generalization of DNNs, olRankNet extends PairRank with a neural scoring function and demonstrates the best empirical performance and exploitation for neural ranking models in sophisticated online learning scenarios is challenging.

With the constructed confidence interval of the estimated pairwise document rank order, olRankNet decides to randomize its output ranking might be wrong because of this uncertainty. olRankNet determines to balance exploration and exploitation for improved model estimation subsequently. Based on the neural tangent kernel technique, the uncertainty of olRankNet’s estimated pairwise rank order can be analytically quantified and upper bounded with a high probability, under the assumption that pairwise click noise follows a R-sub-distribution (Jia and Wang, 2022). This is described in the following lemma.

Lemma 3.1. (Confidence Interval of Pairwise Rank Order in olRankNet). There exist positive constants $C_1$ and $C_2$ such that for any $\delta_1 \in (0, 1)$, with satisfied step size of gradient descent $\eta_t$ and the network width $m$, at round $t < T$, for any document pair $(i, j)$ under query $q_t$, with probability at least $1 - \delta_1$,

$$|\sigma(f_{ij}^t) - \sigma(f_{ij}^s)| \leq \alpha_t m^{-1/2} + \epsilon(m),$$

where $\epsilon(m)$ is the approximation error from gradient descent in neural network optimization, $f_{ij}^s$ is the ground-truth difference between document pair $(i, j)$, $\sigma(g(x; \theta) - g(x; \theta))$ as the gradient of input $x$ with respect to the entire network parameters, $\alpha_t = C_1 \left( \sqrt{2R \log(\det(A_t)) + \det(L) + \lambda/t} \right)$ with $C_1$ and $C_2$ as positive constants, $A_t = \sum_{s=1}^{t-1} (\sigma_{i,j}^s) = \sum_{(i,j) \in \Omega_t} \frac{1}{m} g_i^s g_j^s \theta_{ij}^s + \lambda I$.

With the constructed confidence interval of the estimated pairwise document rank order, olRankNet separates all the candidate document pairs into two sets, certain rank order $S_t^c$ and uncertain rank order $S_t^u$. $S_t^c$ contains the document pairs where with high probability the estimated
pairwise order is correct. For example, for document \(i\) and \(j\), if the lower confidence bound of the probability that \(i\) is better than \(j\), i.e., \(\sigma(f_{ij}) - \frac{\alpha_t \|g_{ij}\|}{\sqrt{m}} A^{-1} - \epsilon(m)\), is larger than 0.5, then \((i,j)\) belongs to \(S_c\). Otherwise, the pair belongs to \(S_u\), which indicates that the predicted rank order can still be wrong.

When constructing the ranked list, olRankNet first builds a ranking graph with all the candidate documents and the certain rank orders in \(S_c\). Then topological sort is performed, where the certain rank orders will be followed (i.e., exploitation), and uncertain rank orders will be randomized (i.e., exploration). With such a ranking strategy, olRankNet is proved to have an \(O(\log^2(T))\) cumulative pairwise regret upper bound.

Although olRankNet has a strong theoretical foundation, its scalability is severely limited due to the additional computation required for constructing the confidence interval. In particular, the covariance matrix \(A\) in Lemma 3.1 is constructed with the gradient of the scoring function with respect to the network parameters, of which the size \(p\) is very large. In order to construct the confidence interval according to Eq (3.2), the inverse of the covariance matrix \(A\) has to be computed whenever the model is updated, which results in an unacceptably high computational cost (around \(O(p^3)\)). As a consequence, it is practically impossible for olRankNet to be exactly executed. In (Jia and Wang, 2022), approximation is employed to make olRankNet operational in practice, e.g., only using the diagonal of \(A\). However, there is no theoretical guarantee for such an approximation, which unfortunately breaks the theoretical promise of olRankNet and directly leads to an unknown gap between its theoretical and empirical performance.

### 3.3 Scalable Exploration with Perturbed Feedback

To bridge the gap, we develop an efficient and scalable strategy for recognizing the certain and uncertain rank orders without explicitly constructing the confidence set. And our basic idea is to leverage the bootstrapping technique to create randomness in a neural ranker’s output. In particular, at each round, we perturb the entire user feedback history for \(N\) times with noise freshly and independently sampled from a zero-mean Gaussian distribution, and train the corresponding neural ranker as usual. Denote model \(\theta^{(n)}\) for \(n \in [N]\) as the solution of minimizing the following objective

![Figure 1: Comparison between PairRank/olRankNet and P^2LinRank/P^2NeuRank for the procedures in model training and result ranking.](image)
function with gradient descent,

\[
\theta^{(n)} = \min \sum_{s=1}^{t} \sum_{(i,j) \in \Omega_s} \left(1 - (y_{ij}^s + \gamma_{ij}^{s,(n)}) \log (1 - \sigma(f_{ij})) \right) - (y_{ij}^s + \gamma_{ij}^{s,(n)}) \log (\sigma(f_{ij})) + m \lambda/2 ||\theta - \theta_0||^2,
\]

where \(\{\gamma_{ij}^{s,(n)}\}_{s=1}^{t} \sim N(0, \nu^2)\) are Gaussian random variables that are independently sampled in each round \(t\), and \(\nu\) is a hyper-parameter that controls the strength of perturbation (and thus the exploration) in \(P^2NeurRank\).

The detailed procedure of \(P^2NeurRank\) is given in Algorithm 1. The algorithm starts by initializing the \(N\) neural rankers. At each round of interaction, given a query \(q_t\), for each pair of candidate documents, \(N\) parallel predictions about their rank order will be generated by the set of neural rankers. If all the \(N\) estimations give the same prediction about the document pair’s rank order, e.g., \(i > j\) for document \(i\) and document \(j\), then \((i, j)\) is considered as a certain rank order (line 9 - line 13 in Algorithm 1). Otherwise, the relation between these two documents is still uncertain and further exploration is needed there when generating the ranked list. Once the sets of certain and uncertain rank orders are determined, we follow the same procedure of \(olRankNet\) to generate the ranked list via topological sort with respect to the certain rank orders.

The key intuition for \(P^2NeurRank\) is to utilize the variance introduced in the randomly perturbed click feedback to encourage exploration. With the injected perturbation, there are two kinds of deviations existing in the estimated pairwise preference in each of the \(N\) parallel neural rankers: 1) the deviation caused by the observation noise introduced by the click feedback; 2) the deviation caused by the added perturbations. By properly setting the variance parameter \(\nu\) for the added perturbation, the corresponding deviation will introduce enough randomness in the model estimation. For each round of interaction, we maintain \(N\) models and with high probability, the minimum of the estimated pairwise preference serves as the pessimistic estimate of the preference.

Compared to \(olRankNet\), which requires to maintain the inverse of the covariance matrix, \(P^2NeurRank\) does not need any added computation for the purpose of exploration, besides the regular neural network updates. As a result, \(P^2NeurRank\) greatly alleviates the computation burden in neural OL2R. More specifically, in each round, the online neural ranking algorithm generally takes the following steps: (1) predict the rank order in each document pair, (2) generate the ranked list by topology sort via the constructed certain and uncertain rank orders, and (3) update the model according to the newly received click feedback. With \(p\) representing the total number of parameters in a neural ranker, \(olRankNet\) has the time complexity \(O(V_t p + V_t^2)\) for the first step. As the \(N\) neural models in \(P^2NeurRank\) are independent from each other, the time complexity of \(P^2NeurRank\) in the first step is also \(O(V_t p + V_t^2)\) by executing the \(N\) ranker’s predictions in parallel. For the third step, again by the training the \(N\) neural rankers in parallel using gradient descent, both \(P^2NeurRank\) and \(olRankNet\) have the time complexity of \(O(\tau p \sum_{s=1}^{t} |\Omega_s|)\) where \(\tau\) is the number of epochs for training the neural network. The key difference lies in the second step. \(olRankNet\) requires the inverse of covariance matrix, which has the time complexity at least \(O(p^2.2373)\). Besides, constructing the confidence interval for all the document pairs has the time complexity of \(O(V_t^2 p^2)\). While for \(P^2NeurRank\), finding the minimum of the \(N\) predictions for all the document pairs costs \(O(NV_t^2)\). Once the certain and uncertain rank orders are determined, both algorithms require \(O(V_t + E_t)\) for the topological sort, where \(E_t\) represents the number of certain rank orders and \(E_t \leq V_t\). Therefore, for the second step, \(olRankNet\) has the total time complexity as \(O(p^2.2373 + V_t^2 p^2 + V_t + E_t) = O(p^2.2373 + V_t^2 p^2)\), while \(P^2NeurRank\) has the time complexity as \(O(NV_t^2 + V_t + E_t) = O(NV_t^2)\). As \(p\) is oftentimes in the order of tens of thousands (if not less), \(P^2NeurRank\) greatly reduces the time required for performing exploration in neural OL2R. And also empirically, the number of parallel rankers \(N\) in \(P^2NeurRank\) does not need to be large. For example, in our experiments, we found \(N = 2\) already led to promising performance of \(P^2NeurRank\) comparing to \(olRankNet\).
We want to highlight that our proposed perturbation-based exploration strategy can also be applied to linear ranking models, e.g., PairRank (Jia et al., 2021). The procedure is almost the same as described in Algorithm 1, and so is its computational advantage in linear models, especially when the dimension of the feature vectors is large. In our experiments, we empirically evaluated our perturbation-based method in PairRank (named P²LinRank) and observed its expected performance and computational advantages.

4. Regret Analysis

In this section, we provide the regret analysis of the proposed exploration strategy. For better readability, we present the analysis of a linear ranker. According to the analysis in (Jia and Wang, 2022), under the neural tangent technique and the convergence analysis of the gradient descent in neural network optimization, the linear analysis can be readily applied to the neural ranker. And we discuss the difference between the analysis between the linear ranker and neural ranker in the appendix.

Follow the standard assumption in (Jia et al., 2021; Jia and Wang, 2022), we assume that on the examined documents where \( \pi_t(i) \leq \alpha_t \), the obtained feedback \( C_t \) is independent from each other given the true relevance of documents, so is their noise (Joachims et al., 2005; Guo et al., 2009a,b). Therefore, the noise in the inferred preference pair becomes the sum of noise from the clicks in the two associated documents. And we also only use the independent pairs to construct \( \Omega_t \) as suggested in PairRank and oR ankNet. Thus, the pairwise noise satisfies the following proposition.
Proposition 4.1. For any \( t \geq 1 \), \( \forall (i, j) \in \Omega_t \), the pairwise feedback follows \( y_{ij}^t = \sigma(f(x_i; \theta^*) - f(x_j; \theta^*)) + \epsilon_{ij}^t \), where \( \epsilon_{ij}^t \) satisfies that for all \( \beta \in \mathbb{R} \), \( \mathbb{E}[\exp(\beta \epsilon_{ij}^t)] \{ \{ \epsilon_{i,j}' \}_{i',j' \in \Omega_t} \}_{t=1}^{t-1}, \Omega_{1:t-1} \leq \exp(\beta^2 R^2) \), is an R-sub-Gaussian random variable.

