Efficient Pointwise-Pairwise Learning-to-Rank for News Recommendation

Nithish Kannen[⋆‡1]**, Yao Ma**[⋆1]**, Gerrit J.J. van den Burg**[1]**, Jean Baptiste Faddoul**[†2]

[1]Amazon, [2]Zalando SE

nithishkannen@gmail.com,
{yaoom, gvdburg}@amazon.com,
jean.baptiste.faddoul@zalando.de

Abstract

News recommendation is a challenging task that involves personalization based on the interaction history and preferences of each user. Recent works have leveraged the power of pre-trained language models (PLMs) to directly rank news items by using inference approaches that predominately fall into three categories: pointwise, pairwise, and listwise learning-to-rank. While pointwise methods offer linear inference complexity, they fail to capture crucial comparative information between items that is more effective for ranking tasks. Conversely, pairwise and listwise approaches excel at incorporating these comparisons but suffer from practical limitations: pairwise approaches are either computationally expensive or lack theoretical guarantees, and listwise methods often perform poorly in practice. In this paper, we propose a novel framework for PLM-based news recommendation that integrates both pointwise relevance prediction and pairwise comparisons in a scalable manner. We present a rigorous theoretical analysis of our framework, establishing conditions under which our approach guarantees improved performance. Extensive experiments show that our approach outperforms the state-of-the-art methods on the MIND and Adressa news recommendation datasets.

1 Introduction

Online news services have become important platforms for a large population of users to stay informed. A massive number of news articles are generated and posted online every day, making it all the more important to personalize news recommendation for the users. The text-rich nature of news articles makes them particularly well-suited for the application of pre-trained language models (PLMs) (Wu et al., 2021). Despite the impressive advancements in large language models (LLMs), practical constraints in real-world news recommendation systems necessitate the use of more computationally efficient, and often smaller, PLMs.

One way to approach news recommendation using PLMs is through pointwise ranking. This method predicts a relevance score for each candidate based on the user's previously clicked items. While this approach is scalable, the candidate scores are obtained *independently* without comparing the relative usefulness of a candidate with regards to its competitors. Naturally, the task of recommendation requires *comparing* candidates with each other. This intuition is supported by empirical results that demonstrate pairwise/listwise approaches outperform pointwise ones (Li, 2011). Theoretically, listwise approaches are expected to perform the best as ranking is a list-level task (Cao et al., 2007; Liu et al., 2023b). However, as shown by Qin et al. (2024), listwise recommendation performs poorly when paired with PLMs for two main reasons: 1) PLMs often generate conflicting or useless outputs when provided large quantities of information, and 2) PLMs have a fixed maximum input prompt length that may be exceeded while attempting to encode all available candidate items. On the other hand, pairwise approaches leverage the comparison between items without the downsides of listwise approaches described above. However, they come with higher computational complexity compared to pointwise approaches.

There is a body of relevant work on pairwise approaches for text-based recommendation tasks: Qin et al. (2024) propose *pairwise prompting* for ranking with PLMs by performing bubble-sort at inference time. Pradeep et al. (2021) propose a two-stage approach that first ranks using pointwise scores and subsequently modifies the rankings by computing aggregation scores from pairwise comparisons. However, both methods are computation-

---

ally expensive and scale poorly in practice due to the requirement for $O(n^2)$ comparisons. To optimize pairwise approaches, Gienapp et al. (2022) propose to sparsify the number of pairwise comparisons by random and skip-window sampling of pairs. While this improves scalability, it only compares several random aggregation strategies without theoretical guarantees.

In this work we propose a theoretically-guaranteed method that efficiently performs pairwise ranking. We summarize our main contributions as follows:

- A multi-task model jointly trained to perform both pointwise and pairwise predictions. We use a text-to-text approach where both are treated as classification tasks (prediction of pre-defined target words), hence aligning the tasks with the PLM pre-training objective.

- An efficient inference strategy in which we initialise with a ranking obtained from pointwise model scores and perform Right-To-Left (RTL) passes for pairwise reranking. That is, adjacently ranked elements are compared using the pairwise functionality starting from the rightmost and until the leftmost position. In this way, we perform only $n-1$ pairwise comparisons per RTL pass.

- A theoretical framework for this approach based on Markov chains. We derive testable conditions that can be used to verify the strategy is beneficial for a given ranking metric.

- Extensive experiments that show our approach outperforms the state of the art on the MIND and Adressa news recommendation datasets.

We emphasize that although our focus is on news recommendation, our proposed algorithm is directly applicable to any text-based task.

2 The Proposed Algorithm: GLIMPSE

The task of news recommendation in this paper refers to ranking a set of candidate news items $X = \{x_1, x_2, x_3, \ldots, x_K\}$ given the user's click history. The goal is to rank an item higher than others if it is preferred by the user. This goal implies both a pointwise relevance prediction task and a pairwise preference task: a user click on an item should be treated not only as an absolute judgment for relevancy, but also as a preference

judgement. Motivated by this, we propose the **GLIMPSE** (**G**enerative Tup**L**e-**WI**se Pro**MP**ting for New**S** recomm**E**ndation) algorithm for optimizing both objectives simultaneously. GLIMPSE is a general method for recommendation problems and works with any generative model.

We first describe the multi-task fine-tuning method of a generative model that aligns both the relevance prediction objective and the preference prediction objective into a single text generation task. By performing multi-task fine-tuning, we obtain a single model that can be used as a predictor for both objectives at the same time. To combine the tasks into a complete ranking method, we subsequently propose a novel inference strategy by initialising with pointwise relevance ranking and using unidirectional (right-to-left) adjacent element swaps with pairwise preference comparisons to obtain a final ranking (illustration in Figure 1).

2.1 Multi-task Fine-tuning for Ranking

The proposed multitask fine-tuning strategy is motivated by the fact that jointly optimizing ranking and classification objectives has been shown to achieve better performance than using ranking-only or classification-only objectives, especially when observations are limited (Sculley, 2010). The strategy aligns the two objectives into a single one, in our case through text generation.

Pointwise Relevance Prediction (Rel). The first task we consider is a classification task where the goal is to predict the relevance of a candidate item given the user history. This task looks at a *single* candidate item at a time. It involves predicting $y_{ij} \in \{0, 1\}$ for a candidate item x_i conditioned on user's click history $\mathcal{H}_j$. In practice, this is a classification task to classify if a candidate x_i is relevant ($y_{ij} = 1$) or not relevant ($y_{ij} = 0$) given history $\mathcal{H}_j$. To fine-tune the generative model on this task, we prompt the input text sequence to essentially answer the question, *Is the item a suitable recommendation for the user?* After fine-tuning, we obtain a relevance prediction function Rel which gives the probability that an item is relevant for a user with history $\mathcal{H}_j$, i.e.,

$$\text{Rel}(x_i \,|\, \mathcal{H}_j) := P(y_{ij} = 1 | x_i, \mathcal{H}_j). \quad (1)$$

In practice, we use the probability of the positive target token as the probability of relevance.

