IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 29 April 2025, accepted 24 May 2025, date of publication 27 May 2025, date of current version 10 June 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3574073

RESEARCH ARTICLE

Leveraging RAG With Transformer for Context-Based Personalized Recommendations

FATEN S. ALAMRI[1], AMJAD REHMAN[2], (Senior Member, IEEE), BAYAN ALGHOFAILY[2], ADEEL AHMED[3], AND KHALID SALEEM[3]

[1]Department of Mathematical Sciences, College of Science, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11564, Saudi Arabia
[2]Artificial Intelligence and Data Analytics Lab, CCIS Prince Sultan University, Riyadh 11586, Saudi Arabia
[3]Department of Computer Science, Quaid-i-Azam University, Islamabad 45320, Pakistan

Corresponding author: Amjad Rehman (arkhan@psu.edu.sa)

This research was funded by Princess Nourah bint Abdulrahman University and Researchers Supporting Project number (PNURSP2025R346), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

ABSTRACT Recent advancements in large language models (LLMs) have shown significant progress in addressing challenges related to data sparsity and the cold-start problem. In e-commerce, recommendation systems are widely used as strategic tools to boost sales and enhance the customer experience by helping users find relevant products. Custom LLMs, leveraging textual features from user feedback, have been successfully applied to recommendation systems, yielding improvements across various recommendation scenarios. However, most existing methods rely on training-free recommendation approaches, which depend heavily on pre-trained knowledge. When LLMs are trained on sparse data or lack historical information, their performance in recommendation systems can be negatively impacted. Furthermore, inference with LLMs tends to be slow due to autoregressive generation, which limits the efficiency of traditional recommendation methods. To address these challenges, our contributions are: We proposed the Retrieval Augmented Generation with Transformer Recommendation (RAGX11Rec) framework. This framework integrated LLMs with a transformer-based model in a two-step process: 1) RankRAG is used to filter the top-k preferences via tuning the LLM for effective context ranking, 2) a transformer model with 11 embedded layers generated the top-N recommendations based on ranked preferences. Our instruction-tuned transformer module demonstrates superior performance by incorporating a fraction of ranked data into the training process. We evaluated the effectiveness of RAGX11Rec against state-of-the-art baseline methods using two public datasets taken from AliExpress and Epinions. Experimental results indicate that RAGX11Rec consistently outperforms other methods in recommendation accuracy and efficiency. Our key findings are; 1) RAGX11Rec effectively addresses the cold-start problem by leveraging retrieval-augmented generation (RAG) and transformer-based ranking. Unlike traditional models, it can deliver high-quality recommendations even when user interaction data is limited. 2) Tested on public datasets, RAGX11Rec delivered consistent improvements over other models and proved its scalability and adaptability across diverse product categories. This suggests the framework is robust and adaptable enough for large-scale commercial use.

INDEX TERMS Recommender systems, transformer model, large language model, e-commerce, ranking, context, retrieval, augmented generation.

I. INTRODUCTION

Recommender systems were invented in the 1990s to help users select items that suit their tastes. There are many e-commerce sites such as Alibaba, Epinions, eBay, Amazon, and many others where traditional recommender systems provide recommendations. Indeed, many techniques are used for generating personalized recommendations, such as collaborative filtering [3], content-based, trust-aware [1], users' context information in conjunction with social networks to

The associate editor coordinating the review of this manuscript and approving it for publication was Syed Islam.

recommend news articles [5], [7], community- based [6], cross-domain [4] and time aware recommender systems [2]. Recent breakthroughs in large language models (LLMs) have made substantial strides in linguistic understanding and generation tasks. Using human evaluation and benchmark datasets, the Llama 2 model performs better than available open-source language models [16]. Building on these developments, LLMs are also applied to recommendation tasks such as retrieval and ranking, enhancing performance across multiple scenarios [8], [9], [10], [11], [12], [13], [14], [15]. Ranking enhances information retrieval results, and the RAG pipeline has further improved generated content quality [17], [18]. However, these methods still rely on a moderately sized model, such as BERT and T5, for ranking, which may not fully capture the relationship between the query and context and can be universal in zero-shot learning. While recent studies [19], [20], [21] have shown that large language models performed well on ranking tasks, how to effectively leverage this capability in the RAG pipeline remains an open question. Nowadays, attention mechanisms have been used in recommender systems [22], [23], [24]. The transformer model achieved unprecedented success in machine translation [3] using solely self-attention modules rather than recurrent or convolutional layers. By capturing intricate relationships between words regardless of distance in the sentence, the transformer mechanisms are better than earlier approaches reliant on RNNs and CNNs.

A. CHALLENGES AND MOTIVATION

The cold start problem greatly influences recommender systems' performance [25]. This occurs when a user starts with no ratings against the items. Data sparsity is another main problem; for instance, it is hard to find useful neighbors of a target user since the matrix is sparse; thus, the performance of recommender systems is affected negatively. According to Lü et al. and Bobadilla et al. just 1% of items may be purchased by an active user on Amazon [26], [27]. A popular method often used today is collaborative filtering, where we use the user-item matrix to calculate the similarity of any given user with all existing users. However, sparse input matrices may reduce the quality of recommendations [28]. Xu et al. [29] find optimal k-contexts for long document tasks. In general, a small k often fails to capture all the information of interest, resulting in degraded recall because the search pattern is too simplistic compared with what is done for retrieval. In contrast, a larger k value improves recall but adds unnecessary complexity to chatbots, diminishing their ability to provide accurate responses [30], [31]. RAGRank introduces a ranking mechanism that prioritizes the most contextually relevant preferences, helping ensure that the generated response is more precise and contextually appropriate. Without ranking, the RAG pipeline might struggle with sparse or loosely relevant data. RAGRank refines document selection, helping the model avoid irrelevant content that could otherwise introduce noise [31]. Inspired by RankRAG and transformer, we are motivated to develop a new hybrid recommendation model based on retrieval augmented generation through a self-attention mechanism to solve a user's cold start problem and generate top-N quality recommendations.

B. SIGNIFICANCE OF RESEARCH CONTRIBUTIONS

Enhanced Cold-Start Solutions: Traditional recommendation systems struggle with cold-start issues due to limited user history [25]. RAGX11Rec offers an advanced framework specifically designed to address this by integrating retrieval-augmented generation (RAG) with transformers. This approach helps generate accurate recommendations for new users without requiring extensive historical data, making it a valuable tool for e-commerce and similar platforms. Improved Context Relevance: By introducing RankRAG, the proposed model provides a sophisticated context-ranking mechanism that refines data selection for the recommendation process. This ranking model tunes the LLM to prioritize top-k preferences effectively, reducing irrelevant noise and ensuring the most contextually relevant data informs recommendations. This improvement in context relevancy directly impacts recommendation quality. Efficiency in Data-Sparse Environments: Many recommendation systems are data-intensive, yet RAGX11Rec can achieve high accuracy even in sparse data environments. Its transformer-based model, tuned for efficiency, enables faster processing, making it feasible for large-scale applications where computational efficiency is critical.

C. CONTRIBUTIONS

In this paper, we have the following contributions.

- We propose a framework called RAGX11Rec, a novel framework that enhances the retrieval-augmented generation (RAG) capabilities of large language models based on inference. The LLM re-ranks the retrieved contexts and then generates preferences based on the refined top-k results.
- We design a transformer model with 11 embedded layers that generate top-N recommendations based on the top-k preferences.
- We perform extensive experiments on the *AliExpress* and *Epinions* datasets and compare the proposed model with several strong baselines. We observed that the proposed model outperformed by a notable margin.

The rest of the paper is structured as follows. Section II describes the related work, section III discusses the proposed approach, section IV explains the experimental setup and results, and section V discusses the conclusions.

II. RELATED WORK

We limited the related work to deep learning-based recommendations, large language model-based recommendations and attention mechanisms.

A. DEEP LEARNING BASED RECOMMENDATIONS

A deep neural network plays a significant role in the recommender system for generating recommendations [32], [33]. To alleviate the problem of data sparsity and the computational cost of the deep learning model, it learns the feature representations of items and users in the recommendation process [34]. Pan et al. devised a correlative denoising autoencoder that used three autoencoders to produce recommendations, thus achieving higher coverage because each autoencoder only needs to handle a separate amount of the data [35]. Huang et al. presented a semantic model based on a deep neural network which calculates semantic similarity for top N recommendations [36]. Yang et al. built a model that tackled the dot product problem in matrix factorization that violates the triangle inequality of distance function by replacing it with Euclidean distance [37]. Taleb et al. [38] devised a model for makeup recommendation and used two MLPs to represent expert rules and labelled examples. As for the specific framework in which this is planned, the experts' knowledge guides the learning of the recommender model made by MLPs. However, there is no evidence of experiments to show whether the model had learned efficiently [38]. Guo et al. [39] proposed the DeepFM model. This model shared the deep and wide part of embedding vectors, reducing the time complexity to some extent. However, such a model has no pooling layer needed in learning high-dimensional features [39].