To train a linear ranker, we have the scoring function \( f(x; \theta) = x^\top \theta \). And we assume that \( \|x\| \leq P \) and \( \|\theta\| \leq Q \). The loss function can be rewritten as,

\[
L^{(n)}_t(\theta) = \sum_{s=-d+1}^t \sum_{(i,j) \in \Omega_s} -(y_{ij}^s + \gamma_{ij}^s(n)) \log(\sigma(x_{ij}^s \top \theta)) - (1 - (y_{ij}^s + \gamma_{ij}^s(n))) \log(1 - \sigma(x_{ij}^s \top \theta)),
\]

(4.1)

where \( x_{ij}^s = x_i^s - x_j^s \) is the difference between the feature vectors of document \( i \) and \( j \), and \( d \) is the dimension of the feature vectors. With \( |\Omega_s| = 1 \), \( x_i^s = \sqrt{\lambda} e_i \), \( y_{ij}^s = 0 \) for \( s \in [-d+1, 0] \), this loss function can be interpreted as adding \( l_2 \) regularization to the cross-entropy loss.

Given this objective function is log-convex with respect to \( \theta \), its solution \( \hat{\theta}^{(n)}_t \) of ranker \( n \) for \( n \in [N] \) is unique under the following estimation method at each round \( t \),

\[
\sum_{s=-d+1}^{t-1} \sum_{(i,j) \in \Omega_s} \left( \sigma(x_{ij}^s \top \theta) - (y_{ij}^s + \gamma_{ij}^s(n)) \right) x_{ij}^s + \lambda \theta = 0
\]

(4.2)

Let \( g_t(\theta) = \sum_{s=-d+1}^{t-1} \sum_{(i,j) \in \Omega_s} \sigma(x_{ij}^s \top \theta) x_{ij}^s + \lambda \theta \) be the invertible function such that the estimated parameter \( \hat{\theta}^{(n)}_t \) satisfies \( g_t(\hat{\theta}^{(n)}_t) = \sum_{s=-d+1}^{t-1} \sum_{(i,j) \in \Omega_s} (y_{ij}^s + \gamma_{ij}^s(n)) x_{ij}^s \). As discussed before, there are two kinds of deviations inside this estimation \( \hat{\theta}^{(n)}_t \). To analyze their effect in the estimation, we introduce an auxiliary solution \( \bar{\theta}_t \) for solving the linear objective function, which satisfies \( g_t(\hat{\theta}_t) = \sum_{s=-d+1}^{t-1} \sum_{(i,j) \in \Omega_s} y_{ij}^s x_{ij}^s \). Then, for the two solutions \( \hat{\theta}^{(n)}_t \) and \( \bar{\theta}_t \), we have the following lemmas quantifying the deviations in the estimation.

Lemma 4.2. (Deviation from observation noise). At round \( t < T \), for any pair of document \((x_i^t, x_j^t)\) under query \( q_t \), with probability at least \( 1 - \delta \), we have,

\[
|\sigma(x_{ij}^t \top \hat{\theta}_t) - \sigma(x_{ij}^t \top \bar{\theta}_t)| \leq \alpha_t \|x_{ij}^t\|_{A_t^{-1}},
\]

where \( \alpha_t = (2k_\mu/c_\mu)(\sqrt{R^2 \log(\det(A_t)/(\delta^2 \det(\lambda I)))} + d), A_t = \lambda I + \sum_{s=-d+1}^{t-1} \sum_{(i',j') \in \Omega_s} x_{i',j'}^s x_{i',j'}^s \top, k_\mu \) is the Lipschitz constant of the sigmoid link function \( \sigma, c_\mu = \inf_{\theta \in \Theta} \sigma'(x^\top \theta) \), with \( \sigma' \) as the first derivative of \( \sigma \).

Accordingly, we define \( E_t \) as the success event at round \( t \):

\[
E_t = \{ \forall (i, j) \in [V_t]^2, |\sigma(x_{ij}^t \top \hat{\theta}_t) - \sigma(x_{ij}^t \top \bar{\theta}_t)| \leq \alpha_t \|x_{ij}^t\|_{A_t^{-1}} \}.
\]

Intuitively, \( E_t \) is the event that the auxiliary solution \( \bar{\theta}_t \) is “close” to the optimal model \( \theta^* \) at round \( t \).

As discussed before, we define the regret as the number of mis-ordered pairs. Therefore, the key step in regret analysis is to quantify the probability that an estimated preference is uncertain.

According to Algorithm 1, the certain rank order in the perturbed pairwise ranker is defined as follows,

Definition 4.3. (Certain Rank Order) At round \( t \), the rank order between documents \((i, j) \in [V_t]^2\) belongs to the set of certain rank orders \( \omega_t^c \) if and only if \( \min_{n \in [N]} \sigma(f_{ij}^{t,(n)}) > \frac{1}{2} \) or \( \max_{n \in [N]} \sigma(f_{ij}^{t,(n)}) < \frac{1}{2} \), otherwise, \((i, j) \in \omega_t^u \).
According to the definition, and the deviations caused by the observation noise and the pseudo noise, we have the following lemma quantifying the probability of an estimation being uncertain.

**Lemma 4.4.** There exist positive constants $c$, $C_1$, and $C_2$, that with $t' = \frac{2k_pP}{\epsilon_p\Delta_{\text{min}}^2} \left( \sqrt{R^2 \log(1/\delta)} + \sqrt{\lambda Q} \right) + \left( C_1 \sqrt{1+2C_2 \log(1/\delta) + (P^2 R_k n)/(\sqrt{N}\Delta_{\text{min}})} \right)^2$, $\delta \in (0, 1)$, for round $t \geq t'$, with probability at least $1 - \delta$, event $E_t$ holds with $\alpha_i$ defined in Lemma 4.2, with $N \geq \log \delta / \log(1 - \exp(-\beta^2)/(4\sqrt{\pi} \beta))$, where $\beta = \frac{c}{\epsilon_p^2 \epsilon^2}$, for a document pair $(i, j)$ that $i < j$ for the given query, the probability that the estimated pairwise preference is uncertain is upper bounded as $\mathbb{P}((i, j) \in \omega^t) \leq \frac{2N \nu_2^2 \|x_{ij}\|^2}{c^2 \epsilon^2 \Delta_{\text{min}}^2}$, where $\Delta_{\text{min}} = \min_{t \in T, (i, j) \in [V_t]^2} \sigma(x_{ij}^\top \theta^*) - \frac{1}{2}$ representing the smallest gap of pairwise difference between any pair of documents associated to the same query over time (across all queries).

The detailed proof is provided in the appendix. This lemma provides the upper bound of the probability that an estimated pairwise preference is uncertain. The key idea is to analyze the concentration and anti-concentration property of the deviation caused by the pseudo noise. In particular, the deviation caused by the pseudo noise $\gamma_i$, and the ensemble of $N$ rankers should be sufficiently large so that for document pairs $(i, j)$, the maximum estimated pairwise preference, $\max_{\alpha \in [N]} \sigma(x_{ij}^\top \theta^{(\alpha)}_i)$ is optimism to trigger exploration. On the other hand, with more observations, the probability of being uncertain will be shrinking with the concentration property of the pseudo noise.

Following the assumption in (Kveton et al., 2015; Jia et al., 2021; Jia and Wang, 2022), denote $p_t$ as the probability that the user examines all documents in $\tau_t$ at round $t$, and let $p^* = \min_{1 \leq t \leq T} p_t$ the minimal probability that all documents in a query are examined over time. The regret of the proposed model can be upper bounded as follows.

**Theorem 4.5.** Assume pairwise query-document feature vector $x_{ij}^t$ under query $q_t$, where $(i, j) \in [V_t]^2$ and $t \in [T]$, satisfies Proposition 1. With $\delta \in (0, 1)$, with probability at least $1 - \delta$, the $T$-step regret of the proposed model is upper bounded by:

$$R_T \leq R' + \frac{1}{p^* 2dV_{\text{max}} C \log \left( 1 + \frac{o_{\text{max}} TP^2}{2d\lambda} \right)}$$

where $R' = t' V_{\text{max}}$, $V_{\text{max}}$ represents the maximum number of document associated with the same query over time, and $t'$ is defined in Lemma 4.4, and $w = \sum_{s=\nu}^{T} (V_{s}^2 - 2V_{\text{max}})P^2 / \min(A_{s})$. By choosing $\delta_1 = \delta_2 = 1/T$, we have the expected regret at most $R_T \leq O(d \log^2(T))$.

We provide the detailed proof in the appendix. According to the pairwise exploration strategy, the regret only comes from the document pairs that are uncertain, e.g., random shuffling will be conducted to perform the exploration. With the quantified uncertain probability in Lemma 4.4, the pairwise regret can be upper bounded accordingly.

In neural rankers, the neural network approximation error should be considered in addition to the deviations caused by the noise. According to the analysis in (Jia et al., 2022), the variance of the added noise should be set according to the deviations caused by both the observation noise and the approximation error. Based on the theoretical analysis in (Jia and Wang, 2022), by properly setting the width of the neural network and the step size of gradient descent, the model with a neural ranker will still have a sublinear regret.

5. Experiment

In this section, we empirically compare our proposed model with an extensive list of state-of-the-art OL2R algorithms on two large public learning to rank benchmark datasets.
Table 1: Configuration of simulated click models.

|       | Click Probability | Stop Probability |
|-------|-------------------|------------------|
| 0     | 0.0               | 0.0              |
| 1     | 0.2               | 0.2              |
| 2     | 0.4               | 0.5              |
| 3     | 0.8               | 0.7              |
| 4     | 1.0               | 0.9              |

5.1 Experiment Setup

5.1.1 Dataset.

We experimented on the Yahoo! Learning to Rank Challenge dataset (Chapelle and Chang, 2011), which consists of 292,921 queries and 709,877 documents represented by 700 ranking features, and MSLR-WEB10K (Qin and Liu, 2013), which contains 10,000 queries, each having 125 documents on average represented by 136 ranking features. Both datasets are labeled on a five-grade relevance scale: from not relevant (0) to perfectly relevant (4). These two datasets are the most popularly used in literature for evaluating OL2R algorithms. We followed the train/test/validation split provided in the datasets to perform the cross-validation to make our results comparable to previously reported results.

5.1.2 User interaction simulation.

A standard simulation setting for the user clicks is adopted in our experiment (Oosterhuis and de Rijke, 2018; Jia et al., 2021), which is the most popularly used procedure for OL2R evaluations. At each round, a query is uniformly sampled from the training set. The model will then generate a ranked list and return it to the user. Dependent click model (DCM) (Guo et al., 2009b) is applied to simulate user behaviors, which assumes that the user will sequentially scan the ranked list and make click decisions on the examined documents. In DCM, the probabilities of clicking on a given document and stopping the subsequent examination are both conditioned on the document’s true relevance label. Three different model configurations are employed in our experiments to represent three different types of users. The details are shown in Table 1. In particular, perfect users will click on all relevant documents and do not stop browsing until the last returned document; navigational users are very likely to click on the first encountered highly relevant document and stop there; and informational users tend to examine more documents, but sometimes click on irrelevant documents, which contribute a significant amount of noise in the click feedback. To reflect the presentation bias, all the models only return the top 10 ranked results.