Pairwise Preference Prediction (Pref). In the second task, the model is asked which candidate it
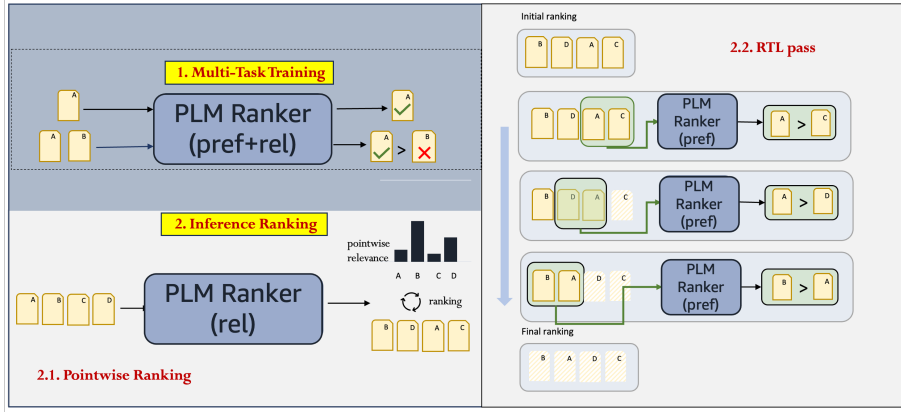
Figure 1: An illustration of the proposed framework. GLIMPSE consists of a multi-task training approach where the PLM is fine-tuned by considering both the relevance prediction and the pairwise preference tasks. During inference, the relevance predictions are used to obtain an initial pointwise ranking, which is subsequently improved by performing one or more right-to-left (RTL) passes using pairwise comparisons.

prefers when it is provided a pair of options. More specifically, given two candidate items x_a and x_b and user history $\mathcal{H}_j$, the model predicts the probability of the preference of one candidate over the other. This too can be framed as a text generation task, through the question: *Given Candidate A and Candidate B, which is a more suitable recommendation for the user?* After fine-tuning, we obtain a prediction of the probability that an item is preferred over another for a specific user, i.e.,

$$\text{Pref}((x_a, x_b)|\mathcal{H}_j) := P(x_a > x_b|\mathcal{H}_j). \quad (2)$$

Similar to the relevance prediction task, the probability of preferring one candidate over another is based on the probability of predicting the corresponding target tokens. To obtain $P(x_a > x_b \,|\, \mathcal{H}_j)$ from these token probabilities we leverage the Bradley-Terry realization (Bradley and Terry, 1952). Writing δ_a and δ_b for the probability of predicting the target words *Candidate A* and *Candidate B* respectively for x_a and x_b, we have:

$$P(x_a > x_b) = \frac{e^{\delta_a}}{e^{\delta_a} + e^{\delta_b}}. \quad (3)$$

Multi-task training. Similar to previous work (Su et al., 2022), we use discrete task-specific prompts to differentiate between the tasks. We cast both tasks into text-generation problems with task prompts and pre-defined target words. In the multi-task fine-tuning stage, the training sample is represented as a tuple $d = (s_t, x, y, \mathcal{H})$. Here s_t refers to the task-prompt with $t \in \{\text{Rel}, \text{Pref}\}$, and x refers to the input candidates. Note that x corresponds to a single item in the Rel task and a pair

of items in the Pref task. Table 6 in Appendix A.2 shows examples of task-prompts and target words. For the relevance task we sample an equal number of positive (clicks) and negative (no-click) samples, while for the preference task we construct a sample by picking one positive and one negative item from the same impression. An impression refers to user activity information containing user click history, including positives and negatives. The set of positives and negatives form the candidate set of an impression. During training, we shuffle and mix data points from different tasks for multi-task fine-tuning (Su et al., 2022). Thus the model is trained by maximizing the likelihood objective defined as

$$\mathcal{L}_\theta = -\sum_d \sum_{l=1}^{|y|} \log \, p_\theta(y^l|y^{<l}; s_t, x, \mathcal{H}), \quad (4)$$

where $|y|$ denotes the length of the target sequence y, y^l, $y^{<l}$ are the l-th token and tokens before l, and θ denotes the model parameters.

2.2 Aggregated Ranking Inference

The multi-task fine-tuned model can both predict a pointwise relevance score and perform pairwise preference comparison. Here we introduce a novel inference approach that incorporates both of these capabilities and leverages the different benefits of pointwise and pairwise predictions for learning to rank.

Rank Aggregation Strategy. We initialize the ranked list by predicting the relevance score $\text{Rel}(x^i)$ for each candidate item x_i and sorting by this score in descending order. Next we perform local refinement by applying the pairwise preference

scores to pairs of items in this sorted list. Specifically, we perform m RTL passes on the top-k ranked items in the sorted list. An RTL pass is defined as a single pass in the right to left direction where adjacent items are compared using the $\mathrm{Pref}(\cdot, \cdot)$ function starting from the rightmost to the leftmost item. Two adjacent items x^i and x^{i+1} are swapped if $\mathrm{Pref}(x^i, x^{i+1}) < \mathrm{Pref}(x^{i+1}, x^i)$. The number of RTL passes m and the top-k items considered in an RTL pass are hyperparameters of our inference approach.

The proposed RTL rank aggregation algorithm is guaranteed to achieve better ranking performance under certain conditions compared to the pointwise relevance ranking strategy (see Section 3). It is clear that one RTL pass of top-k elements consists of $k - 1$ comparisons, as shown in Figure 1. Note that when $k = K$, it leads to the classic Bubble-sort algorithm (Friend, 1956) except the sorting is done in a stochastic manner. The pointwise score prediction only takes $O(K)$ model calls, whereas a pairwise approach can take at most $O(K^2)$ and at best $O(K \log K)$ comparisons with Quicksort, making it an impractical ranking solution. Our approach provides a practical middle ground with $O(K)$ complexity.

3 Theoretical Analysis

In this section, we provide the theoretical analysis for the proposed RTL rank aggregation strategy. Let κ be a permutation of the indices $\{1, \ldots, K\}$ of candidate items $\{x_1, \ldots, x_K\}$. We denote $\mathrm{rank}(x_i | \kappa)$ as the rank of item x_i in the permutation κ and write $r^*(x_i | \mathcal{H}) \in \{0, 1\}$ for the ground-truth relevance of item x_i with history $\mathcal{H}$. We can then consider additive ranking metrics of the form:

$$\Delta(\kappa \,|\, \mathcal{H}, r^*) = \sum_{i=1}^{K} \lambda(\mathrm{rank}(x_i \,|\, \kappa)) \cdot r^*(x_i | \mathcal{H}),$$
(5)

where λ is a function over the rank (Agarwal et al., 2019). For example, the well-known discounted cumulative gain (DCG) metric (Järvelin and Kekäläinen, 2002) corresponds to $\lambda(v) = 1/\log(1 + v)$. For a stochastic algorithm $\mathcal{A}$ we are interested in the expectation of the ranking metric over the possible permutations κ, which we can write as $\mathbb{E}_{\mathcal{A}}(\Delta) = \sum_{\kappa} p(\kappa)\Delta(\kappa \,|\, \mathcal{H}, r^*)$, with $p(\kappa)$ the probability of obtaining the ranking κ from the algorithm for a particular user.

Because the relevance $r^*(\cdot)$ is a binary value (e.g., click or no click), distinct permutations κ can give rise to the same value of the ranking metric. Let $\mathbf{z} \in \{0, 1\}^K$ denote the binary vector of user relevance of items in ranking κ, such that $\mathbf{z} = [r^*(x_i | \mathcal{H}) : i \in \kappa]$. In this setting, the ranking metric becomes $\Delta(\mathbf{z} \,|\, \mathcal{H}) = \sum_{k=1}^{K} \lambda(k)z_k$, with z_k the k-th element of $\mathbf{z}$. By extension, we can write the expectation over the metric in terms of $\mathbf{z}$ as well, $\mathbb{E}_{\mathcal{A}}(\Delta) = \sum_{\mathbf{z}} p(\mathbf{z})\Delta(\mathbf{z} \,|\, \mathcal{H})$, where the summation is over all possible values of $\mathbf{z}$.

We can refine the expression for the expected ranking metric by considering our proposed two-stage inference approach consisting of an initial ranking of the candidates based on relevance prediction, and a subsequent refinement of that ranking through pairwise preferences. Let $\boldsymbol{\pi} \in \mathbb{R}^{2^K}$ denote the probability vector with elements $p(\mathbf{z})$ for all $\mathbf{z} \in \{0, 1\}^K$ that reflects the probability of obtaining $\mathbf{z}$ from pointwise inference. Then, the ranking refinement from pairwise comparisons can be modeled as a Markov chain with transition matrix $\mathbf{T} \in \mathbb{R}^{2^K \times 2^K}$, such that the product $\boldsymbol{\pi}' = \boldsymbol{\pi}\mathbf{T}$ is the probability vector of "states" after the transition induced by pairwise reranking. Specifically, the elements $T_{i,j}$ of the transition matrix $\mathbf{T}$ represent the probability of transitioning from a state $\mathbf{z}^i$ to a state $\mathbf{z}^j$. With a mild abuse of notation we can write $\mathbf{T}_{\mathbf{z}'}$ for the column of $\mathbf{T}$ corresponding to an outcome state $\mathbf{z}'$. Then, the expected value of the ranking metric after our two-stage inference process becomes

$$\mathbb{E}_{\mathcal{A}}(\Delta) = \sum_{\mathbf{z}'} \boldsymbol{\pi}\mathbf{T}_{\mathbf{z}'}\Delta(\mathbf{z}' \,|\, \mathcal{H}).$$
(6)

Repeated application of the pairwise refinement (i.e., multiple RTL passes) can be captured through typical Markov chain notation, $\mathbb{E}_{\mathcal{A}}(\Delta) = \sum_{\mathbf{z}'} \boldsymbol{\pi}\mathbf{T}^{\alpha}\mathbf{T}_{\mathbf{z}'}\Delta(\mathbf{z}' \,|\, \mathcal{H})$, for $\alpha \in \mathbb{Z}^+$ a non-negative integer. We next analyze the transition matrix to understand the conditions under which the RTL passes are beneficial.