B. LARGE LANGUAGE MODEL BASED RECOMMENDATIONS

Large language models (LLMs) have recently made significant progress, with one example being the launch of ChatGPT based on LLM technology. LLMs consist of two main factors: first, making language model types larger; second, increasing the size of pre-training corpora [40], [41], [42], [43], [44], [45]. In a global sense, pre-trained LLMs adopt the mechanism of self-attention to process input texts and are optimized through next-token prediction [42], [46]. LLMs are used to generate recommendations about the text features of items. This means better recommendations [12], [47], do not just come from simply looking at ratings but also include statistical feedback on what users like and have bought recently or earlier via word embedding [10], [13], [14], [48], [49]. For instance, Chat-REC [50] enables ChatGPT to understand user preferences, contribute ideas and assist in explainable recommendations. Another direction in LLM-based recommendation is focused on tuning strategies for subtasks (such as rating prediction) to improve further performance [51]. TallRec [8] makes instructional-tuning decisions about whether to recommend an item. However, most existing work is predicated on specific recommendation tasks. All adopt autoregressive generation for inference purposes, which further increases waiting times. Karpukhin et al. developed a dense-embedding-based standalone retriever that retrieves external related information from the LLM in its original form [52]. This is then available to LLM for generations. Current studies focused on adjusting retrievers to meet the requirements of LLMs for task generation [53], [54], developing multi-step retrieval processes [55], [56], [57], or filtering away irrelevant contexts [58]. Some studies have been conducted to improve the operation capability of LLMs and retrieval augmented generation [59]. For design guidance, tuning methods have been developed to enhance search results on trained corpus [60]. Radawan et al. illustrate how large language models enable the improvement of mental health knowledge discoveries using social media content. The research utilized social media data to develop predictions about mental health patterns and human behaviors [84]. Yating et al. presented TQFLL, a novel unified analytics framework to improve translation quality in human and machine translations of allusions in multilingual corpora. The framework integrates large language models with traditional translation processes to enhance the accuracy and reliability of translation outputs [85]. Rony et al. introduced Medigpt, which explores how LLMs can assist in diagnosing and analyzing medical information, offering significant potential for improving healthcare data processing [86]. Ammar et al. researched how large language models (LLMs) function in predicting Arabic legal judgments and how LLMs can acquire Arabic legal understanding abilities to support automated judicial decisions [87].

C. ATTENTION MECHANISMS

Attention mechanisms have displayed their strength in various tasks, for example, machine translation [62] and image captioning [61]. The basic idea behind such a mechanism is that the output in any sequential form should relate only to parts of an input with some relevance, and the model concentrates upon those progressively. A significant advantage of attention-based approaches is their greater understandability. Recently, attention mechanisms have been used in recommender systems [22], [23], [24]. Attentional factorization machines can capture the interaction between features and how significant the interaction is and are very useful for content-aware recommendations. Nowadays, a purely attention-based sequenced approach, transformer [46], has achieved both state-of-the-art performance and efficiency on machine translation tasks previously dominated by RNN/CNN-based approaches [63], [64]. The transformer model uses the proposed self-attention modules to grasp sentence construction rules and retrieve relevant words to generate the next word. Gheewala et al. developed a deep transformer model for a textual review-based recommendation system that captured user sentiments and preferences in textual data [77]. Zhao et al. proposed a recommender system based on pre-trained large language models. This model required extensive computational overhead during training and fine-tuning, limiting its usability in resource-constrained environments [78]. Recency-based sampling for sequential recommendations made handling larger datasets

more scalable and efficient, tackling the computational challenges. One limitation of the recency-oriented model is that it prevented it from modelling long-term dependencies in user behaviour [79]. In 2024, Reddy et al. [80] proposed a federated transformer model for movie recommendations. This framework provided models trained locally on user devices to tackle user privacy and aggregated to the center-only insights. On the other hand, the federated learning configuration has considerable communication latency, making the model unsuitable for real-time recommendation systems, though it was efficient for privacy preservation [80]. Moreira et al. designed a Transformers4Rec session-based recommendation model that learns about user preferences in a session. Nevertheless, the dependency on session-level data constrained the generalized capability of the model to represent explicit user interaction history, which, in turn, reduced the model's performance in the complete recommendation task [81]. Wang et al. introduced a convformer model that showed better performance in capturing long-term dependencies. One limitation of this model is that the design depended on sequential data, leading to sub-optimal performance when the user interaction is sparse [82].

III. PROPOSED METHODOLOGY

Figure 1 shows the proposed framework for generating top-N recommendations. It consists of two main modules: (i) Select the top-k preferences based on the RankRAG model to filter context-based feedback. (ii) Generate top-N recommendations using transformer models.

A. PROBLEM DEFINITION

To solve the problem of a user cold start in e-commerce, we can formulate the problem as let us assume that $U = \{u_1, u_2, \ldots, u_n\}$ be the set of all users, $I = \{I_1, I_2, \ldots, I_m\}$ be the set of all items provided by users, $P = \{p_1, p_2, \ldots, p_k\}$ be the set of preferences associated with items, and $R = \{r_1, r_2, \ldots, r_k\}$ be the set of ratings given to items. Given a new user U_{new}, who interacts with the system to predict the top-N recommendations by leveraging contextual data from external knowledge sources such as user preferences and relevant contexts from trusted neighbours. No historical data is available for U_{new}, making it a cold start problem.

B. SELECT TOP-K PREFERENCES BASED ON RANKRAG MODEL

Some recent research [54], [65], [66] developed new life into the RAG performance of LLMs by instructing them to engage in context-rich generation learning tasks. In our proposed model, the RankRAG instructions tune the LLM specially for retrieval-augmented generation and context ranking to obtain better top-k contexts in users' preferences.

1) INSTRUCTION TUNING

RankRAG process consists of 'instruction tuning', which focuses on specific instructions that make it easier for a language model to rank relevance when retrieving contexts. The model effectively distinguishes between valid and less useful contexts during this phase. By providing the model with focused training, instruction tuning brings the retrieval and generation processes closer. This significantly improves the model's usefulness for context ranking in users' preferences [45]. We performed the fine-tuning to better rank contexts based on task-specific instructions. Instruction tuning can be formalized as

$$\min_{\theta} \mathcal{L}(M_\theta, D_{\text{instr}})$$

where M_θ is the language model with parameters θ. D_{instr} is the dataset with instruction-specific tasks. $\mathcal{L}$ is the loss function to optimize relevance prediction.

The objective of this phase is to apply the LLM to distinguish between contextually relevant and irrelevant user preferences. To achieve this, we employed a supervised fine-tuning approach where the model is trained on a AliExpress dataset denoted as D_{instr}. This dataset comprises examples formatted as input-output pairs with task-specific instructions, which simulate real-world retrieval ranking tasks. Each instruction prompts the model to rank user preferences based on relevance to a given query. The D_{instr} dataset is constructed by extracting real user-item interaction data from the AliExpress dataset, supplemented by synthetic instructions. Each training example includes:

- A query or context prompt, for example, ''Recommend me electronics under $50''.
- A set of candidate item preferences, where only a subset is genuinely relevant.
- Labels indicating which preferences are relevant vs. irrelevant.

2) CONTEXT RICH FINE TUNING

At this stage, the model performed various training rounds using rich contexts in users' feedback in the Aliexpress dataset. This ensures that the proposed model can generate contextually appropriate and accurate preferences [67]. We fine-tuned the instruction-tuned model using augmented retrieval data to improve its ranking and retrieval capabilities.

$$\min_{\theta} \mathcal{L}(M_\theta, D_{\text{aug}})$$

where D_{aug} is the retrieval-augmented dataset containing contexts enriched with task-specific data. $\mathcal{L}$ is fine-tuning loss to enhance contextual relevance.

We applied context-rich fine-tuning to further enhance the model's ability to make relevance judgments using actual retrieval data from the AliExpress dataset. Context-rich fine-tuning is a critical phase in the RankRAG model that refines the instruction-tuned language model using real-world, contextually enriched data to enhance its ability to rank user preferences accurately. This process leverages a retrieval-augmented dataset D_{aug}. composed of detailed user feedback such as reviews, ratings, product metadata, and behavioral signals from platforms like AliExpress. Unlike instruction tuning, which teaches the model general ranking

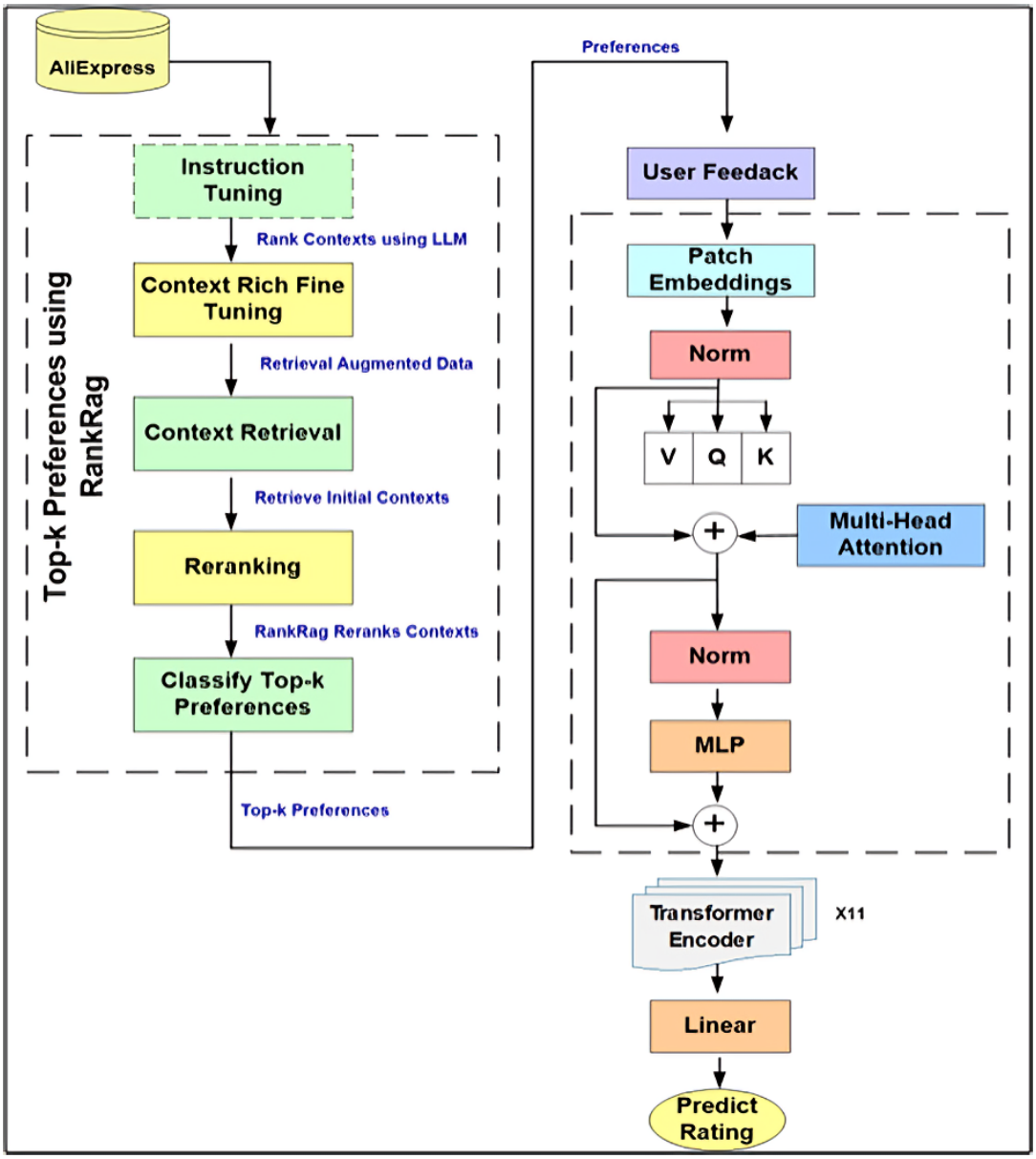


FIGURE 1. Proposed framework for generating top-N recommendations.

behavior, context-rich fine-tuning immerses the model in domain-specific, nuanced scenarios, allowing it to distinguish relevant from irrelevant information and personalize recommendations more effectively. By optimizing a loss function over this dataset, the model learns to prioritize semantically meaningful user preferences, even in sparse or cold-start situations.

