EmbSum: Leveraging the Summarization Capabilities of Large Language Models for Content-Based Recommendations

Chiyu Zhang*† University of British Columbia, Canada
Yifei Sun* Meta AI, USA
Minghao Wu Monash University, Australia
Jun Chen Meta AI, USA
Jie Lei Meta AI, USA
Muhammad Abdul-Mageed University of British Columbia, Canada MBZUAI, UAE
Rong Jin Meta AI, USA
Angli Liu Meta AI, USA
Ji Zhu Meta AI, USA
Sem Park Meta AI, USA
Ning Yao Meta AI, USA
Bo Long Meta AI, USA

Abstract

Content-based recommendation systems play a crucial role in delivering personalized content to users in the digital world. In this work, we introduce EmbSum, a novel framework that enables offline pre-computations of users and candidate items while capturing the interactions within the user engagement history. By utilizing the pretrained encoder-decoder model and poly-attention layers, EmbSum derives User Poly-Embedding (UPE) and Content Poly-Embedding (CPE) to calculate relevance scores between users and candidate items. EmbSum actively learns the long user engagement histories by generating user-interest summary with supervision from large language model (LLM). The effectiveness of EmbSum is validated on two datasets from different domains, surpassing state-of-the-art (SoTA) methods with higher accuracy and fewer parameters. Additionally, the model's ability to generate summaries of user interests serves as a valuable by-product, enhancing its usefulness for personalized content recommendations.

CCS Concepts

• **Information systems** → **Content ranking**; • **Computing methodologies** → *Natural language generation.*

Keywords

Recommendation System, User Interest Summarization, Large Language Model

*Corresponding Authors: chiyuzh@mail.ubc.ca; sunyifei@meta.com
†Work done during Meta internship.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
RecSys '24, October 14–18, 2024, Bari, Italy
© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-0505-2/24/10
https://doi.org/10.1145/3640457.3688185

ACM Reference Format:
Chiyu Zhang, Yifei Sun, Minghao Wu, Jun Chen, Jie Lei, Muhammad Abdul-Mageed, Rong Jin, Angli Liu, Ji Zhu, Sem Park, Ning Yao, and Bo Long. 2024. EmbSum: Leveraging the Summarization Capabilities of Large Language Models for Content-Based Recommendations. In *18th ACM Conference on Recommender Systems (RecSys '24), October 14–18, 2024, Bari, Italy.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3640457.3688185

1 Introduction

In the thriving digital world, billions of users interact daily with a diverse range of digital content, encompassing news, social media updates, e-books, etc. Content-based recommendation systems, as discussed in various studies [4, 16–18], leverage the textual content, such as news articles and books, and the sequence of a user's interaction history. It facilitates the delivery of content recommendations that are more precise, relevant, and customized to each user's preferences.

Recent studies have successfully incorporated Pretrained Language Models (PLMs) [10, 15, 21] into recommendation systems for processing textual inputs [13, 14, 30]. This integration has significantly improved the efficiency of content-based recommendations. Due to the well-known memory limitation of the attention mechanism, previous studies [11, 33] typically encode each piece of user historical content separately and then aggregate them. This approach, however, falls short in modeling the interactions among the user's historical contents. Addressing this, Mao et al. [17] introduce local and global attention mechanisms to encode user histories hierarchically. However, this method needs to truncate the history sequence to 1K tokens due to the limitation of PLMs, thus diminishing the benefits of leveraging extensive engagement history to capture users' comprehensive interests. In another vein, to improve the alignment between user and candidate content, many studies have directly integrated candidate items into user modeling [11, 20, 32]. This online real-time strategy prevents recommendation systems from performing offline pre-computations for efficient inference, which restricts their real-world applications.