3.1 Transition Matrix

The transition matrix $\mathbf{T}$ is parameterized by the output of the pairwise preference predictions. Since the model predicts the preference over any pair of items, we can define the probability of swapping or not swapping items with distinct user relevance values. Formally, for any pair of items (i, j), we can define the following probabilities:

$$\mu := P(x_i > x_j \,|\, r^*(x_i) = 1, r^*(x_j) = 0)$$
$$\nu := P(x_i > x_j \,|\, r^*(x_i) = 0, r^*(x_j) = 1).$$
(7)

Thus, μ is the probability that the model predicts that x_i is preferred over x_j and this *is* the case in the ground truth, while ν is the probability that the model predicts that x_i is preferred over x_j even though this *is not* the case in the ground truth. The converse probabilities $1-\nu$ and $1-\mu$ can be defined analogously.

For a specific inference strategy, we can derive the corresponding transition matrix using the number of **swap** and **no-swap** steps needed to transit between states. Mathematically, any element of the transition matrix $\mathbf{T}$ can be expressed in the following form $\mu^{\alpha_1}(1-\mu)^{\alpha_2}\nu^{\alpha_3}(1-\nu)^{\alpha_4}$. where $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ are the numbers of swap and no-swap steps needed to transit from a state to another.

Now, we show the transition matrix for the proposed RTL rank aggregation strategy. For simplicity we assume $\forall \mathbf{z}, \|\mathbf{z}\|_1 = 1$, which indicates there is a single relevant item. In this case, it is clear that the state space consists of K states $\mathbf{z}^1, ..., \mathbf{z}^K$, where $\mathbf{z}^1 = [1, 0, ..., 0], \mathbf{z}^2 = [0, 1, 0, ..., 0], ..., \mathbf{z}^K = [0, ..., 0, 1]$. Note that the Mean Reciprocal Rank (MRR) metric can be written by using $\lambda(v) = 1/v$ under this assumption. We denote this metric as Δ_M below. We can write the transition matrix $\mathbf{T}$ of the pairwise refinement, whose i, j-th element equals $P(\mathbf{z}^j|\mathbf{z}^i)$, as

$$
T_{i,j} = \begin{cases}
0 & (a) \\
\mu & (b) \\
(1-\mu)(1-\nu)\nu^{i-j} & (c) \\
(1-\nu)\nu^{K-j} & (d) \\
(1-\mu)\nu^{i-1} & (e) \\
\nu^{K-1} & (f),
\end{cases} \tag{8}
$$

where (a) if $j > i+1$ and $1 \leq i < K$, (b) if $j = i+1$ and $1 \leq i < K$, (c) if $1 < j \leq i$ and $1 < i < K$, (d) if $1 < j \leq i$ and $i = K$, (e) if $j = 1$ and $1 \leq i < K$, and (f) if $j = 1$ and $i = K$.

With the defined transition matrix $\mathbf{T}$, we have the following theorem (proof in Appendix A.1).

Theorem 3.1. *For any* $\boldsymbol{\pi} = [p(\mathbf{z}^1), p(\mathbf{z}^2), .., p(\mathbf{z}^K)]$ *obtained from a pointwise inference strategy, the RTL pairwise ranking refinement achieves positive gain in terms of the expected MRR,* $\mathbb{E}_A(\Delta_M) - \mathbb{E}_\pi(\Delta_M) > 0$*, when the pairwise inference satisfies*

$$
\mu < \frac{p(\mathbf{z}^i)}{p(\mathbf{z}^{i+1})}\nu, \quad \forall i = 1, ..., K-1. \tag{9}
$$

Corollary 3.1.1. *By applying the pairwise ranking refinement inference α times, the expected MRR*

achieves positive gain if

$$
\mu < \frac{\pi T_i^\alpha}{\pi T_{i+1}^\alpha}\nu, \qquad \forall i = 1, ..., K, \tag{10}
$$

where πT_i^α is the i-th element of vector πT^α.

Intuitively, Theorem 3.1 shows that the proposed two-stage inference achieves better performance when the pairwise inference model outperforms the pointwise inference model. Here, we also provide the analysis for the state distribution $\boldsymbol{\pi}$ obtained from the pointwise inference model. We assume the relevancy score $s = \text{Rel}(\cdot) \in [0, 1]$ predicted by the pointwise ranker follows two beta distributions, where the predicted score for positive and negative classes follow $\text{Beta}(\alpha_1, \beta_1)$ and $\text{Beta}(\alpha_2, \beta_2)$, respectively. For a given impression containing one positive item and $K - 1$ negative items, the probability $p(\mathbf{z}^k)$ can be written as

$$
\begin{aligned}
p(\mathbf{z}^k) = \binom{K-1}{k-1} \int_0^1 (1 - F(u; \alpha_2, \beta_2))^{k-1} \\
F(u; \alpha_2, \beta_2)^{K-k} f(u; \alpha_1, \beta_1) \mathrm{d}u,
\end{aligned} \tag{11}
$$

where $\binom{K-1}{k-1}$ denotes the binomial coefficient, and $f(\cdot)$ and $F(\cdot)$ are the probability density and cumulative distribution functions of the beta distribution. With the defined state probability, we have for all $k = 1, ..., K - 1$, $\frac{p(\mathbf{z}^k)}{p(\mathbf{z}^{k+1})} \leq \frac{p(\mathbf{z}^1)}{p(\mathbf{z}^2)}$. Together with Theorem 3.1, we can rewrite the condition for obtaining positive gain after pairwise refinement as $\mu \leq \nu \frac{p(\mathbf{z}^1)}{p(\mathbf{z}^2)}$. Though a closed form condition is hard to achieve, the ratio $p(\mathbf{z}^1)/p(\mathbf{z}^2)$ can be estimated through a numerical simulation with Eq. 11.

4 Experiments

In this section we conduct experiments to empirically validate our approach. We first describe a comparison with existing works. Next, we describe an experiment on various inference strategies, followed by an ablation study to better understand the effect of each component of our approach. Additional details about the experimental setup can be found in Appendix A.2.

4.1 Setup

Our experiments focus on two commonly-used datasets for news recommendation, MIND (Wu et al., 2020) and Adressa (Gulla et al., 2017). The Microsoft News Dataset (MIND) contains information regarding user sessions on a news aggregation website. User sessions are organized in "impressions", where an impression consists of an ordered

	MIND				Adressa			
Model	AUC	MRR	nDCG@5	nDCG@10	AUC	MRR	nDCG@5	nDCG@10
BERT-NPA$^{\diamond}$	67.56	31.94	35.34	41.73	56.21	29.44	26.98	32.67
BERT-LSTUR$^{\diamond}$	68.28	32.58	35.99	42.32	56.76	29.87	27.34	32.90
BERT-NRMS$^{\diamond}$	68.60	32.97	36.55	42.78	56.95	30.08	27.89	32.92
DIGAT	68.77	33.46	37.14	43.39	57.13	30.18	27.95	32.90
HDNR	68.23	32.61	36.10	42.29	57.43	30.09	28.34	34.11
Prompt4NR$^{\spadesuit}$	68.48	33.29	37.12	43.25	61.67	30.32	28.98	36.33
UniTRec	68.59	33.76	37.63	43.74	62.29	**31.90**	29.43	36.90
GLIMPSE$_{point}$	68.18	33.56	37.27	43.48	57.82	30.38	26.00	33.14
w/ pair (1 pass, top 2)	68.88	33.70	37.38	43.59	**67.24**	31.84	**30.98**	**39.33**
w/ pair (1 pass, top 3)	68.92	33.71	37.39	43.60	63.15	27.90	27.90	34.95
w/ pair (1 pass, top 5)	68.97	33.73	37.41	43.62	67.23	30.65	30.13	38.47
w/ pair (2 pass, top 5)	**69.14**	**33.88**	**37.66**	**43.88**	67.14	30.86	30.25	38.60