3) CONTEXT RETRIEVAL

The retrieval mechanism searches the dataset and selects a set of potentially relevant preferences. This module separately guarantees the retrieval of relevant preferences using various historical data [52]. It can be modeled as

$$C_{\text{retrieved}} = R(q, M_{\text{fine}}) \tag{1}$$

where R is the retrieval function using the fine-tuned model M_{fine}. q: Input query and $C_{\text{retrieved}}$ is the retrieved set of relevant contexts.

4) RE-RANKING

Re-ranking re-ranked the retrieved contexts. RankRAG applies its re-ranking capabilities, refining the initial list of retrieved contexts to bring the most relevant ones to the top. The LLM-based model uses its learned instructions and fine-tuned context to further enhance the quality of the ranked results [68]. We re-ranked the retrieved contexts $C_{\text{retrieved}}$ to prioritize the most relevant ones based on the score.

$$C_{\text{ranked}} = \text{Sort}(C_{\text{retrieved}}, S(c \mid q, M_{\text{fine}})) \tag{2}$$

where $S(c \mid q, M_{\text{fine}})$ is the scoring function for relevance of each context c given query q and C_{ranked} is the retrieved contexts sorted by their relevance scores.

D_{instr} is the dataset that is primarily synthetic and is used during the instruction-tuning phase. It contains explicit task-oriented instructions paired with ranked preferences or relevance labels. These instructions simulate various user intents and teach the model how to rank items based on user queries. Each entry in D_{instr} is designed to train the model to follow structured commands and generalize ranking behavior across scenarios.

D_{aug} is the augmented retrieval dataset that is derived from real-world user interactions on platforms such as AliExpress and is used during the context-rich fine-tuning phase. It includes actual user-generated content, such as product reviews, textual feedback, ratings, and historical preferences. These entries are retrieved using a pre-trained retriever model and further enriched with auxiliary context (e.g., similar users' preferences, time of purchase, item popularity).

Once context candidates are retrieved, the RankRAG model performs re-ranking using the following steps:

1) **Input:** The retrieved contexts $C_{\text{retrieved}} = \{c_1, c_2, \ldots, c_n\}$ and the original query q.
2) **Scoring:** Each context c_i is passed through the fine-tuned model M_{fine}, which computes a relevance score:
$$S(c_i \mid q, M_{\text{fine}}) = \text{softmax}(f(q, c_i))$$
where $f(q, c_i)$ is a contextual relevance function modeled via transformer-based attention between the query and the candidate context.
3) **Sort and Rank:** The contexts are sorted in descending order of their relevance scores:
$$C_{\text{ranked}} = \text{Sort}(C_{\text{retrieved}}, S(c \mid q, M_{\text{fine}}))$$
4) **Top-K Selection:** From the ranked list C_{ranked}, the top-k preferences $T_k = \{c_1, \ldots, c_k\}$ are selected and passed to the transformer model for generating the final recommendations.

5) CLASSIFY TOP-K PREFERENCES

Once the contexts have been re-ranked, the final step is to classify and select the top-k preferences. The top-k preferences represent the model's best prediction based on the ranked context.

$$T_k = \{c_1, c_2, \ldots, c_k\} \text{ where } c_i \in C_{\text{ranked}}, i \leq k$$

where T_k is Top-k ranked preferences and C_{ranked} is the set of ranked contexts.

6) EVALUATION OF RANKRAG MODEL

Figure 2 compares the effectiveness of three models on AliExpress dataset: RankRAG, PaLM [41], and LLaMA 2 [16]. The results are analyzed based on Precision, Recall, and F1-Measure. RankRAG consistently outperforms the other two models, achieving the highest scores across all metrics. PaLM follows closely behind RankRAG in performance, maintaining slightly lower scores but surpassing LLaMA 2. LLaMA 2 shows the weakest results across all categories. RankRAG's leading performance can be attributed to its hybrid approach of combining retrieval-augmented generation with a ranking mechanism. This allows RankRAG to identify and prioritize the most pertinent contextual details effectively. By selecting the top-ranked feedback, RankRAG enhances its ability to classify user preferences accurately.

C. GENERATE TOP-N RECOMMENDATIONS USING TRANSFORMER MODEL

The proposed model generates the recommendations based on the transformer model. Transformer-based architecture [46], specifically leveraging multi-head attention and several layers of encoders to process user feedback and predict ratings. It consists of the following modules:

1) USER FEEDBACK

The user feedback consists of top-k preferences after filtering from RankRAG model. Let the input data X represent the user feedback, which consists of user preferences. Then X can be described as:

$$X = [x_1, x_2, \ldots, x_n]$$

where x_i represents the features of the feedback for each item i.

2) PATCH EMBEDDINGS

The patch embeddings convert the user feedback into a format the model can process [46]. The user feedback data X is transformed into patch embeddings to create a dense vector representation. The embedding vector is denoted by W_e such that

$$E = W_e X \tag{3}$$

where E is the embedded representation of the user feedback and W_e is the embedding vector learned during training.

The normalization layer ensures that the inputs are standardized to have mean zero and variance one. If E is the input to the normalization layer, the normalized output $\hat{E}$ is computed as

$$\hat{E} = \frac{E - \mu}{\sigma} \tag{4}$$

where μ and σ are the mean and standard deviation of the embedding vector E.

3) MULTI-HEAD ATTENTIONS

The multi-head attention mechanism [46], [69] computes the output by creating three vectors from the input: Query Q,

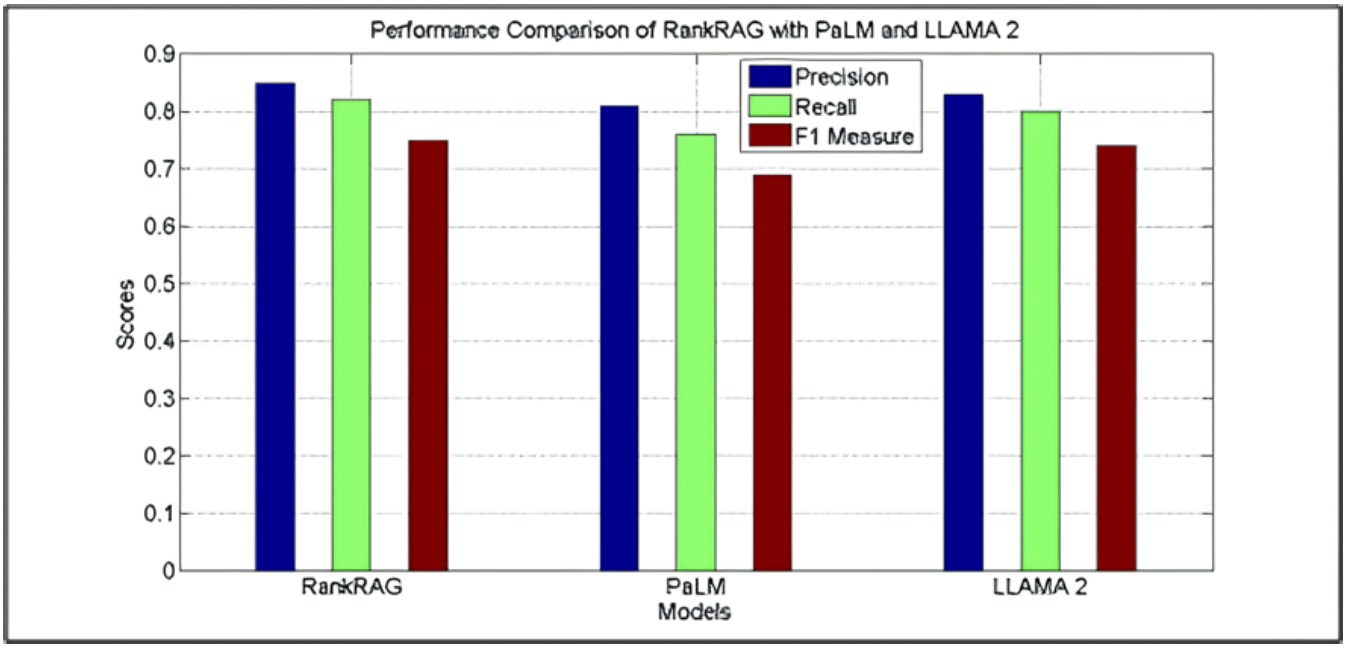


FIGURE 2. Evaluation of RankRAG model based on precision, Recall and F1 measure with baselines.

Key K, and Value V. These are linear projections of the input embeddings E, computed as:

$$Q = W_Q E, \quad K = W_K E, \quad V = W_V E$$

where W_Q, W_K, and W_V are learnable weight matrices for Query, Key, and Value, respectively.

The attention scores are computed by taking the dot product of the Query and Key matrices and scaling by the square root of the dimension d_k:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{5}$$

The Multi-Head Attention mechanism is formed by concatenating multiple heads such as:

$$\text{Multihead Attention}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W_O \tag{6}$$

where each head is computed in (7).

$$\text{head}_i = \text{Attention}(Q_i, K_i, V_i) \tag{7}$$

and W_O is the output projection vector.