To tackle the aforementioned challenges, we introduce a new framework, EmbSum, which enables offline pre-computations of

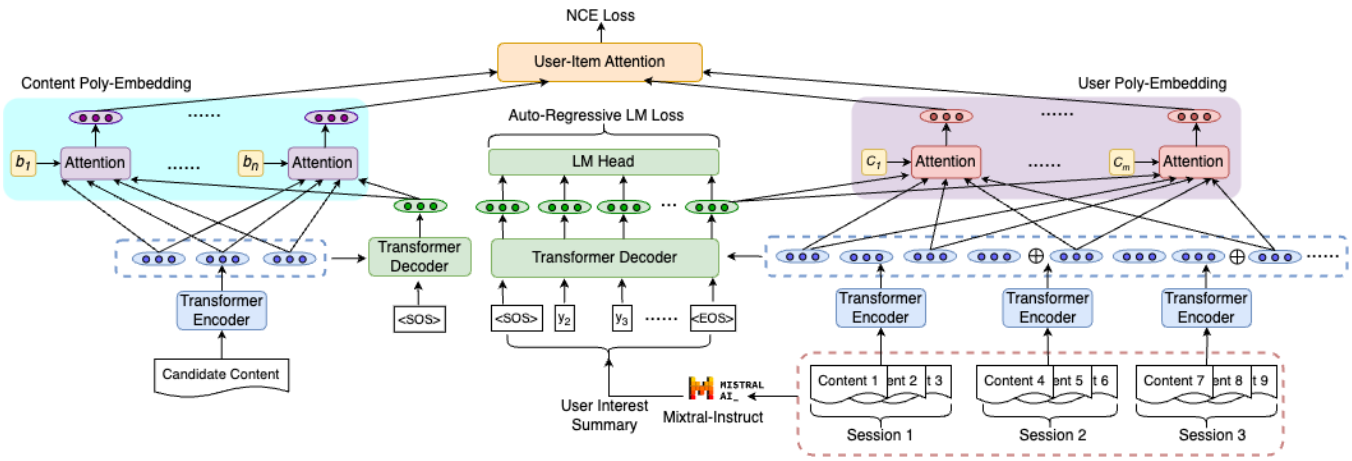

Figure 1: Overview of our EmbSum framework. Note that the user summaries generated by LLMs are only used in training.

embeddings of users and candidate items while capturing the interactions within the user's long engagement history. As Figure 1 shows, we utilize poly-attention [5] layers to derive multiple embeddings for the detailed features of both users and candidate items, referred to as User Poly-Embedding (UPE) and Content Poly-Embedding (CPE), respectively. These embeddings are then used to calculate the relevance scores between users and candidate items. More specifically, we use the pretrained T5 encoder [21] to encode user engagement sessions independently. We hypothesize that merely concatenating the embeddings of history sequences does not effectively model the interactions between user engagement sessions. To address this, we use the T5 decoder to fuse session-based encoded sequences by training it to generate user-interest summaries, supervised by the large language models (LLMs) [9] generated summaries of user's holistic interests when modeling user representations.

Our contributions in this work can be summarized as follows:

(1) We present a new framework, **EmbSum**, for embedding and summarizing user interests in content-based recommendation systems. This framework employs an encoder-decoder architecture to encode extensive user engagement histories and produce summaries of user interests.
(2) We validate the effectiveness of EmbSum by testing it on two popular datasets from different domains. Our approach surpasses SoTA methods, delivering higher accuracy with fewer parameters.
(3) Our model can generate summaries of user interests, which serves as a beneficial by-product, thereby enhancing its usefulness for personalized content recommendations.

2 Methodology

In this section, we first describe the problem formulation of our work (Section 2.1). Then, we provide an overview of our EmbSum framework (Section 2.2). Next, we introduce the details of modeling user engagements (Section 2.3), candidate contents (Section 2.4), and the click-through predictor and training objectives (Section 2.5).

2.1 Problem Formulation

Given a user u_i and a candidate content item $\bar{e}_j$ (such as news articles or books), the objective is to derive a relevance score s^i_j, which indicates the likelihood of user u_i engaging with (e.g., clicking on) the content item $\bar{e}_j$. Considering a set of candidate contents $C = \{\bar{e}_1, \bar{e}_2, \ldots, \bar{e}_j\}$, these contents are ranked based on their relevance scores $\{s^i_1, s^i_2, \ldots, s^i_j\}$ for user u_i. It is crucial to effectively extract user interests from their engagement history. User u_i is characterized by a sequence of k historically engaged contents E_{u_i} (such as browsed news articles or positively rated books), sorted in descending order by engagement time.