Table 1: Results on the MIND and Adressa datasets compared to baseline methods. All results on Adressa are reproduced using publicly-available code. For MIND, ♠ means that we reproduce the scores using the publicly-available code, and ◇ indicates that results are as reported in Zhang and Wang (2023). For other methods results are extracted from the respective papers.

list of news articles and the user click behavior for those articles. To support fair comparison to previous work (Zhang and Wang, 2023; Bi et al., 2022; Qi et al., 2021), we use the MIND-Small subset of MIND for training, which consists of click-behavior for 50,000 users. For evaluation we use the MIND test set, which is the same for both MIND-Small as for MIND-Large. The second dataset we use is Adressa (Gulla et al., 2017), which is sourced from traffic on the Adresseavisen news website in Norway. As in Yi et al. (2021), we build data splits using one week of data. Our training split uses the first six days of click data, where the first five days serve as historical clicks and the sixth day provides impression and click data. The validation split consists of a random 20% sample from the last day, with the remaining 80% used for the test split. Since Adressa lacks impressions with negative samples, we randomly select 20 news articles to construct our impression data. See Table 5 in the appendix for summary statistics.

We compare our approach to baselines from the literature, including BERT-based news recommendation approaches from Wu et al. (2021), which are constructed by replacing traditional text-encoding methods with BERT (Devlin et al., 2019). These methods include BERT-NPA based on Wu et al. (2019a), BERT-LSTUR based on An et al. (2019), and BERT-NRMS based on Wu et al. (2019b). We furthermore compare to DIGAT (Mao et al., 2022), HDNR (Wang et al., 2023), Prompt4NR (Zhang and Wang, 2023), and UniTrec (Mao et al., 2023). For Prompt4NR, we only compare the performance

with single model based variants. The ensembling version of Prompt4NR is not considered in this experiment for a fair comparison among approaches. In line with previous work we ignore baseline methods that use auxiliary inputs such as a topic label.

For GLIMPSE we use an encoder-decoder Flan-T5 model (Chung et al., 2024) from the Hugging-Face library‡ and fine-tune the model using the mixture-of-tasks approach for 4 epochs. We use a learning rate of 10^{-5} with a linear scheduler and perform early stopping. We use the sum of the pointwise and pairwise validation accuracies to select the best checkpoint. The results reported below are using the *base* version of Flan-T5 which consists of 200 million parameters. During inference, we save the pointwise model along with its scores to subsequently perform the RTL pass. We use the same hyperparameters for both datasets and measure model performance.

4.2 Results

Table 1 reports the results for existing methods and several variants of GLIMPSE. We show the performance of the GLIMPSE$_{point}$ method (which refers to our model when trained in a multi-task fashion but without the RTL reranking pass applied during inference), as well as for GLIMPSE with the RTL pass applied on the top k items in the pointwise-ranked list. From the results on the MIND dataset, we can see that our best performing strategy improves over the strongest baseline, UniTRec, by 0.80% on AUC, 0.36% on MRR,

---

‡https://huggingface.co/

Model	Inference Strategy	AUC	MRR	nDCG@5	nDCG@10	Complexity
Pairwise	Pairwise	65.99	30.76	33.64	39.67	$O(n^2)$
GLIMPSE	Pointwise	67.90	31.65	34.92	41.59	$O(n)$
GLIMPSE	Box filling	67.98	32.05	35.13	41.60	$O(n^2)$
GLIMPSE	BubbleSort (random init)	65.03	31.88	34.75	41.58	$O(n^2)$
GLIMPSE	BubbleSort (point init)	68.08	32.13	35.04	41.77	$O(n^2)$
GLIMPSE	N-Window	67.68	32.22	35.10	41.86	$O(mn)$
GLIMPSE	S-window	67.23	31.72	34.85	41.40	$O(mn)$
GLIMPSE	1 RTL pass on top-3	68.54	32.41	35.48	42.16	$O(n)$
GLIMPSE	1 RTL pass on top-5	**68.59**	**32.60**	**35.60**	**42.27**	$O(n)$
GLIMPSE	1 RTL pass on top-10	68.32	32.19	35.18	41.96	$O(n)$
GLIMPSE	2 RTL passes on top-5	68.40	32.34	35.43	42.11	$O(n)$

Table 2: Comparison of different inference strategies on 10% of MIND test set. For the N-Window and S-window strategies the notation corresponds to Gienapp et al. (2022).

0.08% on nDCG@5 and 0.32% on nDCG@10. For Adressa the improvements over baseline methods are even greater, with GLIMPSE improving on the strongest baseline by 7.95% on AUC, 5.27% on nDCG@5, and 6.59% on nDCG@10, while matching performance on MRR. These results emphasize the benefit and efficiency of pairwise comparisons in our inference strategy, with performance improvements against pointwise inference resulting from only 8 pairwise comparisons on MIND and only 1 on Adressa.

4.3 Inference Strategies

Numerous inference strategies can be used to combine the predictions of a pointwise initial ranking and a pairwise comparison model. We analyze several in terms of performance and computational complexity. To evaluate inference strategies empirically we use a subset of 10% of the MIND test set to allow comparison to strategies that are quadratic in the number of items. We include the Prompt4NR method (Zhang and Wang, 2023) for reference.

Table 2 shows the results of these experiments. We use the multi-task GLIMPSE model as a baseline and only vary the inference strategy. The "pointwise" inference strategy refers to only using the relevance prediction for an item (see Section 2.1). Next, the "box filling" approach relies on performing all-pair comparisons to fill a table with pairwise preference scores, and ranking the items based on the marginals of this table. This strategy simulates the method by Jiang et al. (2023a). We also compare to the traditional Bubblesort algorithm (Friend, 1956), using both random and pointwise-inference initializations. We also compare to the "neighborhood-window" (N-Window) and "skip-window" (S-Window) methods proposed in Gienapp et al. (2022). These methods use a

moving window of items as batches for pairwise reranking. Finally, we compare to four variants of our approach, using either one or two RTL passes on the top-k items in the list ranked by pointwise scoring. We observe that the GLIMPSE approach with one RTL pass on the top-5 items ranked by pointwise scoring outperforms the other inference strategies. See Appendix A.3.4 for additional results on running time.

4.4 Ablation Study

To study how our framework performs with varied strengths of the pointwise model we train a weak pointwise model with a subset of pointwise datapoints and subsequently perform RTL passes using a strong pairwise model. For the weak pointwise model, we retain 5% and 10% respectively of the pointwise datapoints and use the weak model for the initial pointwise ranking. We then use a strong pairwise model trained on the complete pairwise data for RTL passes. The results of this experiment are reported in Table 3. This shows that with a 5% pointwise model, performing a single RTL pass results in a relative increase of 2.96% on AUC, 12.91% on MRR, 9.60% on nDCG@5 and 7.85% on nDCG@10, while performing two RTL passes results in an increase of 3.59% on AUC, 14.74% on MRR, 11.06% on nDCG@5, and 9.01% on nDCG@10. On the 10% pointwise model a single pass shows relative improvement of 1.70% on AUC, 5.02% on MRR, 5.43% on nDCG@5, and 5.54% on nDCG@10. Thus we observe that the performance gains from pairwise reranking increase as the pointwise model gets weaker. Note that the gains obtained from RTL passes in this experiment with weak pointwise models are much higher than those in Table 1, confirming the above observation. Figure 2 shows the frequency distribution of the
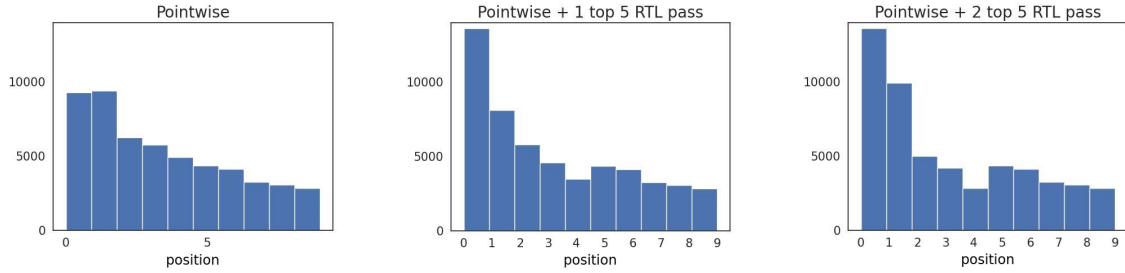
Figure 2: Comparison of the distribution shift after RTL passes. The figures show how often the first positive label is at each position. The figure on the left shows the distribution obtained from a pointwise model using 5% of the data (see main text), the figure in the middle is based on the same weak pointwise model, but uses an RTL pass on the top-5 items. It can be seen that RTL passes progressively move the first positive label to the front of the ranked list.