A residual connection adds the original input back to the output of the attention mechanism. Given input E and attention output A, the result of the residual connection is:

$$A' = A + E \tag{8}$$

This helps the model retain information from earlier layers.

Then, the normalization step is applied to further stabilize the attention mechanism's output. The output from the residual connection A' is passed through a feed-forward neural network called multilayer perceptron (MLP). This MLP typically consists of two linear transformations with a ReLU activation in between:

$$MLP(A') = \text{ReLU}(W_1 A' + b_1)W_2 + b_2 \tag{9}$$

where W_1, W_2, b_1, and b_2 are learnable parameters.

4) TRANSFORMER ENCODER

The model uses multiple transformer encoder layers. Each encoder layer includes multi-head attention, normalization, and a feed-forward network as described above. These layers sequentially refine the representations. Suppose the output of layer i be O_i. For each layer, we compute:

$$O_i = \text{Transformer Layer}(O_{i-1})$$

5) LINEAR LAYER

After passing through 11 encoder layers, the output is reduced in dimensionality using a linear layer computed using (10).

$$P = W_L O_{11} + b_L \tag{10}$$

where W_L is the weight vector, b_L is the bias, and P is the linear projection of the final transformer layer's output.

6) PREDICT RATING

The output P is the predicted rating for the user and is represented as:

$$\hat{y}_i = \frac{e^{P_i}}{\sum_j e^{P_j}} \tag{11}$$

where $\hat{y}_i$ represents the probability of predicting a particular rating class.

Algorithm 1: Generating Top-N Recommendations

Input: User Feedback (Top-k Preferences)
Output: Predicted Rating $\hat{y}_i$

1) $X = [x_1, x_2, \ldots, x_n]$ represents the user feedback with features x_i.
2) Transform user feedback X into patch embeddings using (3).
3) Normalize the embedding vector E using (4) and return $\hat{E}$.
4) Perform multi-head attention (Q, K, V) for Query, Key, and Value from input embeddings $Q = W_Q E, K = W_K E, V = W_V E$.
5) Compute attention scores using (5), (6), and (7).
6) Add residual connection in attention output as formalized in (6) using (8) and return A'.
7) Apply feed-forward network and compute $MLP(A')$ using (9).
8) **for each** layer $i = 1$ to 11 **do**
   a) Apply multi-head attention, residual connection, and MLP such that O_i=Transformer Layer(O_{i-1}).
9) Compute linear projection using (10) and return P.
10) Estimate the rating using (11).
11) **return** $\hat{y}_i$.

D. TIME COMPLEXITY

The time complexity of Algorithm 1 can be analyzed based on its operations. Step 4, which involves multi-head attention, has a time complexity of $O(n^2 d)$, where n is the number of input features (feedback) and d is the embedding dimension, due to pairwise attention computation. Step 8 involves looping through L transformer layers (e.g., $L = 11$), and for each layer, operations such as multi-head attention and feed-forward neural networks dominate. The feed-forward layers have a complexity of $O(nd^2)$. Therefore, for L layers, the complexity becomes:

$$O(L \cdot n^2 d + L \cdot nd^2)$$

The final linear projection step (step 11) and normalization steps have a lower complexity of $O(nd)$. Hence, the overall time complexity of the algorithm is:

$$O(L \cdot n^2 d + L \cdot nd^2)$$

where the multi-head attention operations dominate for large n and d.

IV. EXPERIMENTAL SETUP AND RESULTS

For experiments, we have utilized the dataset of the well-known e-commerce website of aliexpress.com, published by Ahmed et al. in 2021 [4]. The dataset contains more than 18 categories (domains) and 205 subcategories. The dataset includes the users that submitted the reviews against items in 18 different categories, such as electronic, entertainment, education, house, and garden, etc, and the items are rated on at 1 to 5 scale. The statistics about the AliExpress dataset are described in Table 1.

TABLE 1. Statistics of AliExpress dataset.

Total number of users	1,506,850
Total number of items	49,221
Total number of ratings	2,260,923
Total Sub Categories	205
Total Categories	18

We have also utilized another public dataset, a social network dataset called Epinions [83]. Table 2 reported the statistics about Epinions dataset.

TABLE 2. Statistics of Epinions dataset.

Total number of users	22,166
Total number of items	296,277
Total number of ratings	922,267
Total Categories	27

We performed experiments on various data views with 70% training and 30% testing to assess the performance of the proposed model. The views of data included: ***All Users:*** Both datasets contains all users who have provided ratings for items on a scale of 1 to 5. This view ensures that every user who interacted with the system by rating items is accounted for in the evaluation process.

Cold Start Users: Cold Start users refer to those users who have provided minimal interactions. For our experiments, we defined cold-start users as those rated fewer than three items in each domain. This view aims to evaluate the models' capability in addressing the cold-start problem, where limited data is available for these users.

A. EVALUATION METRICS AND BASELINES

The evaluation metrics include mean reciprocal rank (MRR@k), normalized discounted cumulative gain (NDCG@k) [70], [75], and recall (Recall@k) [70], where 'k' is in the range of 5 to 10. We select and save the model with the highest validation scores for evaluation, specifically Recall@10 for retrieval and NDCG@10 for ranking, with predictions ranked against all items in the AliExpress dataset. We implemented several state-of-the-art sequential recommenders, such as RNN-based models (NARM), transformer-based recommenders (SASRec, BERT4Rec), and a linear recurrence replay model (LRURec).

SASRec: A transformer model with unidirectional attention, processing input sequences to predict subsequent items [72].
NARM: NARM is an RNN-based model that uses local and global encoders to capture sequential transition patterns [71].
BERT4Rec: A bidirectional attention model trained

by predicting masked items and enhancing item recommendation [73].

LRURec: An efficient sequential recommender based on LRU, also employed as the retriever model in LlamaRec [74].

GPTRec: Aleksandr et al. introduced GPTRec, a novel approach that leverages the powerful GPT-2 model while addressing large vocabulary issues [76].

B. PARAMETER SETTINGS

Table 3 provides an overview of standard parameters for configuring a proposed RAGX11Rec model and suitable values for each parameter. It includes essential settings like model_name, which defines the type of pre-trained model, and model_checkpoint, which specifies the path to a model's checkpoint. Model size is categorized as *small*, *base*, *large*, or *XL*, while input and output token limits typically range from 512 to 1024 for input and 50 to 100 for output. Parameters such as temperature, top-k, and control generation are adjusted based on training needs for fine-tuning, learning rate, epochs, and weight decay.

Models including SASRec, LRURec, GPTRec, NARM, and BERT4Rec are used. Embedding dimensions vary across models, with typical sizes of 50, 100 for SASRec, LRURec, and NARM, while GPTRec and BERT4Rec use 768 or 1024 dimensions. Most models utilize a learning rate between 1e-3 and 1e-5, with Adam as the optimizer. The dropout rate is generally between 0.1 and 0.5, and batch sizes range from 8 for GPTRec to 256 for SASRec and NARM. GPTRec and BERT4Rec use pre-training (GPT corpus and BERT, respectively), while others do not. SASRec, NARM, and BERT4Rec apply self-attention mechanisms, with SASRec and BERT4Rec using 12 attention heads. Cross-entropy is the most common loss function, and most models are trained for 100 to 200 epochs.

TABLE 3. Parameter settings of RAGX11Rec.

Parameter	Values
model_name	Transformer based
input_max_length	512-1024
output_max_length	50-100
batch_size	250
temperature	0.7
top_k	20-50
max_tokens	20-50
learning_rate	0.03
epochs	100
weight_decay	0.02

C. EXPERIMENTAL RESULTS

We have performed various experiments for parameter analysis on the AliExpress and Epinions datasets.

1) IMPACT OF LEARNING RATE ON RAGX11REC

We studied the impact of the learning rate on the proposed model with a learning rate of 0.01 to 0.05 over

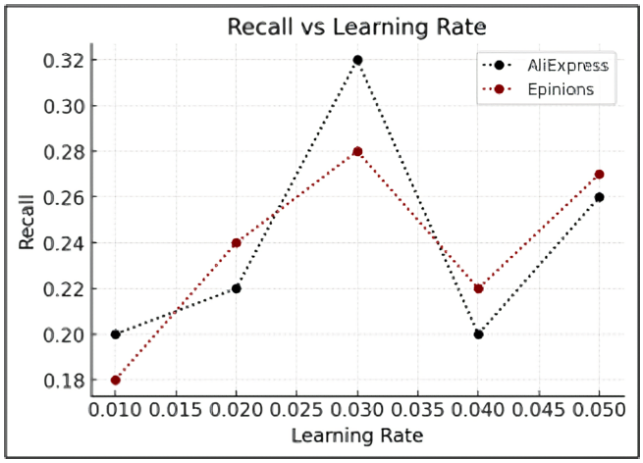


FIGURE 3. Impact of learning rate on RAGX11Rec for AliExpress and Epinions datasets.

AliExpress and Epinions datasets. Figure 3 shows how different learning rates affect the recall of a proposed model. In AliExpress, As the learning rate increases from 0.01 to 0.03, the recall improves steadily, with a peak recall of around 0.32 at a learning rate of 0.03. This suggests that 0.03 is the optimal learning rate for the proposed model, where it most effectively identifies the positive cases. Overall, a learning rate of 0.03 is the most effective for maximizing recall in our experimental study. Similarly, in Epinons dataset, the RAGX11Rec model is converged at 0.03.

2) IMPACT OF DIMENSIONALITY ON RAGX11REC

Figure 4 illustrate the illustrate the performance of three recommendation models, RAGX11Rec, SASRec, and BERT4Rec on the AliExpress and Epinions, for the metrics NDCG@5 and NDCG@10. We can clearly see that RAGX11Rec overcome two other models on all metrics. That is, with the higher-dimensional embeddings, RAGX11Rec seems to be better able to capture user preferences and effectively model the interaction between items and users. The steady growth in NDCG values shows that it scales well and can extract more complicated latent spaces. In contrast, SASRec and BERT4Rec show relatively slower improvements with increasing dimensionality, particularly in the lower-dimensional range (10 to 100). This means that while thesemodels are very strong, they may be limited in their modeling of user behavior.