2.2 Overview of EmbSum

Figure 1 presents an overview of our proposed model, EmbSum. Both the user engagements and the candidate contents are encoded using a pretrained encoder-decoder Transformer model. The click-through rate (CTR) predictions between the users and the candidate content are modeled using noisy contrastive estimation (NCE) loss. Moreover, we also introduce a user-interest summarization objective that is supervised by LLM generations. By leveraging both poly-embedding of user engagement history and candidate content, EmbSum can identify patterns and preferences from user interactions and match them with the most suitable content.

2.3 User Engagement Modeling

Session Encoding. The user engagement history, denoted as E_{u_i}, comprises a sequence of k content items that a user has previously engaged with. To address the high memory demands of processing long sequences with attention mechanisms, we partition these k content items into g distinct sessions, represented as $E_{u_i} = \{\eta_1, \eta_2, \ldots, \eta_g\}$. Each session η_i encapsulates l tokens from p content items, expressed as $\eta_i = \{e_1, e_2, \ldots, e_p\}$. This structure is designed to reflect the user's interests over specific time periods and to improve the interactions within each session. We encode each session using a T5 encoder [21] independently. The representation

of each piece of content is then derived from the hidden state output corresponding to its first token, which is the start-of-sentence symbol [SOS]. This process yields k representation vectors.

User Engagement Summarization. We posit that merely utilizing these k representations falls short in capturing the subtleties of user interests and the dynamics among long-range engaged contents. To address this, we exploit the capabilities of SoTA LLMs to synthesize a distilled summary of user interests, given that recent studies [1, 19, 25] have demonstrated LLMs' capacity for summarizing long sequences. This summary encapsulates a rich and comprehensive perspective of user preferences. We utilize `Mixtral-8x22B-Instruct` [9] to generate these interest summaries from the engagement histories. The resulting summaries are then incorporated into the T5 decoder. Drawing inspiration from the fusion-in-decoder concept [7], we concatenate the hidden states of all tokens from all subsequences encoded in a session-based manner and input this combined sequence into the T5 decoder. We then train the model to produce a summary of the user's interests, employing the following loss:

$$\mathcal{L}_{\text{sum}} = -\sum_{j=1}^{|y_j^{u_i}|} log(p(y_j^{u_i}|E, y_{<j}^{u_i})), \tag{1}$$

where $y_j^{u_i}$ represents the summary generated for user u_i, and $|y_j^{u_i}|$ is the length of the user-interest summary.

User Poly-Embedding. Given that each session is encoded independently, a global representation for all engaged contents is also necessary. We hence acquire a global representation that is derived from the last token of the decoder output (i.e., [EOS] token), representing all user engagements collectively. We then concatenate it with the k representation vectors from session encoding to form a matrix $Z \in \mathrm{R}^{(k+1)\times d}$. With these representations of user interaction history, we employ a poly-attention layer [5] to extract the user's nuanced interests into multiple representations. The computation of each user-interest vector α_a is as follows:

$$\alpha_a = \text{softmax}\left[c_a \tanh(ZW^f)^\top\right] Z, \tag{2}$$

where $c_a \in \mathrm{R}^{1\times p}$ and $W^f \in \mathrm{R}^{d\times p}$ are the trainable parameters. We then concatenate m user-interest vectors into a matrix $A \in \mathrm{R}^{m\times d}$, which serves as the user representation, referring to as the User Poly-Embedding (UPE) in our framework.