first occurrence of positive label in the predicted rankings at different inference stages. After applying RTL passes, we see a shift in the histogram towards the left side, indicating the positive labels are being pushed towards the top positions. Finally, to establish the benefit of multi-task training for pointwise ranking, we compare the performance of pointwise ranking with and without the pairwise comparison task. The results in Table 4 show that without the multi-task training there is a significant performance drop, especially in the ranking metrics MRR (-1.52%), nDCG@5 (-2.17%) and nDCG@10 (-1.31%). This confirms our argument that the task of recommendation naturally benefits from comparisons between candidates.

5 Related Work

In this section, we focus primarily on news recommendation approaches that leverage LLMs, as well as the pairwise ranking literature.

5.1 LLM-based News Recommendation

Prompt4NR (Zhang and Wang, 2023) is a recent baseline for news recommendation on the MIND dataset. The paper proposes continuous and discrete prompts to provide history and treat the task of click prediction as a mask-filling task. Xiao et al. (2023) propose a multi-task pre-training approach for news recommendation. Jiang et al. (2023b) propose RCENR, an explainable model that generates user or news subgraphs to enhance recommendation and extend the dimensions and diversity of reasoning. Wang et al. (2023) introduce HDNR, a hyperboloid model with exponential growth capacity to conduct user and news modeling. UniTrec (Mao et al., 2023) is a unified generative framework for three text-based recommendation tasks, including

news recommendation. It leverages candidate perplexity and discriminative scores to perform final pointwise ranking. Yu et al. (2022) propose Tiny-Newsrec, a self-supervised domain-specific post-training method to address the domain shift problem from pre-training tasks to downstream news recommendation. DebiasGAN (Wu et al., 2022a) alleviates position bias via adversarial learning by modelling the personalized effect of position bias on click behavior to estimate unbiased click scores. DIGAT (Mao et al., 2022) is a dual-interactive graph attention network to model user and news graph channels. Li et al. (2022) introduce MINER, a poly-attention scheme to learn multiple interest vectors for each user to effectively model different aspects of user interest. MTRec (Bi et al., 2022) is a multi-task method to incorporate the multi-field information in order to enhance the news encoding capabilities. Wu et al. (2022b) propose UniRec, a unified method for recall and ranking in news recommendation. Gong and Zhu (2022) propose a framework that leverages both positive and negative feedback. DRPN (Hu et al., 2022) de-noises both positive and negative implicit feedback to simulate noisy real world use-cases.

5.2 Pairwise Comparisons for Ranking

Pairwise ranking has been a long-standing approach ever since the method of ranking using paired comparisons was introduced by Bradley and Terry (1952). RankNet (Burges et al., 2005) is a gradient descent method for pairwise ranking. LambdaRank (Burges et al., 2006) builds on RankNet by incorporating IR measures (e.g., nDCG) into the derivative of the cost function. LambdaRankMart (Burges, 2010) is a boosted version of LambdaRank. PiRank (Swezey et al., 2021) adds new class of differentiable surrogate functions

Model	AUC	MRR	nDCG@5	nDCG@10
GLIMPSE$_{point}$ (5%)	56.99	25.71	27.48	33.60
w/ pair (1pass, top5)	58.68	29.03	30.12	36.24
w/ pair (2pass, top5)	59.04	29.50	30.52	36.63
GLIMPSE$_{point}$ (10%)	65.61	31.02	34.05	40.38
w/ pair (1pass, top5)	66.73	32.58	35.90	42.62

Table 3: Ablation study comparing a weak pointwise model (trained on a 5% or 10% subset of pointwise data) and a strong pairwise model (trained on all pairwise data), using the MIND dataset.

Model	AUC	MRR	nDCG@5	nDCG@10
GLIMPSE$_{point}$	68.18	33.56	37.27	43.48
pointwise	68.63	33.05	36.56	42.91

Table 4: Results of an ablation study to illustrate the benefit of multi-task training for pointwise ranking. The effect of multi-task training is especially noticeable in the ranking metrics.

for ranking. Heckel et al. (2018) propose an active learning approach to select pairs for comparisons that results in an approximate rank with logarithmic complexity. Gienapp et al. (2022) propose to sparsify the number of pairwise comparisons using next M window and skip K window strategies. This way the number of comparisons is less than quadratic and hence achieve a middle ground. Qin et al. (2024) propose *Pairwise Prompting* to perform ranking by sorting items from pairwise comparisons and compare listwise and pairwise strategies and show that bubble sort with pairwise comparisons achieves best results. Hou et al. (2024) formalize the recommendation problem as a conditional ranking task and adopt recency-focused in-context prompting and candidate generation algorithms before directly performing listwise ranking. Pradeep et al. (2021) address the pipelined recommendation problem with a candidate retriever followed by a 'mono' step where they rank using pointwise probability and a subsequent 'duo' step where they use pairwise comparisons for all-pairs. They test different aggregation functions for the all-pairs pairwise scores to re-rank the candidates. LLM Blender (Jiang et al., 2023a) is a box-filling strategy with a pairwise ranker for aggregating pairwise scores for ranking. Dai et al. (2023) use listwise, pairwise, and pointwise prompting on ChatGPT by choosing a fixed candidate set for experiments. Liusie et al. (2024) use pairwise comparative assessment for NLG evaluation.

6 Conclusion

We proposed GLIMPSE, an algorithm for personalized recommendation that leverages PLMs alongside a novel inference strategy to efficiently combine pointwise relevance prediction and pairwise comparison. We presented a multi-task fine-tuning approach to facilitate this inference and conducted extensive experiments on real-world datasets, demonstrating that GLIMPSE outperforms state-of-the-art methods on the MIND and Adressa datasets. Moreover, we provided a rigorous theoretical analysis of our proposed approach and derived conditions under which the RTL re-ranking pass is beneficial. Our work underscores the potential of leveraging language models for news recommendation, emphasizing both performance and real-world efficiency.

7 Limitations

The proposed algorithm is proved to achieve better performance than pointwise ranking approaches. However, it's important to note that this improvement may be modest when dealing with an extensive dataset used for training the pointwise relevance prediction model. If the pointwise model achieves sufficient accuracy, incorporating a pairwise preference model through the RTL rank aggregation strategy might offer limited additional value. Our theoretical findings support this observation, indicating that the gains could even turn negative should the state distribution derived from pointwise predictions fail to meet the defined conditions. While GLIMPSE generally improves ranking performance, its effectiveness hinges on both the size of the training set and the precision of the pointwise relevance prediction model.

References

Aman Agarwal, Kenta Takatsu, Ivan Zaitsev, and Thorsten Joachims. 2019. A general framework for counterfactual learning-to-rank. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 5–14.

Mingxiao An, Fangzhao Wu, Chuhan Wu, Kun Zhang, Zheng Liu, and Xing Xie. 2019. Neural news recommendation with long- and short-term user representations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 336–345, Florence, Italy. Association for Computational Linguistics.