3) IMPACT OF WEIGHT DECAY ON RAGX11REC

The effect of weight decay on the model's precision is shown in Fig. 5, demonstrating a clear tradeoff between regularization and performance. Initially, at a weight decay of 0.00, the precision is around 0.80. That means the proposed model performed well without any regularisation but may tend toward overfit. When weight decay rises to 0.01 or 0.02, precision leaps to 0.85, where the model generally performed best. At this stage, moderate weight

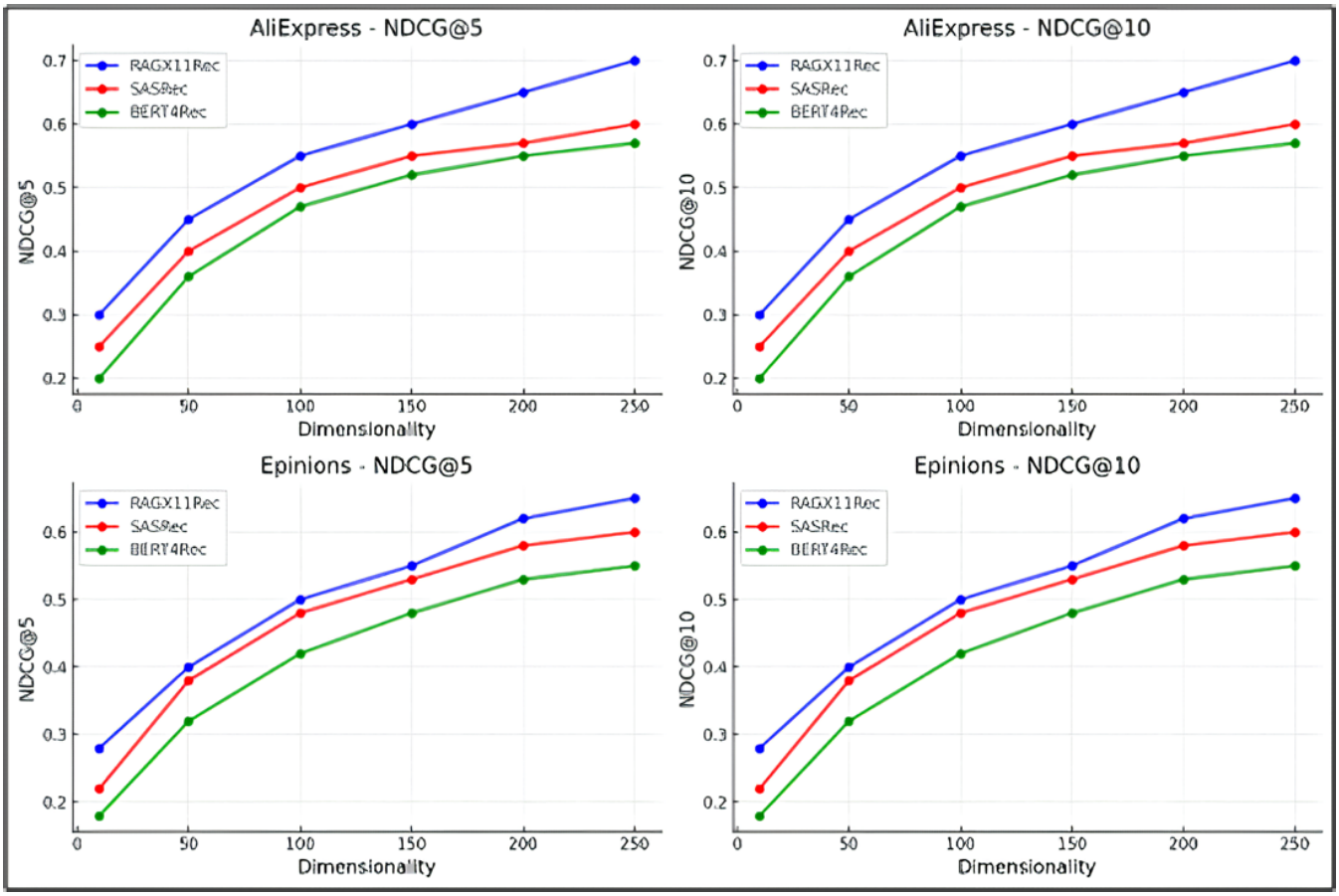


FIGURE 4. Impact of dimensionality on RAGX11Rec for AliExpress and Epinions datasets.

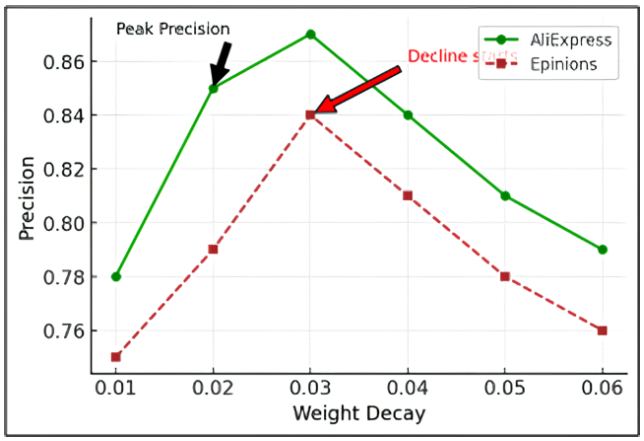


FIGURE 5. Impact of weight decay on RAGX11Rec over AliExpress and Epinions.

decay helps by slightly penalizing the weights, preventing overfitting and improving performance on unseen data. These results demonstrate the importance of tuning weight decay to balance the models' ability to generalize without over- or underfitting, with the best performance occurring at moderate levels of regularization.

4) RAGX11REC MODEL LEARNING ON ALIEXPRESS: TRAINING AND VALIDATION LOSS

Figure 6 shows the training and validation loss over 100 epochs, showing a typical learning curve for the proposed model. Both losses start high, with the training loss initially over 20, but they decrease rapidly within the first 20 epochs, indicating that the model is learning effectively. After this point, the losses plateau, with the training loss stabilizing around 2.0. The validation loss follows a similar trend and remains close to the training loss, suggesting that the model generalizes well and is not overfitting. The fact that both curves stay aligned without significant divergence further confirms this. The overall learning process appears stable, indicating well-tuned hyperparameters, such as the learning rate and regularization, ensuring a balanced performance on training and validation data.

5) RAGX11REC MODEL LEARNING ON EPINIONS: TRAINING AND VALIDATION ACCURACY

Figure 7 shows the accuracy of training and validation over 100 epochs. The training accuracy increases rapidly during the first 20 epochs, reaching nearly 100%, indicating that the model quickly learns and fits the training data. The

TABLE 4. Performance comparison of RAGX11Rec with baselines for all users on AliExpress dataset.

	SASRec	LRURec	GPTRec	NARM	BERT4Rec	RAGX11Rec
MRR@5	0.081	0.2863	0.3109	0.3286	0.3152	0.3207
NDCG@5	0.1556	0.1812	0.2095	0.3917	0.3697	0.3874
Recall@5	0.3083	0.4023	0.3663	0.4919	0.4367	0.4971
MRR@10	0.2534	0.0744	0.3651	0.4953	0.4731	0.4819
NDCG@10	0.3155	0.1161	0.2780	0.4735	0.4183	0.4469
Recall@10	0.4819	0.2794	0.7910	0.5345	0.8189	0.8622

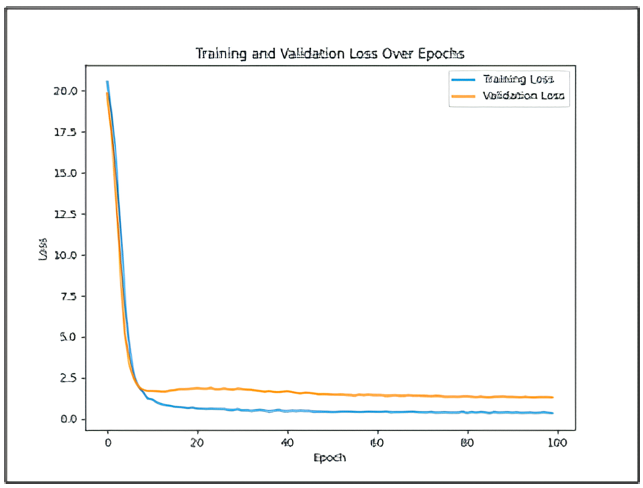


FIGURE 6. RAGX11Rec model training and validation loss on AliExpress.

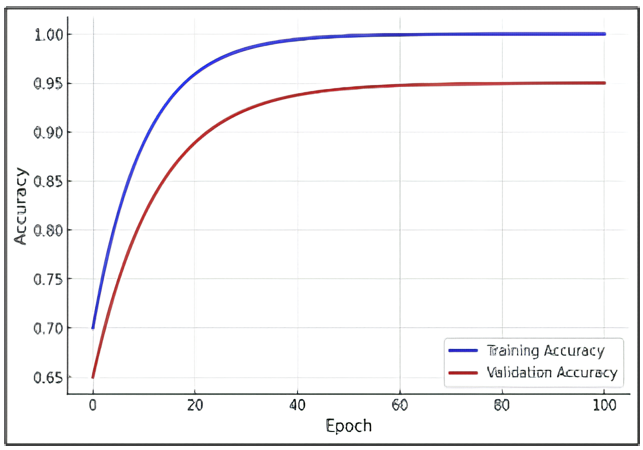


FIGURE 7. RAGX11Rec model training and validation accuracy on Epinions.

curve then flattens as it approaches a value close to 1, suggesting minimal further improvement beyond epoch 40. The validation accuracy also increases initially; however, it stabilizes around 95%, slightly lower than the training accuracy, indicating that the model generalizes well.