2.4 Candidate Content Modeling

Unlike the conventional practice of using only the first token of a sequence for representation, we introduce the novel Content Poly-Embedding (CPE). Similar to UPE, this method employs a set of context codes, denoted as $\{b_1, b_2, \ldots, b_n\}$, to create multiple embeddings for a piece of candidate content. For each piece of candidate content, we generate a corresponding vector β_a through a poly-attention layer defined by the equation referenced as Equation 2, which includes a trainable parameter W^o. The resulting n vectors for candidate content are then aggregated into a matrix $B \in \mathrm{R}^{n\times d}$, providing a more nuanced representation that is expected to improve the performance of relevance scoring in the user-candidate matching predictor.

2.5 CTR Prediction and Training

CTR Prediction. To compute the relevance score s_j^i, we first establish the matching scores between the user representation embedding A_i and the candidate content representation embedding B_j. This computation is performed using the inner product, followed by flattening the resultant matrix:

$$K_j^i = \text{flatten}(A_i^\top B_j). \tag{3}$$

where $K_j^i \in R^{mn}$ is the flattened attention matrix. Subsequently, an attention mechanism is employed to aggregate the matching scores represented by this flattened vector:

$$\begin{aligned} W^p &= \text{softmax}(\text{flatten}(A \cdot \text{gelu}(BW^s)^\top)), \\ s_j^i &= W^p \cdot K_j^i, \end{aligned} \tag{4}$$

where $W^s \in R^{d\times d}$ signifies a trainable parameter matrix, $W^p \in R^{mn}$ represents the attention weights obtained after flattening and the softmax function, and s_j^i is the scaled relevance score.

Training. We follow the method of training the end-to-end recommendation models using the NCE loss [13, 28]:

$$\mathcal{L}_{\text{NCE}} = -\log\left(\frac{\exp(s_+^i)}{\exp(s_+^i) + \sum_j \exp(s_{-,j}^i)}\right), \tag{5}$$

where s_+^i represents the score of the positive sample with which the user engaged, and s_-^i represents the scores of negative samples. Therefore, the overall loss function is defined as:

$$\mathcal{L} = \mathcal{L}_{\text{NCE}} + \lambda \mathcal{L}_{\text{sum}}, \tag{6}$$

where λ is a scaling factor determined to be 0.05 based on the performance on the validation set.

3 Experiments

Baselines. We evaluate our EmbSum against a range of commonly used and SoTA neural network-based content recommendation approaches. These include methods that train text encoders from scratch, such as (1) NAML [26], (2) NRMS [27], (3) Fastformer [29], (4) CAUM [20], and (5) MINS [24]. We also consider systems that utilize PLMs, including (6) NAML-PLM, (7) UNBERT [33], (8) MINER [11], and (9) UniTRec [17]. For NAML, NRMS, Fastformer, and NAML-PLM, we employ the implementations provided by Liu [12]. The CAUM and MINS implementations are obtained from Iana et al. [6]. For UNBERT, MINER, and UniTRec, we use the original scripts released by their respective authors. More Details on these baselines and our implementation are provided in the Appendix.

Dataset. We employ two publicly available benchmark datasets for content-based recommendation. The first dataset, MIND [31], consists of user engagement logs from Microsoft News, incorporating both positive and negative labels determined by user clicks on news articles. We use the smaller version of this dataset, which includes 94K users and 65K news articles. The second dataset is obtained from Goodreads [23], which focuses on book recommendations derived from user ratings. In this dataset, ratings above 3 are considered positive labels, while ratings below 3 are regarded as negative labels. The Goodreads dataset comprises 50K users and 330K books. More details on dataset statistics are in the Appendix.