Qiwei Bi, Jian Li, Lifeng Shang, Xin Jiang, Qun Liu, and Hanfang Yang. 2022. MTRec: Multi-task learning over BERT for news recommendation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2663–2669, Dublin, Ireland. Association for Computational Linguistics.

Ralph Allan Bradley and Milton E. Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39:324.

Christopher J. C. Burges. 2010. From RankNet to LambdaRank to LambdaMART: An overview. Technical Report MSR-TR-2010-82, Microsoft Research.

Christopher J. C. Burges, Robert Ragno, and Quoc Viet Le. 2006. Learning to rank with nonsmooth cost functions. In *Proceedings of the 19th International Conference on Neural Information Processing Systems*, NIPS'06, page 193–200, Cambridge, MA, USA. MIT Press.

Christopher J. C. Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, page 89–96, New York, NY, USA. Association for Computing Machinery.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: From pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, page 129–136, New York, NY, USA. Association for Computing Machinery.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering ChatGPT's capabilities in recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems*. ACM.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Edward H. Friend. 1956. Sorting on electronic computer systems. *J. ACM*, 3(3):134–168.

Lukas Gienapp, Maik Fröbe, Matthias Hagen, and Martin Potthast. 2022. Sparse pairwise re-ranking with pre-trained transformers. In *Proceedings of the 2022 ACM SIGIR International Conference on Theory of Information Retrieval*. ACM.

Shansan Gong and Kenny Q. Zhu. 2022. Positive, negative and neutral: Modeling implicit feedback in session-based news recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 1185–1195, New York, NY, USA. Association for Computing Machinery.

Jon Atle Gulla, Lemei Zhang, Peng Liu, Özlem Özgöbek, and Xiaomeng Su. 2017. The Adressa dataset for news recommendation. In *Proceedings of the international conference on web intelligence*, pages 1042–1048.

Reinhard Heckel, Max Simchowitz, Kannan Ramchandran, and Martin Wainwright. 2018. Approximate ranking from pairwise comparisons. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 1057–1066. PMLR.

Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2024. Large language models are zero-shot rankers for recommender systems. In *European Conference on Information Retrieval*, pages 364–381. Springer.

Yunfan Hu, Zhaopeng Qiu, and Xian Wu. 2022. Denoising neural network for news recommendation with positive and negative implicit feedback. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2320–2329, Seattle, United States. Association for Computational Linguistics.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023a. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, Toronto, Canada. Association for Computational Linguistics.

Hao Jiang, Chuanzhen Li, Juanjuan Cai, and Jingling Wang. 2023b. RCENR: A reinforced and contrastive heterogeneous network reasoning model for explainable news recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 1710–1720, New York, NY, USA. Association for Computing Machinery.

Hang Li. 2011. *Learning to Rank for Information Retrieval and Natural Language Processing, Second Edition*, volume 4. Morgan & Claypool Publishers.

Jian Li, Jieming Zhu, Qiwei Bi, Guohao Cai, Lifeng Shang, Zhenhua Dong, Xin Jiang, and Qun Liu. 2022. MINER: Multi-interest matching network for news recommendation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 343–352, Dublin, Ireland. Association for Computational Linguistics.

Junling Liu, Chao Liu, Peilin Zhou, Renjie Lv, Kang Zhou, and Yan Zhang. 2023a. Is ChatGPT a good recommender? A preliminary study. *arXiv preprint arXiv:2304.10149*.

Shuchang Liu, Qingpeng Cai, Zhankui He, Bowen Sun, Julian McAuley, Done Zheng, Peng Jiang, and Kun Gai. 2023b. Generative flow network for listwise recommendation. *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Adian Liusie, Potsawee Manakul, and Mark Gales. 2024. LLM comparative assessment: Zero-shot NLG evaluation through pairwise comparisons using large language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 139–151, St. Julian's, Malta. Association for Computational Linguistics.

Zhiming Mao, Jian Li, Hongru Wang, Xingshan Zeng, and Kam-Fai Wong. 2022. DIGAT: Modeling news recommendation with dual-graph interaction. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6595–6607, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Zhiming Mao, Huimin Wang, Yiming Du, and Kam-Fai Wong. 2023. UniTRec: A unified text-to-text transformer and joint contrastive learning framework for text-based recommendation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1160–1170, Toronto, Canada. Association for Computational Linguistics.

Ronak Pradeep, Rodrigo Nogueira, and Jimmy Lin. 2021. The expando-mono-duo design pattern for text ranking with pretrained sequence-to-sequence models. *arXiv preprint arXiv:2101.05667*.

Fanchao Qi, Yanhui Yang, Jing Yi, Zhili Cheng, Zhiyuan Liu, and Maosong Sun. 2022. QuoteR: A benchmark of quote recommendation for writing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 336–348, Dublin, Ireland. Association for Computational Linguistics.

Tao Qi, Fangzhao Wu, Chuhan Wu, Peiru Yang, Yang Yu, Xing Xie, and Yongfeng Huang. 2021. HieRec: Hierarchical user interest modeling for personalized news recommendation. *arXiv preprint arXiv:2106.04408*.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and Michael Bendersky. 2024. Large language models are effective text rankers with pairwise ranking prompting. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1504–1518, Mexico City, Mexico. Association for Computational Linguistics.

D. Sculley. 2010. Combined regression and ranking. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, page 979–988, New York, NY, USA. Association for Computing Machinery.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2022. Multi-task pre-training for plug-and-play task-oriented dialogue system. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4661–4676, Dublin, Ireland. Association for Computational Linguistics.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 1441–1450, New York, NY, USA. Association for Computing Machinery.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT good at search? Investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937, Singapore. Association for Computational Linguistics.

Robin Swezey, Aditya Grover, Bruno Charron, and Stefano Ermon. 2021. PiRank: Scalable learning to rank via differentiable sorting. *Advances in Neural Information Processing Systems*, 34:21644–21654.

Lingzhi Wang, Jing Li, Xingshan Zeng, Haisong Zhang, and Kam-Fai Wong. 2020. Continuity of topic, interaction, and query: Learning to quote in online conversations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6640–6650, Online. Association for Computational Linguistics.

Shicheng Wang, Shu Guo, Lihong Wang, Tingwen Liu, and Hongbo Xu. 2023. HDNR: A hyperbolic-based debiased approach for personalized news recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 259–268, New York, NY, USA. Association for Computing Machinery.

Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019a. NPA: neural news recommendation with personalized attention. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2576–2584.

Chuhan Wu, Fangzhao Wu, Suyu Ge, Tao Qi, Yongfeng Huang, and Xing Xie. 2019b. Neural news recommendation with multi-head self-attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6389–6394, Hong Kong, China. Association for Computational Linguistics.

Chuhan Wu, Fangzhao Wu, Xiangnan He, and Yongfeng Huang. 2022a. DebiasGAN: Eliminating position bias in news recommendation with adversarial learning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2933–2938, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021. Empowering news recommendation with pre-trained language models. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 1652–1656, New York, NY, USA. Association for Computing Machinery.

Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2022b. Two birds with one stone: Unified model learning for both recall and ranking in news recommendation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3474–3480, Dublin, Ireland. Association for Computational Linguistics.

Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, and Ming Zhou. 2020. MIND: A large-scale dataset for news recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3597–3606, Online. Association for Computational Linguistics.

Xiongfeng Xiao, Qing Li, Songlin Liu, and Kun Zhou. 2023. Improving news recommendation via bottlenecked multi-task pre-training. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 2082–2086, New York, NY, USA. Association for Computing Machinery.

Jingwei Yi, Fangzhao Wu, Chuhan Wu, Ruixuan Liu, Guangzhong Sun, and Xing Xie. 2021. Efficient-FedRec: Efficient federated learning framework for privacy-preserving news recommendation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2814–2824, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yang Yu, Fangzhao Wu, Chuhan Wu, Jingwei Yi, and Qi Liu. 2022. Tiny-NewsRec: Effective and efficient PLM-based news recommendation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5478–5489, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Qi Zhang, Jingjie Li, Qinglin Jia, Chuyuan Wang, Jieming Zhu, Zhaowei Wang, and Xiuqiang He. 2021. Unbert: User-news matching BERT for news recommendation. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3356–3362. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Zizhuo Zhang and Bang Wang. 2023. Prompt learning for news recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 227–237, New York, NY, USA. Association for Computing Machinery.