5.1.3 Baselines.

We list the OL2R solutions used for our empirical comparisons below. We performed the experiments on both linear and neural rankers to show the general effectiveness of our proposed exploration strategy. For convenience of reference, in the experiment result discussions, we name our solution applied to a linear ranker as P^2LinRank

- \( \epsilon \)-Greedy (linear and neural) (Hofmann et al., 2013): At each rank position from top to bottom, it randomly samples an unranked document with probability \( \epsilon \) or selects the next best document based on the currently learned ranker.

- DBGD (linear and neural) (Yue and Joachims, 2009): DBGD uniformly samples a direction in the entire model space for exploration and model update. Its convergence is only proved for linear rankers, though empirically previous studies also applied it to neural rankers.
• **PDGD (linear and neural)** (Oosterhuis and de Rijke, 2018): PDGD samples the next ranked document from a Plackett-Luce model and estimates gradients from the inferred pairwise preferences based on the observed user feedback. The only known theoretical property about PDGD is its estimated gradient is unbiased, but how this unbiased gradient leads model training, i.e., its convergence, is still unknown.

• **PairRank** (Jia et al., 2021): PairRank learns a pairwise logistic regression ranker online and explores by divide-and-conquer in the pairwise document ranking space. The training of PairRank is known to converge with a sublinear regret defined on the cumulative number of misordered document pairs.

• **PairRank-Diag**: This is a variant of PairRank where the diagonal approximation of its covariance matrix is used to calculate the required confidence interval.

• **olRankNet** (Jia and Wang, 2022): olRankNet is an extension of PairRank with a neural ranker, where the confidence interval is constructed via the neural tangent kernel technique. It inherits good theoretical properties from PairRank.

• **olRankNet-Diag**: This is a variant of olRankNet, where the diagonal approximation of the covariance matrix is applied to calculate the confidence interval.

### 5.1.4 Hyper-Parameter Tuning.

MSLR-WEB10K dataset is equally partitioned into five folds, while Yahoo Learning to Rank dataset is equally partitioned into two folds. We performed cross validation on each dataset. For each fold, the models are trained on the training set, and the hyper-parameters are tuned based on the performance on the validation set.

In our experiments, for all the neural rankers, a two layer neural network with width $m = 100$ is applied. We did a grid search in olRankNet, PairRank, PairRank-Diag for its regularization parameter $\lambda$ over $\{10^{-i}\}_{i=1}^4$, exploration parameter $\alpha$ over $\{10^{-i}\}_{i=1}^4$. For parameter estimation in all neural rankers, we did a grid search for learning rate of gradient descent over $\{10^{-i}\}_{i=1}^3$. PairRank and PairRank-Diag are directly optimized with L-BFGS. The model update in PDGD and DBGD is based on the optimal settings in their original paper (i.e., the exploration step size and learning rate). The hyper-parameters for PDGD and DBGD are the learning rate and the learning rate decay, for which we performed a grid search for learning rate over $\{10^{-i}\}_{i=1}^3$, and the learning rate decay is set to 0.999977. For $\text{P}^2\text{NeurRank}$ and $\text{P}^2\text{LinRank}$, we have an extra hyper-parameter $N$, which is searched over $\{2, 5, 10\}$. We fixed the total number of iterations $T$ to 5000. The experiments are executed for 10 times with different random seeds and the averaged results are reported in this paper.

### 5.1.5 Evaluations.

To evaluate an OL2R model, we report both the offline and online performance during the interactions. The offline performance is evaluated in an “online” fashion where the newly updated ranker is evaluated on a hold-out testing set against its ground-truth relevance labels. This measures how fast an OL2R model improves its ranking quality. Such a setting can be viewed as using one portion of the traffic for model update, while serving another portion with the latest model. NDCG@10 is used to assess the ranking performance. In addition to the offline evaluation, we also evaluate the models’ online result serving quality. This reflects user experience during the interactions and thus should be seriously considered. Sacrificing user experience for model training will compromise the goal of OL2R. We adopt the cumulative Normalized Discounted Cumulative Gain to assess models’ online performance. For $T$ rounds, the cumulative NDCG is calculated as

$$\text{Cumulative NDCG} = \sum_{t=1}^{T} \text{NDCG}(\tau_t) \cdot \gamma^{(t-1)},$$

12
which computes the expected utility a user receives with a probability $\gamma$ that he/she stops searching after each query (Oosterhuis and de Rijke, 2018). Following the previous work (Oosterhuis and de Rijke, 2018; Wang et al., 2019, 2018a; Jia et al., 2021; Jia and Wang, 2022), we set $\gamma = 0.995$.

5.2 Experiment Results

5.2.1 Offline and online performance

We first compare our proposed model with the baselines using a linear ranker. The results are reported in Figure 2 and 3. For $P^2$LinRank, we reported the best performance with $N = 2$ and $N = 5$. We can clearly observe $P^2$LinRank maintained PairRank’s strong advantage over other OL2R solutions, including $\epsilon$-Greedy, DBGD, and PDGD, in both online and offline evaluations across three click models. It is also obvious that a straightforward perturbation of model’s output,
i.e., ε-Greedy, basically led to the worst OL2R performance, although it is often the default choice for exploration in online learning.

More importantly, PairRank with a diagonal approximated covariance matrix showed serious degradation in its ranking performance, especially its online performance. For example, on MSLR-Web10K dataset, PairRank with diagonal approximation was even worse than PDGD in the online evaluation under both perfect and informational click models. This means an approximated covariance matrix cannot accurately measure the ranker’s estimation uncertainty. Furthermore, this inaccuracy’s impact is not deterministic: under perfect click feedback, the seriously degenerated online performance together with mild decrease in offline performance suggest the model over explored; but under informational click feedback, both online and offline performance dropped, which suggests insufficient estimation. This demonstrates the complication of using approximations in OL2R solutions, which loses all theoretical guarantees in the original analysis. As a result, it also strongly suggests olRankNet might not be optimal in practice, given the diagonal approximation employed to make its computation feasible. This will be demonstrated in our experiments next.
It is worth noting that $P^2$LinRank with $N = 2$ already exhibits faster convergence than PairRank, and simply increasing $N$ does not necessarily further improve the model’s performance. This result is very promising: as the only computational overhead in our perturbation-based exploration strategy is to estimate $N - 1$ additional rankers, the actual added cost in practice is minimum when $N = 2$. Similar observation is also obtained when applied to neural rankers.

In Figure 4 and 5, we report the results obtained on the neural rankers. First of all, olRankNet and $P^2$NeurRank still showed significant improvement over other OL2R solutions, including $\epsilon$-Greedy, DBGD, MGD and PDGD. This means the pairwise exploration implemented in olRankNet is still effective for neural OL2R. The most important finding in this experiment is that $P^2$NeurRank outperformed olRankNet, when the network width $m$ is set to 100. As we have repeatedly mentioned, though enjoying nice theoretical advantages, in practice it is impossible to use the required full covariance matrix to compute the confidence interval in olRankNet; as a result, the diagonal approximation creates an unknown gap from its theoretical guarantee to practical performance. In Figure 9, we compare the offline performance of $P^2$NeurRank with $m = 100$, olRankNet with $m = 100$, and neural models with simpler neural structures under the perfect click model. The results demonstrate that for the models with $16 \times 16$ neural network, the diagonal approximation hurts the performance compared to using the full covariance matrix. As proved in our theoretical analysis, $P^2$NeurRank enjoys the same theoretical regret guarantee as olRankNet; but because it does not need to endure the approximation, all its nice theoretical properties are preserved in its actual implementation. Compared to the results obtained in linear models, we have good reasons to believe olRankNet with full covariance matrix could perform even better, but with a much larger (if not infeasible) computational overhead.

Again the impact from the number of parallel rankers one needs to maintain for perturbation-based exploration in $P^2$NeurRank is still not sensitive. As shown in both Figure 4 and 5, $N = 2$ gave us the most promising empirical performance, with the minimum added computational overhead. This is a strongly desired property for applying $P^2$NeurRank in practice.

5.2.2 Zoom into $P^2$NeurRank.

In this experiment, we provide detailed analysis on $P^2$NeurRank. $P^2$NeurRank has only two hyper-parameters, in addition to those inherited from olRankNet, i.e., the number of parallel rankers $N$ and the scaled of pseudo noise $\nu^2$. In Figure 6 and 7, we report the online and offline performance of $P^2$NeurRank with varying value of $N$ for a fixed variance scale $\nu^2$ of the added noise. We can clearly observe that with a larger noise scale, e.g., $\nu = 0.1$, setting $N = 2$ gives the best performance, comparing to $N = 5$ and $N = 10$. When the added noise scale is small, e.g., $\nu = 0.01$, setting $N = 10$ demonstrates better performance than those with fewer number of models. This indicates that the variance of the added noise $\nu$ and the number of parallel rankers $N$ together control the exploration in $P^2$NeurRank. A large variance scale, e.g., $\nu = 0.1$, together with too many models, e.g., $N = 10$, lead to more aggressive exploration and less effective model training. A small variance, e.g., $\nu = 0.01$, together with few models might not lead to sufficient exploration for model update, which also leads to worse performance. Therefore, in practice, the value of $\nu$ and $N$ should be carefully handled to perform effective exploration. And considering the added computational overhead, using fewer parallel rankers with larger scale of added noise should be a preferred solution.

5.2.3 Efficiency comparison.

In this section, we compare the running time of our proposed $P^2$NeurRank and the olRankNet models. We performed the experiments on a NVIDIA GeForce RTX 2080Ti graphical card. As discussed before, with complex neural networks, e.g., $m = 100$, it is impossible to perform the inverse of the full covariance matrix due to both high space and time complexity. Therefore, to compare running time, we perform the experiments on the MSLR-Web10K dataset using a simpler neural network with $m = 16$, where the perfect click model is adopted to generate the clicks. The result
is reported in Figure 8. We compare the olRankNet with a full covariance matrix, olRankNet with diagonal covariance matrix, $P^2$NeurRank with $N=2$ and $P^2$NeurRank with $N=1$. For $P^2$NeurRank, no extra computation is required for exploration. Therefore, the running time of $P^2$NeurRank with $N=1$ can be viewed as the time used for the model training. We can clearly notice the big gap between the running time of olRankNet with full covariance matrix and $P^2$NeurRank with $N=1$, which indicates the computational overhead caused by constructing the confidence interval with the full covariance matrix. Using diagonal approximation greatly reduces the total running time. However, according to our previous discussion, there is no theoretical performance guarantee for such an approximation, and our empirical results show that the diagonal approximation often leads to decreased performance in both offline and online evaluations. On the other hand, $P^2$NeurRank with $N=2$ takes slightly more time than olRankNet with diagonal approximation, while the empirical performance is significantly better (shown in Figure 4 and Figure 5). Besides, in practice, the $N$ models can be trained in parallel, which will further reduce the running time. This demonstrate the feasibility and advantage of our proposed OL2R model in real applications.