D. COMPARISON WITH BASELINES

1) EVALUATING RAGX11REC FOR 'ALL USERS' ON ALIEXPRESS DATASET

We evaluated the performance of RAGX11Rec model for 'All Users' views of data on AliExpress dataset with baseline methods. Table 4 shows the comparative evaluation of the proposed model and baselines. SASRec has the lowest performance across most metrics compared to other models, with MRR, NDCG, and Recall consistently lower. LRURec performs better than SASRec but generally lags behind other models like NARM, GPTRec, BERT4Rec, and RAGX11Rec. NARM has relatively high Recall (R@5 = 0.4919 and R@10 = 0.5452), indicating that it retrieves more relevant items within the top-5 and top-10 recommendations than other models. BERT4Rec performs well overall, especially in recall at both cut-off points (R@5 = 0.4367 and R@10 = 0.8189), showing that it can retrieve relevant items efficiently. RAGX11Rec appears to be the best-performing model with the highest recall. The MRR and NDCG scores of proposed models are also high, suggesting they retrieve relevant items and rank them well.

2) EVALUATING RAGX11REC FOR 'COLD START USERS' ON ALIEXPRESS DATASET

Table 5 presented the performance evaluation for models like SASRec, LRURec, GPTRec, NARM, BERT4Rec, and RAGX11Rec on cold start users for AliExpress dataset, including MRR@5, NDCG@5, Recall@5, MRR@10, NDCG@10, and Recall@10. Among these models, RAGX11Rec delivers the most robust performance, especially with a Recall@10 of 0.4621, and also excels in MRR@10 and NDCG@10. BERTRec follows closely, with a high Recall@10 of 0.4487. NARM offers balanced performance across all metrics, while LRURec, GPTRec and SASRec lag behind. Overall, RAGX11Rec and BERT4Rec are the top performers, demonstrating strong retrieval and ranking capabilities.

3) EVALUATING RAGX11REC FOR 'ALL USERS' AND 'COLD START USERS' ON EPINIONS DATASET

Tables 6 and 7 present the performance of RAGX11Rec compared with baseline models for MRR@5, NDCG@5, Recall@5, MRR@10, NDCG@10 and Recall@10 on the Epinions dataset for both 'All Users' and 'Cold Start Users'. The results reported in Table 6 demonstrate that RAGX11Rec outperforms all baselines across all metrics for 'All Users', which indicates its far superior accuracy. It results in the highest NDCG@10 of 0.454, and Recall@10 of 0.988, showing better ranking and retrieval abilities, especially compared to the second-best model BERT4Rec. However,

TABLE 5. Performance comparison of RAGX11Rec with baselines for cold start users on AliExpress dataset.

	SASRec	LRURec	GPTRec	NARM	BERT4Rec	RAGX11Rec
MRR@5	0.1313	0.1869	0.2150	0.2288	0.2158	0.2409
NDCG@5	0.1145	0.1358	0.1971	0.2519	0.2694	0.2876
Recall@5	0.2039	0.3025	0.2636	0.3918	0.3361	0.3951
MRR@10	0.1536	0.1742	0.2823	0.3253	0.3735	0.3927
NDCG@10	0.2154	0.3120	0.1781	0.3451	0.3156	0.3265
Recall@10	0.3817	0.3792	0.6966	0.4153	0.4487	0.4621

TABLE 6. Performance comparison of RAGX11Rec with baselines for all users on Epinions dataset.

	SASRec	LRURec	GPTRec	NARM	BERT4Rec	RAGX11Rec
MRR@5	0.127	0.053	0.288	0.254	0.290	0.351
NDCG@5	0.291	0.111	0.283	0.354	0.325	0.426
Recall@5	0.433	0.259	0.387	0.554	0.374	0.591
MRR@10	0.376	0.098	0.356	0.403	0.373	0.454
NDCG@10	0.419	0.098	0.319	0.221	0.372	0.454
Recall@10	0.512	0.284	0.745	0.633	0.751	0.988

TABLE 7. Performance comparison of RAGX11Rec with baselines for cold start users on Epinions dataset.

	SASRec	LRURec	GPTRec	NARM	BERT4Rec	RAGX11Rec
MRR@5	0.219	0.076	0.425	0.244	0.255	0.323
NDCG@5	0.297	0.101	0.387	0.451	0.303	0.451
Recall@5	0.379	0.321	0.456	0.500	0.354	0.548
MRR@10	0.314	0.074	0.133	0.421	0.344	0.522
NDCG@10	0.319	0.203	0.133	0.320	0.373	0.554
Recall@10	0.512	0.288	0.832	0.473	0.754	0.988

some baselines like GPTRec achieve a relatively higher MRR, indicating they can also perform with quality ranking.

Table 7 describes the performance of proposed model for Cold Start Users. RAGX11Rec outperforms all methods over most metrics, achieving more than 9% gain, for NDCG@5 (0.527) and MRR@10 (0.477). Its Recall@10 is lower than for all users (0.414), which may indicate cold-start problems. GPTRec achieves competitive MRR and NDCG with reasonable MRR and NDCG, suggesting a high robustness. BERT4Rec performs well but is outperformed by RAGX11Rec, especially in recall metrics.

4) EXECUTION TIME ANALYSIS OF PROPOSED MODEL WITH BASELINES ON ALIEXPRESS DATASET

We conducted more experiments for measuring average recommendation time through the proposed RAGX11Rec setup against state-of-the-art models including SASRec, BERT4Rec, GPTRec, and NARM under equivalent hardware and data conditions for fair evaluation. RAGX11Rec achieved better accuracy and cold-start capabilities at the expense of moderate computational cost arising from its retrieval stage and re-ranking operation combined with transformer-based generation process. The execution duration of RAGX11Rec amounts to 79 milliseconds per recommendation instance while BERT4Rec takes 92 milliseconds and both SASRec and NARM execute in 78 milliseconds and 83 milliseconds respectively. GPTRec executes recommendations in similar periods as it implements autoregressive generation which demands about 135 ms per instance. These results confirm that the framework offers a favorable trade-off between efficiency and performance.

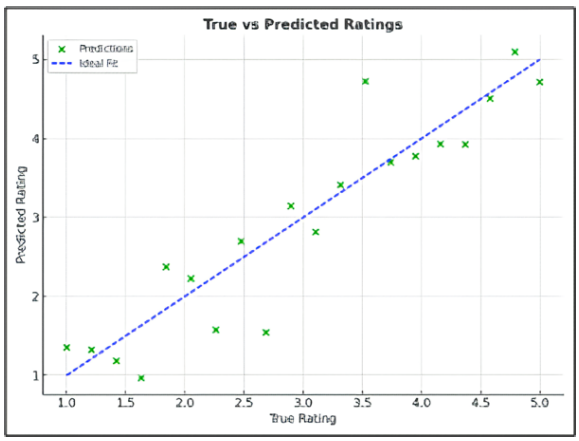


FIGURE 8. Evaluation of RAGX11Rec model with zero-shot learning for 'All Users' on AliExpress.

5) EVALUATING OF RAGX11REC WITH ZERO-SHOT LEARNING FOR 'ALL USERS' ON ALIEXPRESS DATASET

Figure 8 shows the prediction of ratings on the AliExpress dataset using the proposed model via Zero-shot learning. The blue dashed line shows the 'ideal fit', where predictions capture exactly the true values. The majority of the predictions, as denoted by green crosses, closely follow this line,

indicating that RAGX11Rec works well in zero-shot settings and predicts ratings well for new users. It suggests that the model can demonstrate a well-generalized feature space that generates recommendations without training.

V. CONCLUSION

Recommender systems play a vital role in increasing the productivity of e-commerce sites. Recommender systems based on e-commerce face challenges like user cold start, data sparsity and falsified feedback (preferences). Large language and transformer models have recently been the primary focus of solving cold start problems by utilizing external contextual data and transformer-based feedback mechanisms to generate accurate and personalized recommendations, even when historical data is unavailable. In this paper, we developed a framework called RAGX11Rec for generating top-N recommendations based on RankRAG and transformer models. The RankRAG model retrieves relevant contexts for each user in the form of preferences using a retrieval-augmented approach and classifies the top-k preferences. After that, we feed the top-k preferences in the form of user feedback integrated into the process, with patch embeddings passing through multiple transformer layers to predict the rating for the cold start user. Our proposed model leverages textual features to understand user preferences better and is specifically designed to accelerate inference using RankRAG. RAGX11Rec framework is designed to leverage external contextual data and user preferences to generate top-N recommendations, even when historical data is unavailable. The RankRAG model refines retrieved contexts to prioritize relevance, and the transformer model incorporates these ranked preferences to enhance recommendation quality. This hybrid approach ensures scalability and efficiency, making it particularly effective in data-sparse environments. We validated the effectiveness and efficiency of RAGX11Rec through experiments on the AliExpress and Epinions datasets, where it consistently outperforms state-of-the-art baselines. The results show that RAGX11Rec consistently outperforms other models across all metrics, such as M@5, N@5, R@5, M@10, N@10, and R@10, demonstrating its superior ability to provide accurate and relevant recommendations. It achieves the highest scores, indicating intense precision and recall at top-5 and top-10 recommendation levels. RAGX11Rec performance improvements, particularly in cold-start scenarios, make it a valuable tool for e-commerce platforms seeking to improve user experience. In future, we will generate LLM-based recommendations by integrating trust information to enhance user experience. Also by integrating GraphRAG into our proposed RAGX11Rec framework, we can enhance the contextual understanding and coherence of retrieved preferences by modeling interconnections among user feedback and item metadata using a graph structure. In real-world recommendation scenarios, users often interact with multiple related items or categories (e.g., electronics and accessories), which may not be fully captured through linear retrieval and re-ranking. By incorporating a lightweight graph construction layer, RAGX11Rec could benefit from identifying semantic or behavioral relationships across retrieved contexts before ranking them. This would help improve the contextual richness of the input to the transformer model and potentially lead to more diverse and contextually consistent recommendations. On the other hand, concepts from LightRAG can contribute to optimizing the efficiency and scalability of RAGX11Rec, particularly when deployed in real-time e-commerce environments where latency is critical. By adopting LightRAG's principles of retriever and generator decoupling and using smaller, domain-specific retrievers, we can reduce the computational cost of initial candidate generation and re-ranking. This could be particularly useful in scaling RAGX11Rec for mobile platforms or low-resource edge devices without compromising personalization.