A Appendix

A.1 Proof of Theorem 3.1

In this section, we present the proof of Theorem 3.1. Recall that in our analysis we use Markov chain theory to analyze the outcomes from both the pointwise and pairwise stages within the inference strategy. We construct a Markov chain on a discrete state space encompassing all permutations of the ranking results. We define an initial probability distribution over the state space based on the stochastic ranking results generated by the pointwise inference stage. Through pairwise refinement the distribution over states shifts as stochastic pairwise swaps are applied by the pairwise inference stage. Thus, pairwise inference provides a transition probability kernel over the state space. By Theorem 3.1 we have that if the induced transition probability matrix satisfies certain conditions, we are guaranteed to achieve positive gain from two-stage inference. Below we present the proof of this theorem.

The gain obtained after our two-stage inference process compared to the outcome directly from the pointwise inference can be written as

$$
\begin{aligned}
G &= \mathbb{E}_{\mathcal{A}}(\Delta) - \mathbb{E}_{\boldsymbol{\pi}}(\Delta) = \sum_{\mathbf{z}'} \boldsymbol{\pi} \mathbf{T}_{\mathbf{z}'} \Delta(\mathbf{z}' \mid \mathcal{H}) \\
&\quad - \sum_{\mathbf{z}'} \boldsymbol{\pi} \Delta(\mathbf{z}' \mid \mathcal{H}) \\
&= \boldsymbol{\pi}(\mathbf{T} - \mathbf{I})\Delta^T,
\end{aligned}
\tag{12}
$$

where $\Delta = [\Delta(\mathbf{z}_1 \mid \mathcal{H}), \Delta(\mathbf{z}_2 \mid \mathcal{H}), ..., \Delta(\mathbf{z}_K \mid \mathcal{H})]^T$ and $\mathbf{I}$ is the identity matrix. Without loss of generality, we only provide the proof with MRR as the ranking metric for simplicity, i.e., $\Delta = [1, \frac{1}{2}, ..., \frac{1}{K}]^T$. Note that the proof in this section applies to any non-increasing ranking metric.

In order to analyze the gain defined above, we re-write the matrix $\mathbf{T} - \mathbf{I}$ by plugging in the definition of $\mathbf{T}$ shown in Eq. 8 as

$$
\begin{bmatrix}
-\mu & (1-\mu)\nu & (1-\mu)\nu^2 & ... \\
\mu & -\mu-\nu+\mu\nu & (1-\mu)(1-\nu)\nu & ... \\
0 & \mu & -\mu-\nu+\mu\nu & ... \\
\vdots & \vdots & \vdots & ... \\
0 & 0 & 0 & ...
\end{bmatrix}
$$

$$
\left.
\begin{matrix}
(1-\mu)\nu^{K-2} & \nu^{K-1} \\
(1-\mu)(1-\nu)\nu^{K-3} & (1-\nu)\nu^{K-2} \\
(1-\mu)(1-\nu)\nu^{K-4} & (1-\nu)\nu^{K-3} \\
\vdots & \vdots \\
\mu & -\nu
\end{matrix}
\right]. \tag{13}
$$

The matrix $\mathbf{T} - \mathbf{I}$ is an upper Hessenberg matrix whose diagonal elements are negative. The sub-diagonal elements are always equal to μ. As the ranking metric is a non-increasing function, $\mathbf{T} - \mathbf{I}$ will lead to positive gain when the diagonals and sub-diagonal are small enough. Formally, we can rewrite Eq. 12 by following element-wise matrix multiplication as

$$
\begin{aligned}
G &= -\mu\pi_1/2 + \sum_{i=2}^{K-1} \pi_i \big[(1-\mu)\nu^{(i-1)} \\
&\quad + (1-\mu)(1-\nu)\sum_{j=1}^{i-2} \tfrac{\nu^j}{i-j} \\
&\quad + \mu/(i+1) + (\mu\nu - \mu - \nu)/i \big] \\
&\quad + \pi_K \big(\nu^{K-1} - \nu/K \\
&\quad + (1-\nu)\sum_{i=2}^{K-1} \nu^{K-i}/i \big).
\end{aligned}
$$

It is clear that G is a bi-variate $(K-1)$-th degree polynomial function in terms of μ and ν. We decompose G into $K-1$ components $G = G_1 + G_2 + ... + G_{K-1}$ whose kth component is the sum of all order k terms in G. We start with the sum of all first order terms G_1,

$$
\begin{aligned}
G_1 &= -\tfrac{\pi_1 \mu}{2} + \pi_2 \left(\nu + \tfrac{\mu}{3} - \tfrac{\mu+\nu}{2} \right) \\
&\quad + \sum_{i=3}^{K-1} \pi_i \left(\tfrac{\nu}{i-1} + \tfrac{\mu}{i+1} - \tfrac{\mu+\nu}{i} \right) \\
&\quad - \tfrac{\pi_K \nu}{K} + \tfrac{\pi_K \nu}{K-1} \\
&= \sum_{i=1}^{K-1} \left(\tfrac{\pi_{i+1}\nu}{i(i+1)} - \tfrac{\pi_i \mu}{i(i+1)} \right).
\end{aligned}
$$

The sum of all first order terms is guaranteed to be positive if we have $\mu \leq \frac{\pi_{i+1}}{\pi_i}\nu$, for all $i = 1, ..., K-1$. Similarly, the sum of all the second order terms can be written as

$$
\begin{aligned}
G_2 &= -\pi_2 \tfrac{\mu\nu}{2} + \tfrac{\pi_K \nu^2}{K-2} - \tfrac{\pi_K \nu^2}{K-1} \\
&\quad + \sum_{i=3}^{K-1} \pi_i \left(\tfrac{\nu^2}{i-2} - \tfrac{\mu\nu}{i-1} - \tfrac{\nu^2}{i-1} + \tfrac{\mu\nu}{i} \right) \\
&= \sum_{i=2}^{K-1} \left(\tfrac{\pi_{i+1}\nu^2}{i(i-1)} - \tfrac{\pi_i \mu\nu}{i(i-1)} \right).
\end{aligned}
$$

We have $G_2 \geq 0$ if $\mu \leq \frac{\pi_{i+1}}{\pi_i}\nu, \forall i = 2, ..., K-1$. For any k-th order terms, there is

$$
G_k = \sum_{i=k}^{K-1} \left(\tfrac{\pi_i \nu^k}{(i-k-1)(i-k)} - \tfrac{\pi_i \mu\nu^{k-1}}{(i-k)(i-k+1)} \right).
$$

For the last term with order $K-1$,

$$
\begin{aligned}
G_{K-1} &= -\pi_{K-1}\mu\nu^{K-2} + \pi_{K-1}\tfrac{\mu\nu^{K-2}}{2} \\
&\quad + \pi_K \nu^{K-1} - \tfrac{\pi_K \nu^{K-1}}{2} \\
&= \tfrac{\pi_K \nu^{K-1}}{2} - \tfrac{\pi_{K-1}\mu\nu^{K-2}}{2}.
\end{aligned}
$$

When μ and ν satisfy $\mu \leq \frac{\pi_i}{\pi_{i+1}}\nu, \forall i = 1, ..., K-1$, we are guaranteed to have positive gain G from one pass of RTL inference.

A.2 Experimental Details

In this section, we provides more detailed description for the experimental setup used in Section 4.

A.2.1 Datasets

The statistics for datasets used in our experiment in shown in Table 5.

A.2.2 Model architecture and fine-tuning

The LLM chosen in this paper is the Flan-T5-base model (Chung et al., 2024), obtained from HuggingFace. We chose the base version, which has a parameter size of 200M, in order to maintain the same model size as previous works for a fair comparison. We use the Flan version of T5, which is additionally instruction-tuned, as we employ instruction prompts for our multi-task training.

We finetuned the Flan-T5 model in a multi-task setup as described in Section 2.1. The model is trained by maximizing the likelihood objective given in Eq 4. We train the model for 4 epochs

using our mixture-of-tasks approach with a linear learning rate schedule starting from 1e-5 with early stopping. We use the validation split of the dataset for deciding hyperparameters used for training.