To further investigate this difference, we examined neural rankers’ performance under different neural network configurations (mainly the network width), with and without diagonal approximation of the covariance matrix, in Figure 9. In particular, we tested a small enough network structure of $16 \times 16$, on which we can maintain a full covariance matrix for olRankNet. On both datasets, a full covariance matrix for olRankNet with a $16 \times 16$ network structure led to better performance than its counterpart using the diagonal approximation, while our $P^2$NeurRank maintained very similar performance with the full covariance version of olRankNet. And a larger network width indeed led to consistently better performance; however, the advantage is seriously hampered by the
diagonal approximation employed in olRankNet, because of the compromise it has to make subject to the computational overhead. Again our \( \text{P}^{2}\text{NeurRank} \) did not suffer from this limitation and thus maintained its encouraging ranking performance.
6. Conclusion

In this work, we developed a provable efficient exploration strategy for neural OL2R based on bootstrapping. Previous solutions for the purpose either do not have theoretical guarantees (Yue and Joachims, 2009; Oosterhuis and de Rijke, 2018) or are too expensive to be exactly implemented in practice (Jia and Wang, 2022). Our solution has an edge on both sides: it is proved to induce a sublinear upper regret bound counted over the number of mis-ordered pairs during online result serving, and its added computational overhead is feasible. Our extensive empirical evaluations demonstrate that our perturbation-based exploration unleashes the power of neural rankers in OL2R, with minimally added computational overhead (e.g., oftentimes only one additional ranker is needed to introduce the required exploration). And our perturbation-based exploration is general and can also be used in linear models when the input feature dimension is very large.

Our current theoretical analysis depends on gradient descent over the entire training set for model update in each round, which is still expensive and should be further optimized. We would like to investigate the possibility of more efficient model update, e.g., online stochastic gradient descent or continual learning, and the corresponding effect on model convergence and regret analysis. In addition, how to generalize our neural ranker architecture to more flexible choices, e.g., recurrent neural networks and Transformers, is another important direction of our future work.

7. Acknowledgements

We want to thank the reviewers for their insightful comments. This work is based upon work supported by National Science Foundation under grant IIS-2128019, IIS-1718216 and IIS-1553568, and Google Faculty Research Award.

References

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In Advances in Neural Information Processing Systems, pages 2312–2320, 2011.

Eugene Agichtein, Eric Brill, and Susan Dumais. Improving web search ranking by incorporating user behavior information. In Proceedings of the 29th ACM SIGIR, pages 19–26. ACM, 2006.

Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. In Advances in Neural Information Processing Systems, 2019.

Jordan T Ash, Cyril Zhang, Surbhi Goel, Akshay Krishnamurthy, and Sham Kakade. Anti-concentrated confidence bonuses for scalable exploration. arXiv preprint arXiv:2110.11202, 2021.

Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. Journal of Machine Learning Research, 3(Nov):397–422, 2002.

Christopher JC Burges. From ranknet to lambdarank to lambdamart: An overview. Learning, 11 (23-581):81, 2010.

Yuan Cao and Quanquan Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. In Advances in Neural Information Processing Systems, 2019.

Yuan Cao and Quanquan Gu. Generalization error bounds of gradient descent for learning over-parameterized deep relu networks. In the Thirty-Fourth AAAI Conference on Artificial Intelligence, 2020.
Olivier Chapelle and Yi Chang. Yahoo! learning to rank challenge overview. In Proceedings of the Learning to Rank Challenge, pages 1–24, 2011.

Olivier Chapelle, Thorsten Joachims, Filip Radlinski, and Yisong Yue. Large-scale validation and analysis of interleaved search evaluation. ACM TOIS, 30(1):6, 2012.

Zixiang Chen, Yuan Cao, Difan Zou, and Quanquan Gu. How much over-parameterization is sufficient to learn deep relu networks? arXiv preprint arXiv:1911.12360, 2019.

Amit Daniely. SGD learns the conjugate kernel class of the network. In Advances in Neural Information Processing Systems, pages 2422–2430, 2017.

Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W Bruce Croft. Neural ranking models with weak supervision. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 65–74, 2017.

Simon Du, Jason Lee, Haocchuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In International Conference on Machine Learning, pages 1675–1685, 2019.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. MIT Press, 2016. http://www.deeplearningbook.org.

Fan Guo, Chao Liu, Anitha Kannan, Tom Minka, Michael Taylor, Yi-Min Wang, and Christos Faloutsos. Click chain model in web search. In Proceedings of the 18th WWW, pages 11–20, 2009a.

Fan Guo, Chao Liu, and Yi Min Wang. Efficient multiple-click models in web search. In Proceedings of the 2nd WSDM, pages 124–131, 2009b.

Jiafeng Guo, Yixing Fan, Liang Pang, Liu Yang, Qingyao Ai, Hamed Zamani, Chen Wu, W Bruce Croft, and Xueqi Cheng. A deep look into neural ranking models for information retrieval. Information Processing & Management, 57(6):102067, 2020.

Boris Hanin and Mark Sellke. Approximating continuous functions by ReLU nets of minimal width. arXiv preprint arXiv:1710.11278, 2017.

Katja Hofmann, Shimon Whiteson, and Maarten de Rijke. Balancing exploration and exploitation in listwise and pairwise online learning to rank for information retrieval. Information Retrieval, 16(1):63–90, 2013.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. In Proceedings of the 22nd ACM international conference on Information & Knowledge Management, pages 2333–2338, 2013.

Haque Ishfaq, Qiwen Cui, Viet Nguyen, Alex Ayoub, Zhuoran Yang, Zhaoran Wang, Doina Precup, and Lin F Yang. Randomized exploration for reinforcement learning with general value function approximation. arXiv preprint arXiv:2106.07841, 2021.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In Advances in neural information processing systems, pages 8571–8580, 2018.

Yiling Jia and Hongning Wang. Learning neural ranking models online from implicit user feedback. arXiv preprint arXiv:2201.06658, 2022.
Yiling Jia, Huazheng Wang, Stephen Guo, and Hongning Wang. Pairrank: Online pairwise learning to rank by divide-and-conquer. In *Proceedings of the Web Conference 2021*, pages 146–157, 2021.

Yiling Jia, Weitong Zhang, Dongruo Zhou, Quanquan Gu, and Hongning Wang. Learning contextual bandits through perturbed rewards. *arXiv preprint arXiv:2201.09910*, 2022.

Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the 28th ACM SIGIR*, pages 154–161. ACM, 2005.

Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, Filip Radlinski, and Geri Gay. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM TOIS*, 25(2):7, 2007.

Branislav Kveton, Zheng Wen, Azin Ashkan, and Csaba Szepesvari. Combinatorial cascading bandits. In *NIPS*, pages 1450–1458, 2015.

Branislav Kveton, Chang Li, Tor Lattimore, Ilya Markov, Maarten de Rijke, Csaba Szepesvari, and Masrour Zoghi. Bubblerank: Safe online learning to rerank. *arXiv preprint arXiv:1806.05819*, 2018.

Branislav Kveton, Csaba Szepesvári, Mohammad Ghavamzadeh, and Craig Boutilier. Perturbed-history exploration in stochastic linear bandits. In *Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence (UAI)*, page 176, 2019a.

Branislav Kveton, Csaba Szepesvári, Sharan Vaswani, Zheng Wen, Tor Lattimore, and Mohammad Ghavamzadeh. Garbage in, reward out: Bootstrapping exploration in multi-armed bandits. In *International Conference on Machine Learning*, pages 3601–3610. PMLR, 2019b.

Branislav Kveton, Manzil Zaheer, Csaba Szepesvári, Lihong Li, Mohammad Ghavamzadeh, and Craig Boutilier. Randomized exploration in generalized linear bandits. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, 2020.

Tor Lattimore and Csaba Szepesvári. *Bandit Algorithms*. Cambridge University Press, 2019. In press.

Tor Lattimore, Branislav Kveton, Shuai Li, and Csaba Szepesvari. Toprank: A practical algorithm for online stochastic ranking. In *NIPS*, pages 3945–3954, 2018.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

Shuai Li, Tor Lattimore, and Csaba Szepesvári. Online learning to rank with features. *arXiv preprint arXiv:1810.02567*, 2018.

Shiyu Liang and R Srikanth. Why deep neural networks for function approximation? *arXiv preprint arXiv:1610.04161*, 2016.

Tie-Yan Liu. Learning to rank for information retrieval. 2011.

Haihao Lu and Kenji Kawaguchi. Depth creates no bad local minima. *arXiv preprint arXiv:1702.08580*, 2017.

Harrie Oosterhuis and Maarten de Rijke. Differentiable unbiased online learning to rank. In *Proceedings of the 27th ACM CIKM*, pages 1293–1302, 2018.
Rama Kumar Pasumarthi, Sebastian Bruch, Xuanhui Wang, Cheng Li, Michael Bendersky, Marc Najork, Jan Pfeifer, Nadav Golbandi, Rohan Anil, and Stephan Wolf. Tf-ranking: Scalable tensorflow library for learning-to-rank. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 2970–2978, 2019.

Tao Qin and Tie-Yan Liu. Introducing letor 4.0 datasets, 2013.

Anne Schuth, Floor Sietsma, Shimon Whiteson, Damien Lefortier, and Maarten de Rijke. Multi-leaveled comparisons for fast online evaluation. In Proceedings of the 23rd ACM CIKM, pages 71–80. ACM, 2014.

Anne Schuth, Harrie Oosterhuis, Shimon Whiteson, and Maarten de Rijke. Multileave gradient descent for fast online learning to rank. In Proceeding of the 9th ACM WSDM, pages 457–466, 2016.

Aliaksei Severyn and Alessandro Moschitti. Learning to rank short text pairs with convolutional deep neural networks. In Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval, pages 373–382, 2015.

Matus Telgarsky. Representation benefits of deep feedforward networks. arXiv preprint arXiv:1509.08101, 2015.

Matus Telgarsky. Benefits of depth in neural networks. arXiv preprint arXiv:1602.04485, 2016.

Huazheng Wang, Ramsey Langley, Sonwoo Kim, Eric McCord-Snook, and Hongning Wang. Efficient exploration of gradient space for online learning to rank. In SIGIR 2018, pages 145–154, 2018a.

Huazheng Wang, Sonwoo Kim, Eric McCord-Snook, Qingyun Wu, and Hongning Wang. Variance reduction in gradient exploration for online learning to rank. In SIGIR 2019, pages 835–844, 2019.