ACKNOWLEDGMENT

This research was supported by Princess Nourah bint Abdulrahman University and Researchers Supporting Project number (PNURSP2025R346). The authors would like to acknowledge the support of Artificial Intelligence and Data Analytics (AIDA) Lab College of Computer and Information Science (CCIS) Prince Sultan University, Riyadh Saudi Arabia for APC support.

REFERENCES

[1] A. Ahmed, K. Saleem, U. Rashid, and A. Baz, "Modeling trust-aware recommendations with temporal dynamics in social networks," *IEEE Access*, vol. 8, pp. 149676–149705, 2020.
[2] S. Lee, "Time-aware similarity integration for user-based collaborative filtering," *J. Comput. Cognit. Eng.*, vol. 3, no. 3, pp. 285–294, Apr. 2024.
[3] R. Sharma, D. Gopalani, and Y. Meena, "Collaborative filtering-based recommender system: Approaches and research challenges," in *Proc. 3rd Int. Conf. Comput. Intell. Commun. Technol. (CICT)*, Feb. 2017, pp. 1–6.
[4] A. Ahmed, K. Saleem, O. Khalid, and U. Rashid, "On deep neural network for trust aware cross domain recommendations in e-commerce," *Expert Syst. Appl.*, vol. 174, Jul. 2021, Art. no. 114757.
[5] Z. Sun, L. Han, W. Huang, X. Wang, and X. Zeng, "Recommender systems based on social networks," *J. Syst. Softw.*, vol. 99, pp. 109–119, Jan. 2015.
[6] J. P. Pereyra, S. Hernández, and L. Terán, "TWITCHCOMM: A community-based recommender system for twitch users," in *Proc. 11th IEEE Swiss Conf. Data Sci. (SDS)*, May 2024, pp. 151–158.
[7] G. Adomavicius, K. Bauman, A. Tuzhilin, and M. Unger, "Context-aware recommender systems: From foundations to recent developments," in *Recommender Systems Handbook*. New York, NY, USA: Springer, 2022, pp. 211–250.
[8] K. Bao, J. Zhang, Y. Zhang, W. Wang, F. Feng, and X. He, "TALLRec: An effective and efficient tuning framework to align large language model with recommendation," 2023, *arXiv:2305.00447*.
[9] F. Yang, Z. Chen, Z. Jiang, E. Cho, X. Huang, and Y. Lu, "PALR: Personalization aware LLMs for recommendation," 2023, *arXiv:2305.07622*.
[10] Y. Hou, J. Zhang, Z. Lin, H. Lu, R. Xie, J. McAuley, and W. X. Zhao, "Large language models are zero-shot rankers for recommender systems," 2023, *arXiv:2305.08845*.
[11] W.-C. Kang, J. Ni, N. Mehta, M. Sathiamoorthy, L. Hong, E. Chi, and D. Zhiyuan Cheng, "Do LLMs understand user preferences? Evaluating LLMs on user rating prediction," 2023, *arXiv:2305.06474*.
[12] L. Li, Y. Zhang, and L. Chen, "Prompt distillation for efficient LLM-based recommendation," *Training*, vol. 1, p. P1, Jan. 2023.
[13] J. Liu, C. Liu, P. Zhou, R. Lv, K. Zhou, and Y. Zhang, "Is ChatGPT a good recommender? A preliminary study," 2023, *arXiv:2304.10149*.
[14] L. Wang and E.-P. Lim, "Zero-shot next-item recommendation using large pretrained language models," 2023, *arXiv:2304.03153*.
[15] J. Zhang, R. Xie, Y. Hou, W. Xin Zhao, L. Lin, and J.-R. Wen, "Recommendation as instruction following: A large language model empowered recommendation approach," 2023, *arXiv:2305.07001*.

[16] H. Touvron et al., "Llama 2: Open foundation and fine-tuned chat models," 2023, *arXiv:2307.09288*.

[17] M. Glass, G. Rossiello, M. F. M. Chowdhury, A. Naik, P. Cai, and A. Gliozzo, "Re2G: Retrieve, rerank, generate," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2022, pp. 2701–2715.

[18] O. Ram, Y. Levine, I. Dalmedigos, D. Muhlgay, A. Shashua, K. Leyton-Brown, and Y. Shoham, "In-context retrieval-augmented language models," *Trans. Assoc. Comput. Linguistics*, vol. 11, pp. 1316–1331, Nov. 2023.

[19] M. Khalifa, L. Logeswaran, M. Lee, H. Lee, and L. Wang, "Few-shot reranking for multi-hop QA via language model prompting," in *Proc. 61st Annu. Meeting Assoc. Comput. Linguistics*, 2023, pp. 15882–15897.

[20] Z. Qin, R. Jagerman, K. Hui, H. Zhuang, J. Wu, L. Yan, J. Shen, T. Liu, J. Liu, D. Metzler, X. Wang, and M. Bendersky, "Large language models are effective text rankers with pairwise ranking prompting," in *Proc. Findings Assoc. Comput. Linguistics, NAACL*, 2024, pp. 15882–15897.

[21] W. Sun, L. Yan, X. Ma, S. Wang, P. Ren, Z. Chen, D. Yin, and Z. Ren, "Is ChatGPT good at search? Investigating large language models as re-ranking agents," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2023, pp. 2701–2715.

[22] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, and T.-S. Chua, "Attentive collaborative filtering: Multimedia recommendation with item- and component-level attention," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 2017, pp. 335–344.

[23] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, and T.-S. Chua, "Attentional factorization machines: Learning the weight of feature interactions via attention networks," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 3119–3125.

[24] S. Wang, L. Hu, L. Cao, X. Huang, D. Lian, and W. Liu, "Attention-based transactional context embedding for next-item recommendation," in *Proc. 32nd AAAI Conf. Artif. Intell. (AAAI)*, 2018, pp. 2532–2539.

[25] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," in *Recommender Systems Handbook*.Boston, MA, USA: Springer, 2011, pp. 1–35.

[26] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowl.-Based Syst.*, vol. 46, pp. 109–132, Jul. 2013.

[27] L. Lü, M. Medo, C. H. Yeung, Y.-C. Zhang, Z.-K. Zhang, and T. Zhou, "Recommender systems," *Phys. Rep.*, vol. 519, no. 1, pp. 1–49, 2012.

[28] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[29] P. Xu, W. Ping, X. Wu, L. McAfee, C. Zhu, Z. Liu, S. Subramanian, E. Bakhturina, M. Shoeybi, and B. Catanzaro, "Retrieval meets long context large language models," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2024, pp. 1–22.

[30] O. Yoran, T. Wolfson, O. Ram, and J. Berant, "Making retrieval-augmented language models robust to irrelevant context," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2024, pp. 1–20.

[31] W. Yu, H. Zhang, X. Pan, K. Ma, H. Wang, and D. Yu, "Chain-of-note: Enhancing robustness in retrieval-augmented language models," 2023, *arXiv:2311.09210*.

[32] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Comput. Surveys*, vol. 52, no. 1, pp. 1–38, Feb. 2019.

[33] S. M. J. Jalali, M. Ahmadian, S. Ahmadian, A. Khosravi, M. Alazab, and S. Nahavandi, "An oppositional-cauchy based GSK evolutionary algorithm with a novel deep ensemble reinforcement learning strategy for COVID-19 diagnosis," *Appl. Soft Comput.*, vol. 111, Nov. 2021, Art. no. 107675.

[34] J. Liu and C. Wu, "Deep learning-based recommendation: A survey," in *Proc. Int. Conf. Inf. Sci. Appl. (ICISA)*, Springer, Singapore. Springer, 2017, pp. 451–458.

[35] Y. Pan, F. He, and H. Yu, "A correlative denoising autoencoder to model social influence for top-N recommender system," *Frontiers Comput. Sci.*, vol. 14, no. 3, pp. 1–13, Jun. 2020.

[36] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for Web search using clickthrough data," in *Proc. 22nd ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2013, pp. 2333–2338.

[37] C.-K. Hsieh, L. Yang, Y. Cui, T.-Y. Lin, S. Belongie, and D. Estrin, "Collaborative metric learning," in *Proc. 26th Int. Conf. World Wide Web*, Apr. 2017, pp. 193–201.

[38] A. Taleb, S. Jiang, S. Wang, and Y. Fu, "Examples-rules guided deep neural network for makeup recommendation," in *Proc. 31st AAAI Conf. Artif. Intell. (AAAI)*, 2017, vol. 31, no. 1, pp. 1725–1731.

[39] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "DeepFM: A factorization-machine based neural network for CTR prediction," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 1725–1731.

[40] T. B. Brown et al., "Language models are few-shot learners," in *Proc. NIPS*, 2020, pp. 1877–1901.

[41] A. Chowdhery et al., "PaLM: Scaling language modeling with pathways," 2022, *arXiv:2204.02311*.

[42] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," Tech. Rep., 2018.

[43] A. Roberts, C. Raffel, K. Lee, M. Matena, N. Shazeer, P. J. Liu, S. Narang, W. Li, and Y. Zhou, "Exploring the limits of transfer learning with a unified text-to-text transformer," 2019, *arXiv:1910.10683*.

[44] R. Thoppilan et al., "LaMDA: Language models for dialog applications," 2022, *arXiv:2201.08239*.

[45] J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le, "Fine-tuned language models are zero-shot learners," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2022, pp. 1–25.

[46] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 5998–6008.

[47] J. Li, M. Wang, J. Li, J. Fu, X. Shen, J. Shang, and J. McAuley, "Text is all you need: Learning language representations for sequential recommendation," 2023, *arXiv:2305.13731*.

[48] D. Sileo, W. Vossen, and R. Raymaekers, "Zero-shot recommendation as language modeling," in *Proc. Eur. Conf. Inf. Retr. (ECIR)*. Cham, Switzerland: Springer, 2022, pp. 223–230.