A.2.3 Prompt Construction

For the MIND dataset, we use the same prompts used by Zhang and Wang (2023) to facilitate fair comparison. We utilize the headings of the news articles in MIND-small training dataset. To limit the input prompt length to 512 tokens, we include a maximum of 50 user click history items, each with a maximum length of 10 tokens. The entire history is capped at a maximum length of 450 tokens. Additionally, for each candidate title, we allow a maximum of 20 tokens. All these hyperparameters align with the code repository of Zhang and Wang (2023). We use the T5 separator token '<s>' to separate history as well as candidate items. The instruction prompts used along with the context and candidates are reported in Table 6. For the QuoteRec dataset, we reuse the prompts reported by UniTrec (Mao et al., 2023). We utilize the same instruction prompts as for MIND, as well as the same item separators.

A.3 Additional Experiments

In this section, we present additional experiments for analyzing the performance of the proposed approach.

A.3.1 From pair-wise to list-wise

To understand the performance of GLIMPSE beyond pair-wise, we delve further into extending the approach to list-wise input by incorporating more than two samples at a time during training. Without loss of generality, we consider triple-wise comparison task only in this experiment. More specifically, during training, we present the model with three candidate documents simultaneously and ask it to pick out the most relevant one. Employing a sampling strategy akin to the pairwise task outlined in Section 2.1, we ensure a balanced representation of training samples across point-wise, pair-wise, and triple-wise tasks. For evaluation, we compared the point-pair-triple wise multi-task trained model with GLIMPSE on point-wise ranking inference task. The result is presented in Table 7.

Our results indicate that the inclusion of this triple-wise comparison task led to a degradation in pointwise performance in Table 7. This shows that processing more than two candidate items at a time

poses a challenge for small-sized models, which justifies our proposed combination of pointwise and pairwise approaches. Furthermore, this finding aligns with claims from related literature (Qin et al., 2024; Sun et al., 2023) that show that list-wise approaches perform more poorly than pairwise ones.

A.3.2 Effect of point-wise initialisation

As we presented, GLIMPSE is a two-stage approach which the point-wise and pair-wise inferences are used for initialisation and refinement respectively. To understand the effect of point-wise initialisation, we conduct an experiment to compare random initialisation with the proposed point-wise initialization approach. More specifically, with the same multi-task trained model, we perform different inference strategies. The first strategy is based on random initialization, where we perform RTL passes on a randomly-initialized list instead of utilizing a point-wise prior on the MIND dataset. The result is presented in Table 8. We can observe that the results of random initialisation are much worse compared to the proposed point-wise initialisation followed by RTL passes. This observation holds even as we increase the number of RTL passes.

Beside empirical results, our main theory presented also provides the justification of the point-wise initialization. From a theoretical standpoint, the point-wise initialization yields a more advantageous initial state distribution in the underlying Markov chain. As demonstrated in the main paper, the initial distribution, or prior distribution resulting from point-wise inference, can be expressed as Eq. 11. The induced distribution can be approximated as an exponential distribution, with its parameter determined by the precision of the point-wise model. This implies that a robust point-wise inference model can offer an adequate starting distribution for our two-stage inference approach. In contrast, a randomly-initialized prior distribution would adhere to a uniform distribution, limiting the extent of enhancement that the pairwise inference can achieve via the transition matrix.

A.3.3 Quote Recommendation Task

In addition to the news recommendation datasets, we demonstrate that our approach is competitive on other text recommendation tasks. We use the QuoteRec dataset (Wang et al., 2020), which focuses on recommending a quotation appropriate to a conversational context. For fair comparison to previous work, we use the *Reddit-quotation* dataset

with the same splits as in Mao et al. (2023). Each conversation in this dataset has one positive label, with 1111 quotations in total. For this dataset we follow the work of Mao et al. (2023) to ensure we use the same splits, and thus compare to BERT4Rec (Sun et al., 2019), RoBERTa-Sim (Qi et al., 2022), UNBERT (Zhang et al., 2021), and UniTRec (Mao et al., 2023). Summary statistics can be found in Table 5 and results on this dataset are in Table 9. The performance of our approach on the QuoteRec dataset follows the same trend as above: pointwise scores are improved as we perform RTL passes. Here we see that our method outperforms all existing baselines except UniTRec (Mao et al., 2023). We hypothesize that this is because the QuoteRec dataset has only one positive and 1110 negative samples in each impression, which reduces the benefit of pairwise ranking.

A.3.4 Runtime Comparison

In order to show the efficiency of the proposed GLIMPSE algorithm, we also compare the inference times with baselines in Table 2. The result is shown in Table 10. We include the results to present the runtime of our approach along with other inference baselines. We report the running time results, measured in seconds, for our approach and baseline inference methods on 10% of the MIND test data in the following table. These results are based on three repeated runs of each experiment, with the average and standard deviation provided to ensure reliability. For some baselines, we were unable to obtain a complete runtime due to excessive duration. In these cases, we estimated the runtime based on early stopping at 10% completion.

A.3.5 Model Size Considerations for News Recommendation

While large language models (LLMs) have demonstrated impressive capabilities in recommendation tasks (Liu et al., 2023a; Qin et al., 2024) , news recommendation presents unique challenges that necessitate careful consideration of model size. The fast-paced nature of news, and an evolving information landscape, demands rapid response times and efficient computation. Consequently, existing body of research in news recommendation primarily focuses on smaller, more efficient models (Zhang and Wang, 2023; Mao et al., 2023). In line with this emphasis on practicality, our work prioritizes model size considerations to ensure real-world applicability. Our proposed integration of point-wise and

pairwise learning with $O(K)$ complexity further exemplifies how news ranking performance can be enhanced while maintaining efficiency, a crucial aspect in news recommendation.

Dataset	Property	Train	Val.	Test
MIND	Impressions	149116	7849	73152
MIND	Users	49123	6981	50000
Adressa	Impressions	290523	63226	252902
Adressa	Users	131740	46919	115458
QuoteRec	Conversations	35633	4454	4454
QuoteRec	Quotes	1111	830	795

Table 5: Summary statistics for datasets used in our experiments. In line with existing literature we use the MIND-Small subset of MIND for training, and the MIND test set for evaluation.

Task	Input prompt	Target word
Rel(.)	Given that user has clicked on <user_history>. Is candidate A: <candidate_title> a good recommendation to user? Respond Yes or No.	Yes/No
Pref(.)	Given that user has clicked on <user_history>, amongst the 2 news titles, candidate A: <candidate_title1> and candidate B: <candidate_title2> which is more appropriate for recommendation to user? Respond Candidate A or Candidate B.	Candidate A/ Candidate B

Table 6: Sample prompts used for each task.

Model	AUC	MRR	nDCG@5	nDCG@10
Multi-task Point-Pair-Triple	68.17	33.13	36.65	42.84
Multi-task Point-Pair	68.18	33.56	37.27	43.48

Table 7: Comparison of triple-wise and pair-wise on MIND test set.

Initialisation Method	AUC	MRR	nDCG@5	nDCG@10
Random	52.14	25.48	25.31	31.65
Point-wise inference	68.97	33.73	37.41	43.62

Table 8: Comparison of different inference initialisation methods on MIND test set.

Model	MRR	nDCG@5	nDCG@10
Bert4Rec	33.59	34.26	37.37
RoBERTa-Sim	37.13	37.96	41.18
UNBERT	39.75	40.74	43.69
UniTRec	**41.24**	**42.38**	**45.31**
GLIMPSE$_{point}$	40.18	41.39	43.70
w/ pair (top 2)	40.31	41.46	43.77
w/ pair (top 5)	40.43	41.53	43.85

Table 9: Results on the QuoteRec dataset compared to baseline methods. Results for baseline methods are extracted from the respective papers. Inference with GLIMPSE is performed using a single RTL pass.

Inference Method	Pairwise	Pointwise	Boxfilling	BubbleSort	N-window	S-window	1 RTL top-5	1 RTL top-10
Runtime (s)	> 20000	651.9±0.8	>20000	>20000	>8000	>9000	1234.1±7.5	1872.1±15.1

Table 10: Inference runtime comparison (in seconds) of different approaches on 10% of MIND test data.