Xuanhui Wang, Cheng Li, Nadav Golbandi, Michael Bendersky, and Marc Najork. The lambdaloss framework for ranking metric optimization. In CIKM ’18, pages 1313–1322. ACM, 2018b.

Virginia Vassilevska Williams. Multiplying matrices faster than coppersmith-winograd. In Proceedings of the forty-fourth annual ACM symposium on Theory of computing, pages 887–898, 2012.

Dmitry Yarotsky. Error bounds for approximations with deep ReLU networks. Neural Networks, 94:103–114, 2017.

Dmitry Yarotsky. Optimal approximation of continuous functions by very deep ReLU networks. arXiv preprint arXiv:1802.03620, 2018.

Yisong Yue and Thorsten Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In ICML, pages 1201–1208, 2009.

Weitong Zhang, Dongruo Zhou, Lihong Li, and Quanquan Gu. Neural thompson sampling. arXiv preprint arXiv:2010.00827, 2020.

Dongruo Zhou, Lihong Li, and Quanquan Gu. Neural contextual bandits with ucb-based exploration. arXiv preprint arXiv:1911.04462, 2019.

Dongruo Zhou, Lihong Li, and Quanquan Gu. Neural contextual bandits with ucb-based exploration. In International Conference on Machine Learning, pages 11492–11502. PMLR, 2020.

Difan Zou and Quanquan Gu. An improved analysis of training over-parameterized deep neural networks. In Advances in Neural Information Processing Systems, 2019.

Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Stochastic gradient descent optimizes over-parameterized deep ReLU networks. Machine Learning, 2019.
Appendix A. Appendix

A.1 Proof of Lemma 4.2

Proof. According to the definition of \(g_t(\theta)\) in Section 4, we have the following equation for the auxiliary solution \(\hat{\theta}_t\),

\[
g_t(\hat{\theta}_t) = \sum_{s=-d+1}^{t-1} \sum_{(i,j) \in \Omega_s} y^s_{ij} x^s_{ij}
\]

Then, to bound the deviation caused by the observation noise, we have the following derivations for any input \(x\)

\[
|\sigma(x^T \hat{\theta}_t) - \sigma(x^T \theta^*)| \\
\leq k_\mu |x^T (\hat{\theta}_t - \theta^*)| \\
= k_\mu |x^T G_i^{-1} (g_t(\hat{\theta}_t) - g_t(\theta^*))| \\
\leq \frac{k_\mu}{c_\mu} |x^T \Lambda_i^{-1} (g_t(\hat{\theta}_t) - g_t(\theta^*))| \\
\leq \frac{k_\mu}{c_\mu} \|x\|_{\Lambda_i^{-1}} \|g_t(\hat{\theta}_t) - g_t(\theta^*)\|_{\Lambda_i^{-1}}
\]

The first inequality is due to the Lipschitz continuity of the logistic function. As logistic function is continuously differentiable, the second equality is according to the Fundamental Theorem of Calculus, where \(G_i = \sum_{s=1}^{t-1} \sum_{(i,j) \in \Omega_s} \hat{\sigma}(x^s_{ij} x^s_{ij} \gamma^s_{ij} + \lambda I)\). In the third inequality, \(A_i = \sum_{s=1}^{t-1} \sum_{(i,j) \in \Omega_s} x^s_{ij} x^s_{ij} \gamma^s_{ij} + \lambda I\). And this inequality holds as \(G_i > c_\mu A_i\), with \(c_\mu = \inf_{\theta \in \Theta} \sigma(x^T \theta)\).

Next, we will bound \(\|g_t(\hat{\theta}_t) - g_t(\theta^*)\|_{A_i^{-1}}\):

\[
\|g_t(\hat{\theta}_t) - g_t(\theta^*)\|_{A_i^{-1}} = \| \sum_{s=-d+1}^{t-1} y^s_{ij} x^s_{ij} - \sigma(x^s_{ij} \theta^*) x^s_{ij} \|_{A_i^{-1}} \\
\leq \| \sum_{s=-d+1}^{t-1} \sigma(x^s_{ij} \theta^*) x^s_{ij} \|_{A_i^{-1}} + \| \sum_{s=1}^{t-1} \epsilon^s_{ij} x^s_{ij} \|_{A_i^{-1}} \leq d + R \sqrt{\log(\det(A_i) / \delta^2 \det(A_i^\gamma))}
\]

The first equality is based on the definition of function \(g_t\), and \(\hat{\theta}_t\). The last inequality is according to the self-normalized bound for martingales in (Abbasi-Yadkori et al., 2011). This completes the proof.

A.2 Proof of Lemma 4.4

Proof. According to Lemma 4.2, with probability at least 1 – \(\delta\), event \(E_t\) defined in Section 4 occurs. Under event \(E_t\), for document pair \((i,j)\) satisfying \(i > j\) for the given query (e.g., \(\sigma(x^T_{ij} \theta^*) > \frac{1}{2}\)), we first analyze the probability that at least one estimate \(\sigma(x^T_{ij} \hat{\theta}_t^{(n)}) > \frac{1}{2}\) for \(n \in [N]\), e.g.,

\[
\mathbb{P}(\max_{n \in [N]} \sigma(x^T_{ij} \hat{\theta}_t^{(n)}) > \frac{1}{2})
\]

For simplicity, in the following analysis, we use \(\hat{\sigma}_t^{(n)}\), \(\hat{\sigma}_t\) and \(\sigma^*\) to present \(\sigma(x^T_{ij} \hat{\theta}_t^{(n)})\), \(\sigma(x^T_{ij} \hat{\theta}_t)\) and \(\sigma(x^T_{ij} \theta^*)\) respectively.

\[
\mathbb{P}(\max_{n \in [N]} \hat{\sigma}_t^{(n)} \geq \frac{1}{2}) = 1 - \prod_{n=1}^{N} \mathbb{P}(\hat{\sigma}_t^{(n)} < \frac{1}{2})
\]
For any \( n \in [N] \), we have the following bound for \( \mathbb{P}(\sigma_i^{(n)} < \frac{1}{2}) \).

\[
\mathbb{P}(\sigma_i^{(n)} < \frac{1}{2}) \leq \mathbb{P}(\sigma_i^{(n)} < \sigma^*) = \mathbb{P}(\sigma_i^{(n)} - \bar{\sigma}_t < \sigma^* - \bar{\sigma}_t) \\
\quad \leq \mathbb{P}(\sigma_i^{(n)} - \bar{\sigma}_t < \alpha_t\|x_{ij}^t\|_{A_t^{-1}}) \\
\quad \leq \mathbb{P}(c_n x_{ij}^t \top (\hat{\theta}_t - \theta) < \alpha_t\|x_{ij}^t\|_{A_t^{-1}}) \\
\quad \leq \mathbb{P}(c_n x_{ij}^t \top g_t^{-1}(g_t(\hat{\theta}_t) - g_t(\theta)) < \alpha_t\|x_{ij}^t\|_{A_t^{-1}}) \\
\quad \leq \mathbb{P}\left(\frac{c_n}{\sqrt{k}} x_{ij}^t \top A_t^{-1}(g_t(\hat{\theta}_t) - g_t(\theta)) < \alpha_t\|x_{ij}^t\|_{A_t^{-1}}\right) \\
\quad = \mathbb{P}(U(x_{ij}^t) < \frac{k}{c_n} \alpha_t\|x_{ij}^t\|_{A_t^{-1}})
\]

where \( U(x_{ij}^t) = x_{ij}^t \top A_t^{-1}(g_t(\hat{\theta}_t) - g_t(\theta)) \). According to the definition of \( \bar{\theta}_t \), we know that,

\[
U(x_{ij}^t) = x_{ij}^t \top A_t^{-1} \sum_{s=-d+1}^{t-1} \gamma_{ij}^s x_{ij}^s
\]

It is easy to obtain that \( \mathbb{E}[U(x_{ij}^t)] = 0 \) with \( \mathbb{E}[\gamma] = 0 \). And the variance of \( U(x_{ij}^t) \) is,

\[
Var[U(x_{ij}^t)] = \nu^2 x_{ij}^t \top A_t \left( \sum_{s=-d+1}^{t-1} x_{ij}^s x_{ij}^s \top \right) A_t x_{ij}^t \\
\quad = \nu^2 x_{ij}^t \top A_t^{-1}(\lambda I + \sum_{s=1}^{t-1} x_{ij}^s x_{ij}^s \top) A_t^{-1} x_{ij}^t \\
\quad = \nu^2 \|x_{ij}^t\|^2_{A_t^{-1}}
\]

Therefore, we have \( U(x_{ij}^t) \sim \mathcal{N}(0, \nu^2\|x_{ij}^t\|^2_{A_t^{-1}}) \), and the probability can be upper bounded as,

\[
\mathbb{P}(U(x_{ij}^t) < \frac{k}{c_n} \alpha_t\|x_{ij}^t\|_{A_t^{-1}}) = 1 - \mathbb{P}(U(x_{ij}^t) > \frac{k}{c_n} \alpha_t\|x_{ij}^t\|_{A_t^{-1}}) \\
\quad \leq 1 - \frac{\exp(-\beta^2)}{4\sqrt{\pi}\beta}
\]

where \( \beta = \frac{k\alpha_t}{c_n\|x_{ij}^t\|_{A_t^{-1}}} \). By chaining all the inequalities, we have that with \( N \geq \log \delta / \log(1 - \exp(-\beta^2)/(4\sqrt{\pi}\beta)) \), with probability at least \( 1 - \delta \), \( \mathbb{P}(\max_{n \in [N]} \bar{\sigma}_t^{(n)} \geq \frac{1}{2}) \).