[49] Y. Wang, Z. Jiang, Z. Chen, F. Yang, Y. Zhou, E. Cho, X. Fan, X. Huang, Y. Lu, and Y. Yang, "RecMind: Large language model powered agent for recommendation," 2023, *arXiv:2308.14296*.

[50] Y. Gao, T. Sheng, Y. Xiang, Y. Xiong, H. Wang, and J. Zhang, "Chat-REC: Towards interactive and explainable LLMs-augmented recommender system," 2023, *arXiv:2303.14524*.

[51] Z. Chen, "PALR: Personalization-aware LLMs for recommendation," 2023, *arXiv:2305.07622*.

[52] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-T. Yih, "Dense passage retrieval for open-domain question answering," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2020, pp. 6769–6781.

[53] W. Shi, S. Min, M. Yasunaga, M. Seo, R. James, M. Lewis, L. Zettlemoyer, and W.-T. Yih, "REPLUG: Retrieval-augmented black-box language models," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Human Lang. Technol.*, 2024, pp. 1–15.

[54] X.V. Lin, X. Chen, M. Chen, W. Shi, M. Lomeli, R. James, P. Rodriguez, J. Kahn, G. Szilvasy, M. Lewis, L. Zettlemoyer, and W.-T. Yih, "RA-DIT: Retrieval-augmented dual instruction tuning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2024, pp. 1–30.

[55] H. Trivedi, N. Balasubramanian, T. Khot, and A. Sabharwal, "Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions," in *Proc. 61st Annu. Meeting Assoc. Comput. Linguistics*, 2023, pp. 1–16.

[56] Z. Jiang, F. F. Xu, L. Gao, Z. Sun, Q. Liu, J. Dwivedi-Yu, Y. Yang, J. Callan, and G. Neubig, "Active retrieval augmented generation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2023, pp. 1–20.

[57] S. Jeong, J. Baek, S. Cho, S. J. Hwang, and J. Park, "Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Human Lang. Technol.*, 2024, pp. 1–15.

[58] B. Wang, W. Ping, L. McAfee, P. Xu, B. Li, M. Shoeybi, and B. Catanzaro, "InstructRetro: Instruction tuning post retrieval-augmented pre-training," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2024, pp. 1–25.

[59] Y. Zhu, P. Zhang, C. Zhang, Y. Chen, B. Xie, Z. Liu, J.-R. Wen, and Z. Dou, "INTERS: Unlocking the power of large language models in search with instruction tuning," in *Proc. 62nd Annu. Meeting Assoc. Comput. Linguistics*, 2024, pp. 2782–2809.

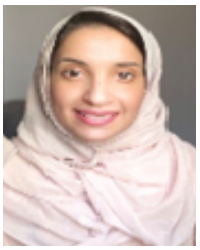
[60] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, "Self-RAG: Learning to retrieve, generate, and critique through self-reflection," in *Proc. Int. Conf. Learn. Represent. ICLR)*, 2024, pp. 1–30.

[61] K. Xu, J. Ba, R. Kiros, K. Cho, A. C. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2015, pp. 2048–2057.

[62] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–15.
[63] Y. Wu et al., "Google's neural machine translation system: Bridging the gap between human and machine translation," 2016, *arXiv:1609.08144*.
[64] J. Zhou, Y. Cao, X. Wang, P. Li, and W. Xu, "Deep recurrent models with fast-forward connections for neural machine translation," *Trans. Assoc. Comput. Linguistics*, vol. 4, pp. 371–383, Jul. 2016.
[65] T. Zhang, S. G. Patil, N. Jain, S. Shen, M. Zaharia, I. Stoica, and J. E. Gonzalez, "RAFT: Adapting language model to domain specific RAG," 2024, *arXiv:2403.10131*.
[66] Z. Liu, W. Ping, R. Roy, P. Xu, C. Lee, M. Shoeybi, and B. Catanzaro, "ChatQA: Surpassing GPT-4 on conversational QA and RAG," 2024, *arXiv:2401.10225*.
[67] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ questions for machine comprehension of text," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 2383–2392.
[68] R. Nogueira, Z. Jiang, R. Pradeep, and J. Lin, "Document ranking with a pretrained sequence-to-sequence model," in *Proc. Findings Assoc. Comput. Linguistics, EMNLP*, 2020, pp. 708–718.
[69] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.
[70] A. Ahmed, K. Saleem, O. Khalid, J. Gao, and U. Rashid, "Trust-aware denoising autoencoder with spatial–temporal activity for cross-domain personalized recommendations," *Neurocomputing*, vol. 511, pp. 477–494, Oct. 2022.
[71] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation," in *Proc. ACM Conf. Inf. Knowl. Manage.*, Nov. 2017, pp. 1419–1428.
[72] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2018, pp. 197–206.
[73] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, "BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Nov. 2019, pp. 1441–1450.
[74] Z. Yue, Y. Wang, Z. He, H. Zeng, J. McAuley, and D. Wang, "Linear recurrent units for sequential recommendation," Tech. Rep., 2023.
[75] J. Wang, A. P. de Vries, and M. J. T. Reinders, "Unifying user-based and item-based collaborative filtering approaches by similarity fusion," in *Proc. 29th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 2006, pp. 501–508.
[76] A. V. Petrov and C. Macdonald, "Generative sequential recommendation with GPTRec," 2023, *arXiv:2306.11114*.
[77] S. Gheewala, S. Xu, S. Yeom, and S. Maqsood, "Exploiting deep transformer models in textual review-based recommender systems," *Expert Syst. Appl.*, vol. 233, May 2024, Art. no. 117432.
[78] Z. Zhao, W. Fan, J. Li, Y. Liu, and X. Mei, "Recommender systems in the era of large language models," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 1, pp. 345–358, Jan. 2024.
[79] A. V. Petrov, "Effective and efficient transformer models for sequential recommendation," in *Proc. ACM Int. Conf. Recommender Syst. (RecSys)*, 2024, vol. 17, no. 1, pp. 102–113.
[80] M. S. Reddy and H. Karnati, "Transformer-based federated learning models for recommendation systems," *IEEE Access*, vol. 12, pp. 10253–10266, 2024.
[81] G. S. P. Moreira, S. Rabhi, and J. M. Lee, "Transformers4Rec: Bridging the gap between NLP and session-based recommendation," in *Proc. ACM Conf. Recommender Syst. (RecSys)*, 2021, vol. 15, no. 1, pp. 345–358.
[82] H. Wang, J. Lian, M. Wu, H. Li, J. Fan, W. Xu, C. Li, and X. Xie, "ConvFormer: Revisiting transformer for sequential user modeling," 2023, *arXiv:2308.02925*.
[83] J. Tang, H. Gao, H. Liu, and A. D. Sarma, "ETrust: Understanding trust evolution in an online world," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2012, pp. 253–261.
[84] A. Radwan, M. Amarneh, H. Alawneh, H. I. Ashqar, A. AlSobeh, and A. A. A. R. Magableh, "Predictive analytics in mental health leveraging LLM embeddings and machine learning models for social media analysis," *Int. J. Web Services Res.*, vol. 21, no. 1, pp. 1–22, Feb. 2024.
[85] L. Yating, M. Afzaal, S. Xiao, and D. A. S. El-Dakhs, "TQFLL: A novel unified analytics framework for translation quality framework for large language model and human translation of allusions in multilingual corpora," *Automatika*, vol. 66, no. 1, pp. 91–102, Jan. 2025.
[86] M. A. T. Rony, T. Sultan, S. Alshathri, and W. El-Shafai, "MediGPT: Exploring potentials of conventional and large language models on medical data," *IEEE Access*, vol. 12, pp. 103473–103487, 2024.
[87] A. Ammar, A. Koubaa, B. Benjdira, O. Nacar, and S. Sibaee, "Prediction of Arabic legal rulings using large language models," *Electronics*, vol. 13, no. 4, p. 764, Feb. 2024.

FATEN S. ALAMRI received the Ph.D. degree in system modeling and analysis in statistics from Virginia Commonwealth University, USA, in 2020. She is currently an Assistant Professor with the Department of Mathematical Sciences, College of Science, Princess Nourah bint Abdulrahman University. Her Ph.D. research included Bayesian dose-response modeling, experimental design, and nonparametric modeling. Her research interests include spatial area, environmental statistics, and brain imaging.

AMJAD REHMAN (Senior Member, IEEE) received the Ph.D. degree in image processing and pattern recognition from Universiti Teknologi Malaysia, Malaysia, in 2010. During the Ph.D. study, he proposed novel techniques for pattern recognition based on novel feature mining strategies. He is currently conducting research under the supervision of three Ph.D. students. He is the author of dozens of papers published in international journals and conferences of high repute. His research interests include information security, data mining, and document analysis and recognition.



BAYAN ALGHOFAILY received the master's and Ph.D. degrees in computer science from Toronto Metropolitan University, Toronto, Canada. During that period, she was a member of the Distributed Applications and Broadband Networks Laboratory (DABNEL). She focused on studying how the performance of machine learning models is affected by dataset features. She is currently an Assistant Professor with the Department of Information Systems, CCIS, Prince Sultan University (PSU). She is also a member of the Artificial Intelligence and Data Analytics (AIDA) Laboratory, CCIS, PSU. She continues to explore this further in her research. Her research interests include AI, NLP, ML, and neural networks.

ADEEL AHMED received the Ph.D. degree in computer science from Quaid-i-Azam University, Islamabad, Pakistan, in 2022. He was with the software industry for some years. His research interests include swarm intelligence, machine learning, recommendation systems, social network analysis, and information visualization.

KHALID SALEEM received the M.Sc. degree in computer science from Quaid-i-Azam University, Pakistan, in 1994, and the M.Phil. and Ph.D. degrees in computer science from the University of Montpellier 2, France, in 2005 and 2008, respectively. He was with the software industry for some years. He is currently an Assistant Professor with Quaid-i-Azam University, Pakistan. He is the President of PAK-France Alumni Network. His research interests include machine learning, deep learning, schema matching and integration, data analytics, steganography, cryptography, and mobile computing.