Therefore, under event \( E_t \), with \( N \geq \log \delta / \log(1 - \exp(-\beta^2)/(4\sqrt{\pi}\beta)) \), based on the definition of \( \omega_t^n \) in Section 3.2, we know that for document \( i \) and \( j \) at round \( t \), \((i, j) \in \omega_t^n \) if and only if \( \min_{n \in [N]} \sigma \left(x_{ij}^t \top \hat{\theta}_t^n\right) > \frac{1}{2} \). Then the probability of being uncertain can be bounded as,

\[
\mathbb{P}((i, j) \in \omega_t^n) = 1 - \mathbb{P}\left( \min_{n \in [N]} \hat{\sigma}_t^{(n)} > 1/2 \right) \\
\quad = 1 - \prod_{n=1}^{N} \mathbb{P}(\hat{\sigma}_t^{(n)} > 1/2) = 1 - (\mathbb{P}(\hat{\sigma}_t^{(n)} > 1/2))^N
\]

Based on the definition of \( \Delta_{\min}, \sigma^* - 1/2 \geq \Delta_{\min} \) and \( \sigma^* - \bar{\sigma}_t \leq CB \). And according to the random matrix theory and Lemma 2 in (Jia et al., 2021), when \( t > t^* \), \( \Delta_{\min} - CB > 0 \) which can be viewed
as a constant $c \Delta_{\min}$. Therefore, we have the following inequalities,
\[
\mathbb{P}(\hat{\sigma}_t^{(m)} > 1/2) = \mathbb{P}(\hat{\sigma}_t^{(m)} - \sigma^* + \sigma^* - \hat{\sigma}_t + \hat{\sigma}_t > 1/2)
\geq \mathbb{P}(\hat{\sigma}_t - \hat{\sigma}_t^{(m)} < \Delta_{\min} - CB) = 1 - \frac{1}{2} \mathbb{P}(|\hat{\sigma}_t - \hat{\sigma}_t^{(m)}| \geq \Delta_{\min} - CB)
\geq 1 - \frac{1}{2} \mathbb{P}(k_\mu|U(x)| \geq \Delta_{\min} - CB)
\geq 1 - \exp(-\frac{c^2 \mu^2 \Delta_{\min}^2}{2 k_\mu^2 \nu^2 ||x||^2 A_t^{-1}})
\]

Let $B = \frac{c^2 \mu^2 \Delta_{\min}^2}{2 k_\mu^2 \nu^2 ||x||^2 A_t^{-1}}$, we have the probability of being uncertain rank order upper bounded as,
\[
\mathbb{P}((i, j) \in \omega_t^n) \leq 1 - (1 - \exp(-B))^N
= 1 - \exp(N \log(1 - \exp(-B))
\leq - N \log(1 - \exp(-B)) \leq -N \log \exp(-1/B)
\]

Therefore, $\mathbb{P}((i, j) \in \omega_t^n) \leq \min \left\{1, \frac{2 N k_\mu^2 \nu^2 ||x||^2 A_t^{-1}}{c^2 \mu^2 \Delta_{\min}^2} \right\}$. This completes the proof. \hfill \qed

A.3 Proof of Theorem 4.5

Proof. Once the certain and uncertain rank orders are determined, our proposed model will generate the ranked list by topological sort with respect to the certain rank orders. Therefore, the regret only comes from the uncertain rank orders. In each round of result serving, as the model $\theta_t$ would not change until the next round, the expected number of uncertain rank orders $U_t$, can be estimated by summing the uncertain probabilities over all possible pairwise comparisons under the current query $q_t$, e.g., $\mathbb{E}[U_t] = \frac{V_t(V_t-1)}{2} \mathbb{P}((i, j) \in \omega_t^n)$.

Based on Lemma 4.4, the cumulative number of mis-ordered pairs can be bounded by the probability of observing uncertain rank orders in each round, which shrinks with more observations become available over time,
\[
\mathbb{E}[\sum_{s=t'}^{T} U_t] \leq \mathbb{E}\left[\frac{1}{2} \sum_{s=t'}^{T} \sum_{(i, j) \in [V_t]^2} \frac{2 N k_\mu^2 \nu^2 ||x||^2 A_t^{-1}}{c^2 \mu^2 \Delta_{\min}^2} \right].
\]

As $A_t$ only contains information of observed document pairs so far, the number of mis-ordered pairs among the observed documents is guaranteed to be upper bounded. To reason about the number of mis-ordered pairs in those unobserved documents (i.e., from $o_t$ to $V_t$ for each query $q_t$), we leverage the constant $p^*$, which is defined as the minimal probability that all documents in a query are examined over time,
\[
\mathbb{E}[\sum_{t=t'}^{T} \sum_{(i, j) \in [V_t]^2} ||x||^2 A_t^{-1} \max \{o_t = V_t\}]
\leq p^* \mathbb{E}[\sum_{t=t'}^{T} \sum_{(i, j) \in [V_t]^2} ||x||^2 A_t^{-1} \max \{o_t = V_t\}]
\]

Besides, we only use the independent pairs $\Omega_t$ to update the model and the corresponding $A_t$ matrix. Therefore, to bound the regret, the pairs can be divided into two parts based on whether they are belonging to the observed set $\Omega_t$. Then, we have the following inequalities,
\[
\sum_{s=t'}^{T} \sum_{(i, j) \in \Omega_s} \sum_{k \in [V_s]\{i, j\}} 2 x_{ik} A_t^{-1} x_{jk} \leq \sum_{s=t'}^{T} (V_{\max}^2 - 2V_{\max}) P^2 / \lambda_{\min}(A_s).
\]
And for the pairs belonging to the \( \{\Omega_s\}_{s=1}^T \), based on Lemma 10 and Lemma 11 in (Abbasi-Yadkori et al., 2011), we have,

\[
\sum_{s=t}^T \sum_{(i,j) \in \Omega_s} (V_{ij} - 1)\|x_{ij}\|_2 \leq 2dV_{\text{max}} \log \left( 1 + \frac{o_{\text{max}}T P^2}{2d\lambda} \right)
\]

Chaining all the inequalities, we have when event \( E_t \) happens, the regret can be upper bounded as,

\[
R_T \leq R' + \sum_{s=t}^T r_s \leq R' + \frac{1}{p} 2dV_{\text{max}} C \log \left( 1 + \frac{o_{\text{max}}T P^2}{2d\lambda} \right)
\]

where \( C = 2Nk^2 \nu^2 / \gamma_{\mu}^2 \Delta^{2}_{\text{min}}, R' = t'V_{\text{max}}, \) with \( t' \) defined in Lemma 4.4. This completes the proof.

\( \square \)

### A.4 Regret analysis of P^2NeurRank

According to the setting of olRankNet (Jia and Wang, 2022), we assume for the learning to rank problem, there exists an unknown function \( h(\cdot) \) that models the relevance quality of document \( x \) under the given query \( q \) as \( h(x) \). In order to learn this function, we utilize a fully connected neural network \( f(x; \theta) = \sqrt{m} W_L \phi(W_{L-1} \phi(\ldots(\phi(W_1 x))) \right), \) where depth \( L \geq 2, \phi(x) = \max\{x, 0\} \), and \( W_1 \in \mathbb{R}^{m \times d}, W_i \in \mathbb{R}^{m \times m}, 2 \leq i \leq L - 1, W_L \in \mathbb{R}^{m \times 1}, \) and \( \theta = [\vec{v}(W_1)^\top, \ldots, \vec{v}(W_L)^\top] \in \mathbb{R}^p \) with \( p = m + md + m^2(L - 2) \). Without loss of generality, we assume the width of each hidden layer is the same as \( m \), concerning the simplicity of theoretical analysis. We also denote the gradient of the neural network function as \( \nabla g(x; \theta) = \nabla f(x; \theta) \in \mathbb{R}^p \). Therefore, the loss function is,

\[
L^{(n)}_i(\theta) = \sum_{s=1}^{t-1} \sum_{(i,j) \in \Omega_s} - (y_{ij}^s + \gamma_{ij}^{s,(n)}) \log(\sigma(f_{ij}^s)) - (1 - (y_{ij}^s + \gamma_{ij}^{s,(n)})) \log (1 - \sigma(f_{ij}^s))
\]

\[
+ \frac{m\lambda}{2} \sum_{l=1}^p (e_l^\top (\theta - \theta_0))^2,
\]

where the \( \ell_2 \) regularization \( \frac{m\lambda}{2} \|\theta - \theta_0\|^2 \) in Eq (3.3) is re-written as \( \frac{m\lambda}{2} \sum_{l=1}^p (e_l^\top (\theta - \theta_0))^2 \) with \( e_l \) as the standard basis of the \( p \)-dimensional space.

For the regret analysis of the neural ranker, P^2NeurRank, we slightly modified the perturbation strategy, where the loss function of the \( n \)-th model is,

\[
L^{(n)}_i(\theta) = \sum_{s=1}^{t-1} \sum_{(i,j) \in \Omega_s} - (y_{ij}^s + \gamma_{ij}^{s,(n)}) \log(\sigma(f_{ij}^s)) - (1 - (y_{ij}^s + \gamma_{ij}^{s,(n)})) \log (1 - \sigma(f_{ij}^s))
\]

\[
+ \frac{m\lambda}{2} \sum_{l=1}^p (e_l^\top (\theta - \theta_0) + 1/\sqrt{m\lambda} \gamma_i^{(n)} e_l)^2.
\]

Different from the linear model where we only perturb the observed click feedback, we also perturbed the model by adding the pseudo noise to each dimension of the model parameter, \( \gamma_i^{(n)} e_l \). In particular, both the pseudo noise for the observed click feedback and the model parameters are sampled from the same distribution: \( \gamma_{ij}^{s,(n)} \sim \mathcal{N}(0, \nu^2) \) and \( \gamma_i^{(n)} \sim \mathcal{N}(0, \nu^2) \). To first confirm this difference does not change the empirical performance of P^2NeurRank, we followed the same setting in Section 5 to evaluate this new variant. Figure 10 and Figure 11 show the comparison between the P^2NeurRank solution that only perturbs the click observations and that also perturbs the model parameters. We can observe that both solutions can achieve better offline and online performance.
than the existing baselines and the olRankNet model with diagonal approximation of the covariance matrix. For the two models with different choices of the perturbation, they have similar performance.

The following is the key conclusion of the regret analysis of P²NeurRank. The difference between the analysis of P²NeurRank and P²LinRank lies in the quantification of the probability that an estimated pairwise preference being uncertain. With a neural ranker, we need to take the neural network’s approximation error into consideration when quantifying the probability.

**Lemma A.1.** There exist positive constants $c, \{C_i\}_{i=1}^4, \{C_i\}_{i=1}^4$, if the step size of gradient descent $\eta \leq C_1(TmL + m\lambda)^{-1}$ and $m \geq C_2 \max \{\lambda^{-1/2}L^{-3/2}(\log(TV_{\max}L^2/\delta))^{3/2}, T^7L^{-7}L^{23}(\log m)^3\}$, with

$$t'' = \frac{2k_P \rho}{\sqrt{\min(\lambda)}} \left( \sqrt{R^2 \log(\frac{1}{\delta})} + \sqrt{XQ} \right) + \left( \frac{c_2 \sqrt{k_P \rho} \log(1/\delta) + (P^2 R_{E_0} \delta^{-1/2})}{\sqrt{XQ} \Delta_{\min}(\lambda)} \right)^2, \delta \in (0, 1),$$

for round $t \geq t''$, with probability at least $1 - \delta$, event $E_t$ holds with $\alpha_t$ defined in Lemma 4.2, with $N \geq \log \delta/\log(1 - \exp(-\beta^2/(4\sqrt{\beta})))$, where $\beta = k_2 \rho^2 \sqrt{\frac{t''}{P^2 \rho}}$, for a document pair $(i, j)$ that $i \succ j$ for the given query, the probability that the estimated pairwise preference is uncertain is upper bounded.
as \( P((i, j) \in \omega^p) \leq \min \left\{ 1, \frac{2N\nu^2\|\theta^*\|^2}{c^2\Delta_{\text{min}}^2} \right\} \), where \( \Delta_{\text{min}} = \min_{t \in T, (i, j) \in [V_t]^2} |\sigma(h_{ij}) - \frac{1}{2}| \) representing the smallest gap of pairwise difference between any pair of documents associated to the same query over time (across all queries), \( \epsilon(m) = \tilde{C}_1 \left( T^{7/6}m^{-1/6}\lambda^{-7/6}L^4 \sqrt{\log(m)}(1+\sqrt{T/\lambda})+(1-\eta m \lambda)^{3/2}\sqrt{TL/\lambda}+T^{1/6}m^{-1/6}\lambda^{-1/6}L^{7/2} \sqrt{\log(m)}S+T^{2/3}m^{-1/6}\lambda^{-2/3}L^3 \sqrt{\log(m)} \right) \) is the approximation error of the neural network.

### A.4.1 Proof of Lemma A.1

To quantify the probability that an estimated pairwise preference being uncertain, we start from introducing the background of neural tangent kernel technique.

We first assume that there are \( n_T \) possible documents to be evaluated during the model learning. It is easy to conclude that \( n_T = \sum_{t=1}^{T} V_T \leq TV_{\text{max}} \). Now, we introduce the neural tangent kernel matrix defined on the \( n_T \) possible query-document feature vectors across \( T \) rounds, \( \{x_i\}_{i=1}^{n_T} \).

**Definition A.2** (Jacot et al. (2018); Cao and Gu (2019)). Let \( \{x_i\}_{i=1}^{n_T} \) be the set of all pairwise document feature vectors. Define

\[
\tilde{H}_{i,j}^{(l)} = \Sigma_{i,j}^{(l)} = \langle x_i, x_j \rangle, \quad B_{i,j}^{(l)} = \left( \begin{array}{cc} \Sigma_{i,i}^{(l)} & \Sigma_{i,j}^{(l)} \\ \Sigma_{i,j}^{(l)} & \Sigma_{j,j}^{(l)} \end{array} \right), \\
\Sigma_{i,j}^{(l+1)} = 2E_{(u,v) \sim N(0, B_{i,j}^{(l)})} \left[ \phi(u)\phi(v) \right], \\
\tilde{H}_{i,j}^{(l+1)} = 2\tilde{H}_{i,j}^{(l)}E_{(u,v) \sim N(0, B_{i,j}^{(l)})} \left[ \phi'(u)\phi'(v) \right] + \Sigma_{i,j}^{(l+1)}.
\]

Then, \( \mathbf{H} = (\tilde{H}^{(L)} + \Sigma^{(L)})/2 \) is called the neural tangent kernel (NTK) matrix on the context set \( \{x_i\}_{i=1}^{n_T} \).

We also need the following assumption on the NTK matrix and the corresponding feature set.

**Assumption A.3.** \( \mathbf{H} \geq \lambda_0 \mathbf{I} \); moreover, for any \( 1 \leq i \leq n, \|x_i\|_2 = 1 \) and \( |x_i|_j = |x_i|_{j+d/2} \).

With this assumption, the NTK matrix is assumed to be non-singular (Du et al., 2019; Arora et al., 2019; Cao and Gu, 2019). This assumption can be easily satisfied when no two query-document feature vectors are in parallel. The second assumption is for convenience in analysis and can be easily satisfied by: for any context \( x, \|x\|_2 = 1 \), we can construct a new context \( x' = [x^T, x^1]^T/\sqrt{2} \). Equipped with this assumption, it can be verified that with \( \theta_0 \) initialized as in Algorithm 1, \( f(x_i; \theta_0) = 0 \) for any \( i \in [n_T] \).

Besides, we need the following technical lemmas for the regret analysis.

**Lemma A.4** (Lemma 5.1, Zhou et al. (2020)). There exists a positive constant \( \tilde{C} \) such that for any \( \delta \in (0, 1) \), if \( m \geq \tilde{C}n_T^2L^6 \log(n_T^2L^2/\delta)/\lambda_0^2 \), then with probability at least \( 1 - \delta \), there exists a \( \theta^* \in \mathbb{R}^p \) such that for any \( i \in [n_T] \), with \( h(x_i) = (h(x_{i1}), \ldots, h(x_{id})) \),

\[
h(x_i) = \langle g(x_i; \theta_0), \theta^* - \theta_0 \rangle, \quad \sqrt{m}\|\theta^* - \theta_0\|_2 \leq \sqrt{2h^\top H^{-1}h} \leq S, \quad \text{(A.3)}
\]

**Lemma A.5** (Lemma B.4, Zhou et al. (2020)). There exist constants \( \{C_i\}_{i=1}^{3} \) such that for any \( \delta > 0 \), if \( \tau \) satisfies that

\[
C_i m^{-3/2}L^{-3/2}\log(n_T L^2/\delta)^{3/2} \leq \tau \leq C_2 L^{-6}\log m^{-3/2},
\]

then with probability at least \( 1 - \delta \), for any \( \tilde{\theta} \) and \( \hat{\theta} \) satisfying \( \|\tilde{\theta} - \theta_0\|_2 \leq \tau, \|\hat{\theta} - \theta_0\|_2 \leq \tau \) and \( i \in [n_T] \) we have

\[
|f(x_i; \tilde{\theta}) - f(x_i; \hat{\theta}) - \langle g(x_i; \hat{\theta}), \tilde{\theta} - \hat{\theta} \rangle| \leq C_3 \tau^{4/3} L \sqrt{m \log m}.
\]
Lemma A.6 (Lemma B.5, Zhou et al. (2020)). There exist constants $\{C^i_1\}_{i=1}^3 > 0$ such that for any $\delta \in (0, 1)$, if $\tau$ satisfies that
\[
C^i_1 m^{-3/2} L^{-3/2} \max\{\log^{-3/2} m, \log^{3/2}(nT/\delta)\} \leq \tau \leq C^i_2 L^{-9/2} \log^{-3} m,
\]
then with probability at least $1 - \delta$, for all $\|\theta - \theta_0\|_2 \leq \tau$ and $i \in [nT]$ we have
\[
\|g(x_i; \theta) - g(x_i; \theta_0)\|_2 \leq C^i_3 \sqrt{\log m} \tau^{1/3} L^3 \|g(x_i; \theta_0)\|_2.
\]

Lemma A.7 (Lemma B.6, Zhou et al. (2020)). There exist constants $\{C^i_1\}_{i=1}^3 > 0$ such that for any $\delta > 0$, if $\tau$ satisfies that
\[
C^i_1 m^{-3/2} L^{-3/2} \max\{\log(nT^2/\delta)\}/3^3 \leq \tau \leq C^i_2 L^{-6} [\log m]^{-3/2},
\]
then with probability at least $1 - \delta$, for any $\|\theta - \theta_0\|_2 \leq \tau$ and $i \in [nT]$ we have $\|g(x_i; \theta)\|_F \leq C^i_3 \sqrt{mL}$.

Lemma A.8 (Lemma B.2, Zhou et al. (2020)). There exist constants $\{C^i_1\}_{i=1}^3 > 0$ such that for any $\delta > 0$, if for all $t \in [T]$, $\eta$ and $m$ satisfy
\[
\sqrt{2n^L_p/(m\lambda)} \geq C_1 m^{-3/2} L^{-3/2} \max\{\log(nT^2/\delta)\}/3^3,
\]
then with probability at least $1 - \delta$, we have that $\|\theta_t - \theta_0\|_2 \leq \sqrt{2n^L_p/(m\lambda)}$ and
\[
\|\theta_t - \theta_0 - \tilde{\gamma}_t\|_2 \leq (1 - \eta m\lambda)^{1/2} \sqrt{2n^L_p/(m\lambda)} - m^{-2/3} \sqrt{\log m L^7/2 (n^L_p)^{-7/6} \lambda^{-7/6} (C_5 + C_6 \sqrt{n^L_p/\lambda})}.
\]

Based on the above definitions and technical lemmas borrowed from previous work, we are now equipped to prove Lemma A.1. We first present the following two lemmas specific to our perturbation based exploration strategy for the regret analysis.

Lemma A.9. For any $t \in [T]$, with $\tilde{\omega}^a_t$ defined as the solution of the following equation,
\[
\tilde{\omega}^a_t = \min_{\omega_t} \sum_{s=1}^{t-1} \sum_{(i,j) \in \Omega_s} - (y_{i,j}^s + \gamma_{i,j}^{s(n)}) \log \left( \sigma(\langle g_{i,j}^{s,0}, \omega_t \rangle) \right) - (1 - y_{i,j}^s - \gamma_{i,j}^{s(n)}) \log \left( 1 - \sigma(\langle g_{i,j}^{s,0}, \omega_t \rangle) \right)
+ \frac{m\lambda}{2} \sum_{l=1}^{p} (e_i^l)(\theta - \theta_0) + \sqrt{\frac{\Lambda}{m}} \gamma^{(n)}_l e_l^2,
\]
where $\tilde{\gamma}$ represents the expectation of the random variable $\gamma$. Then, with the pairwise noise $\gamma_{i,j}$ satisfying Proposition 4.1, for any document pair $i$ and $j$, with probability at least $1 - \delta$, we have,
\[
|\langle g_{i,j}^{s,0}, \tilde{\omega}^a_\tau - \omega^* \rangle| \leq \alpha_t \|g_{i,j}^{s,0}/\sqrt{m}\|_{A_t^{-1}}, \quad \text{with } \alpha_t = \frac{1}{c_\nu} (R\sqrt{\log \det(A_t^{-1})} + \sqrt{\Lambda} S), \quad A_t = \sum_{s=1}^{t} \sum_{(i,j) \in \Omega_s} g_{i,j}^s g_{i,j}^{s,0\top} / m + \lambda I, \quad S \text{ is the upper bound of } \sqrt{m\|\theta^* - \theta_0\|_2} \text{ in Lemma A.4}.
\]

Lemma A.10. For any $t \in [T]$, with $\tilde{\omega}^a_t$ defined as the solution of the following equation,
\[
\tilde{\omega}^a_t = \min_{\omega_t} \sum_{s=1}^{t-1} \sum_{(i,j) \in \Omega_s} - (y_{i,j}^s + \gamma_{i,j}^{s(n)}) \log \left( \sigma(\langle g_{i,j}^{s,0}, \omega_t \rangle) \right) - (1 - y_{i,j}^s - \gamma_{i,j}^{s(n)}) \log \left( 1 - \sigma(\langle g_{i,j}^{s,0}, \omega_t \rangle) \right)
+ \frac{m\lambda}{2} \sum_{l=1}^{p} (e_i^l)(\theta - \theta_0) + \sqrt{\frac{\Lambda}{m}} \gamma^{(n)}_l e_l^2.
\]
Then, for any pairwise gradient $g$, we have $U(g) \leq g^\top (\tilde{\omega}_t - \tilde{\omega}_t) \leq \frac{1}{c_\nu} U(g)$, and $U(g) \sim \mathcal{N}(0, \nu^2 \|g/\sqrt{m}\|_{A_t^{-1}}^2)$.
Based on everything presented above, we are finally ready to provide the detailed proofs of Lemma A.1 as follows.

**Proof.** Similar to the proof of Lemma 4.4, we first analyze for document pair $(i, j)$ satisfying the probability that $i > j$ for the given query (e.g., $\sigma(x_{ij}^v \hat{\theta}^*) > \frac{1}{2}$), the probability that at least one estimation $\sigma(f_{i,j}^{t(n)}) > \frac{1}{2}$ for $n \in [N]$, e.g., $\mathbb{P}(\max_{n \in [N]} \sigma(f_{i,j}^{t(n)}) > \frac{1}{2})$. Then we have

$$\mathbb{P}(\max_{n \in [N]} \sigma(f_{i,j}^{t(n)}) > \frac{1}{2}) = 1 - \prod_{n=1}^N \mathbb{P}(\sigma(f_{i,j}^{t(n)}) < \frac{1}{2})$$

For any $n \in [N]$, according to the condition, we have the following bound for $\mathbb{P}(\sigma(f_{i,j}^{t(n)}) < \frac{1}{2})$:

$$\mathbb{P}(\sigma(f_{i,j}^{t(n)}) < \frac{1}{2}) \leq \mathbb{P}(\sigma(f_{i,j}^{t(n)}) < \sigma(h_{i,j}) - c_{\mu} \epsilon(m)) \leq \mathbb{P}(c_{\mu}(f_{i,j}^{t(n)} - h_{i,j}) < -c_{\mu} \epsilon(m))$$

$$\leq \mathbb{P}(\langle g_i^0, \hat{\omega}_t - \hat{\omega}_i \rangle < \langle g_i^0, \omega^* - \hat{\omega}_i \rangle) \leq \mathbb{P}(\langle g_i^0, \hat{\omega}_t - \hat{\omega}_i \rangle < \alpha_t \|g_i^0\|_A^{-1})$$

$$\leq \mathbb{P}(U(g_i^0) < \alpha_t \|g_i^0\|_A^{-1}) \leq 1 - \frac{\exp(-\beta^2)}{4\sqrt{\pi} \beta}$$

where $\beta = \frac{c_{\mu} \epsilon(m)}{\|g_i^0\|_A^{-1}}$. The first inequality is based on the definition of the lemma, where $\sigma(h_{i,j}) - c_{\mu} \epsilon(m) > 1/2$. The second inequality holds with $c_{\mu} = \inf \hat{\sigma}$. The third inequality holds with the following inequalities.

$$f(x_i; \hat{\theta}_{t-1}) - h(x_i) = f(x_i; \hat{\theta}_{t-1}) - f(x_i; \theta_0) - \langle g(x_i; \theta_0), \theta_{t-1} - \theta_0 \rangle + \langle g(x_i; \theta_0), \hat{\omega}_t - \hat{\omega}_i \rangle + \langle g(x_i; \theta_0), \hat{\omega}_t - \hat{\omega}_i \rangle + \langle g(x_i; \theta_0), \hat{\omega}_t - \hat{\omega}_i \rangle + \langle g(x_i; \theta_0), \hat{\omega}_t - \hat{\omega}_i \rangle$$

where according to Lemma A.5, Lemma A.6, Lemma A.7 and Lemma A.8, $\epsilon(m)/2 = C_2^2 L^{-6}[\log m]^{-3/2} + C_3 \sqrt{mL} \cdot ((1 - \eta m)^{1/2} / (m \lambda) + m^{-2/3} \log m \cdot \sqrt{\lambda} / (m \lambda))$. The fourth inequality is based on Lemma A.9. The fifth inequality is based on Lemma A.10. By chaining all the inequalities, we have that with $N > \log \delta / (1 - \log(1 - \exp(-\beta^2)/(4\sqrt{\pi} \beta)))$, with probability at least $1 - \delta$, we have $\max_{n \in [N]} \sigma(f_{i,j}^{t(n)}) \geq \frac{1}{2}$.

Therefore, under event $E_t$, with the satisfied setting of $N$, based on the definition of $\omega_i^t$ in Section 3.2, we know that for document $i$ and $j$ at round $t$, $(i, j) \in \omega_i^t$ if and only if $\min_{n \in [N]} \sigma(f_{i,j}^{t(n)}) > \frac{1}{2}$. Then the probability of being uncertain can be bounded as,

$$\mathbb{P}((i, j) \in \omega_i^t) = 1 - \mathbb{P}(\min_{n \in [N]} \sigma(f_{i,j}^{t(n)}) > \frac{1}{2}) = 1 - (1 - \mathbb{P}(\sigma(f_{i,j}^{t(n)}) > \frac{1}{2}))^N$$

And according to the random matrix theory and Lemma 2 in (Jia et al., 2021), when $t > t''$, $\Delta_{\min}/c_{\mu} - CB - \epsilon(m) > 0$ which can be viewed as a constant $c\Delta_{\min}$. Therefore, we have the following inequalities,

$$\mathbb{P}(\sigma(f_{i,j}^{t(n)}) > 1/2) \geq \mathbb{P}(\sigma(f_{i,j}^{t(n)}) - \sigma(h_{i,j}) > 1/2 - \sigma(h_{i,j})) \geq \mathbb{P}(\sigma(f_{i,j}^{t(n)}) - \sigma(h_{i,j}) > -\Delta_{\min})$$

$$\geq \mathbb{P}(f_{i,j}^{t(n)} - h_{i,j} > -\Delta_\min/c_{\mu}) \geq \mathbb{P}(\langle g_i^{0, t}, \hat{\omega}_t - \hat{\omega}_i \rangle < \Delta_{\min}/c_{\mu} - \alpha_t \|g_i^{0, t}\|_A^{-1} - \epsilon(m))$$

$$\geq \mathbb{P}(U(g_i^{0, t}) < \Delta_{\min} - \alpha_t c_{\mu} \|g_i^{0, t}\|_A^{-1} - \epsilon(m)) \geq 1 - \exp(-\frac{c_{\mu}^2 \Delta_{\min}^2}{\pi^2 \cdot \|g_i^{0, t}\|_A^{-2}})$$

Therefore, similar to the analysis for the linear ranker, $\mathbb{P}((i, j) \in \omega_i^t) \leq \min\left\{1, \frac{2N \nu^2 \|g_i^{0, t}\|^2}{c_{\mu}^2 \Delta_{\min}^2} \right\}$. This completes the proof.
A.4.2 Proof of Lemma A.9

Proof. To obtain the solution $\hat{\omega}_i^0$ for Eq (A.4), we first take gradient of the loss function with respect to the parameter. Therefore, $\hat{\omega}_i^0$ is the solution for the following equation.

$$\sum_{s=1}^{t-1} \sum_{(i,j) \in \Omega_s} \sigma(s_{ij} \tilde{g}_{ij} \omega) - y_{ij} \tilde{g}_{ij}^0 + m \lambda \omega = 0$$

Let $q_t(\omega) = \sum_{s=1}^{t-1} \sum_{(i,j) \in \Omega_s} \sigma(s_{ij} \tilde{g}_{ij} \omega) / m + \lambda \omega$. We know that $q_t(\hat{\omega}_i^0) = \sum_{s=1}^{t-1} \sum_{(i,j) \in \Omega_s} y_{ij} \tilde{g}_{ij}^0$. Then we can quantify the uncertainty caused by the observation noise for any document pair $i$ and $j$ with the pairwise gradient as $g$.

$$|g^T(\hat{\omega}_i - \omega^*)| = |g^T Q^{-1}_t(q_t(\hat{\omega}_i) - q_t(\omega^*))| \leq \frac{1}{c_\mu} |g^T A^{-1}_t(q_t(\hat{\omega}_i) - q_t(\omega^*))| \leq \frac{1}{c_\mu} \|g^T A^{-1}_t(q_t(\hat{\omega}_i) - q_t(\omega^*))\|_{A^{-1}_t}$$

The first equality stands with $Q_t = \sum_{s=1}^{t-1} \sigma(g^T \omega) g_{ij}^0 / m + \lambda I$. According to the definition of $A_t$ and $c_\mu$, it is easy to verify that $Q^{-1}_t \leq \frac{1}{c_\mu} A^{-1}_t$. For the second component of the last equation, we can further bound it by

$$\|\sum_{s=1}^{t-1} \xi_{ij} \tilde{g}_{ij}^0 / \sqrt{m}\|_{A^{-1}_t} = \sqrt{\log \frac{\det(A_t)}{\sigma^2 \det(\lambda I)}} + \sqrt{\lambda S}$$

It is easy to have the first and second inequalities. The third inequality holds according to Theorem 2 in (Abbasi-Yadkori et al., 2011), and the upper bound $S$ defined in Lemma A.4. This completes the proof. \qed

A.4.3 Proof of Lemma A.10

Proof. According to the definition of $q_t$ in Section A.4.2, with $\hat{\omega}_i^0$ as the solution of Eq (A.5), we know that $q_t(\hat{\omega}_i^0) = \sum_{s=1}^{t-1} \sum_{(i,j) \in \Omega_s} y_{ij}^0 \gamma_{ij}^s \tilde{g}_{ij}^0 / m - \sqrt{\lambda / m} \sum_{l=1}^p \gamma_l \tilde{e}_l$. Therefore, for any document pair $i$ and $j$ with the pairwise gradient $g$, we know that,

$$g^T(\tilde{\omega}_i - \omega_i) = g^T Q^{-1}_t(q_t(\hat{\omega}_i) - q_t(\omega_i)) \leq \frac{1}{c_\mu} g^T A^{-1}_t \left( \sum_{s=1}^{t-1} \sum_{(i,j) \in \Omega_s} \gamma_{ij}^s \tilde{g}_{ij}^0 / m - \sqrt{\lambda / m} \sum_{l=1}^p \gamma_l \tilde{e}_l \right)$$

Let $U(g) = (g / \sqrt{m})^T A^{-1}_t \left( \sum_{s=1}^{t-1} \sum_{(i,j) \in \Omega_s} \gamma_{ij}^s \tilde{g}_{ij}^0 / \sqrt{m} - \sqrt{\lambda} \sum_{l=1}^p \gamma_l \tilde{e}_l \right)$. According to the definition of $A_t$ and $Q_t$, it is easy to have $Q^{-1}_t \geq A^{-1}_t$. And we can have $g^T(\tilde{\omega}_i - \omega_i) \geq U(g)$. With $\gamma_{ij}^s \sim N(0, \nu^2)$, we have $\mathbb{E}[U(g)] = 0$. We have the variance of $U(g)$ as,

$$Var[U(g)] = \nu^2 \left( \frac{g^T A^{-1}_t \left( \sum_{s=1}^{t-1} \sum_{(i,j) \in \Omega_s} \gamma_{ij}^s \tilde{g}_{ij}^0 / \sqrt{m} + \lambda \sum_{l=1}^p \tilde{e}_l \tilde{e}_l^T \right) A^{-1}_t \frac{g}{\sqrt{m}} \right) = \nu^2 \|g / \sqrt{m}\|_{A^{-1}_t}^2$$

This completes the proof. \qed