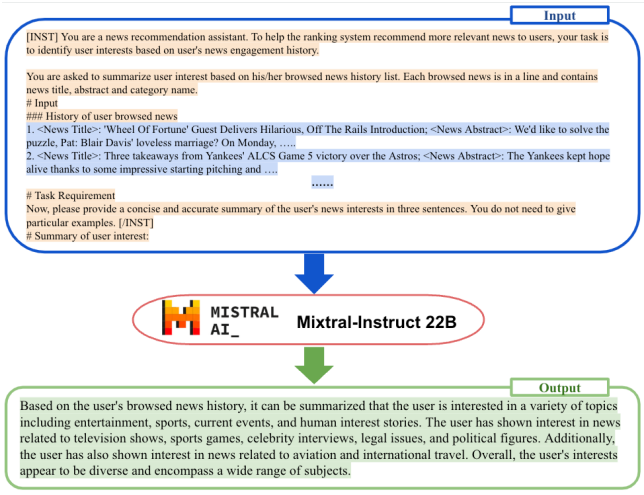

Figure 2: Illustration of using an LLM for user interest profiling. The input provided to the LLM is enclosed in a red box, and the output generated by the LLM is shown in a green box. The segment marked in orange within the input specifies the instruction for the task, whereas the portion in blue highlights the history of news browsed by the user.

Evaluation. We utilize a variety of metrics to assess the performance of content-based recommendation systems. These metrics include the classification-based metric AUC [3], and ranking-based metrics such as MRR [22] and nDCG@top$N$ (with top$N$ = 5, 10) [8]. Metric calculations are performed using the Python library TorchMetrics [2]. We determine the best model on the Dev set using AUC and report the corresponding Test performance.

Content Formatting. As an example, for each content in MIND dataset, we combine its fields into a single text sequence using the following template: "News Title: $\langle title \rangle$; News Abstract: $\langle abstract \rangle$; News Category: $\langle category \rangle$".

LLM Based User-Interest Summary. We leverage the open-source `Mixtral-8x22B-Instruct` [9] to create summaries that reflect users' interests based on their engagement history. Figure 2 illustrates an example of the input provided and the corresponding summary generated for the MIND dataset. The process starts with an instruction to frame the task, followed by a list of news items the user has viewed, ordered from the most recent to the oldest. Each item includes its title, abstract, and category. The input is limited to 60 engagement items, with extended news abstracts or book descriptions being condensed to 100 words. The instruction concludes with a request for the model to condense the user's interests into three sentences. The latter part of Figure 2 shows a sample output from the LLM. Additionally, we evaluate the LLM-generated user-interest summaries using GPT-4 (i.e., `gpt-4o` API) as a judge. The results indicate that most of the generated summaries accurately capture the user's interests. Further details of this experiment are provided in the Appendix. The average length of summaries generated by this process is 76 tokens for the MIND dataset and 115 tokens for the Goodreads dataset.

	AUC	MRR	nDCG@5	nDCG@10
MIND				
NAML	66.10	34.65	32.80	39.14
NRMS	63.28	33.10	31.50	37.68
Fastformer	66.32	34.75	33.03	39.30
CAUM	62.56	34.40	32.88	38.90
MINS	61.43	35.99	34.13	40.54
NAML-PLM	67.01	35.67	34.10	40.32
UNBERT	71.73	38.06	36.67	42.92
MINER	70.20	38.10	36.35	42.63
UniTRec	69.38	37.62	36.01	42.20
EmbSum (ours)	**71.95**	**38.58**	**36.75**	**42.97**
Goodreads				
NAML	59.35	72.16	53.49	67.81
NRMS	60.51	72.15	53.69	68.03
Fastformer	59.39	71.11	52.38	67.05
CAUM	55.13	73.06	**54.97**	69.02
MINS	53.02	71.81	53.72	68.00
NAML-PLM	59.57	72.54	53.98	68.41
UNBERT	61.40	73.34	54.67	68.71
MINER	60.72	72.72	54.17	68.42
UniTRec	60.00	72.60	53.73	67.96
EmbSum (ours)	**61.64**	**73.75**	54.86	**69.08**

Table 1: Results on MIND-small and Goodreads. The best results are highlighted in bold. The second-best results are highlighted in underscore.

Implementation. We utilize the pretrained T5-small model [21], which comprises 61M parameters. After hyperparameter tuning, we determined that the optimal codebook sizes for the UPE and CPE layers are 32 and 4, respectively, for both datasets. The model is trained with a learning rate of $5e-4$ and a batch size of 128 over 10 epochs. In all experiments, we cosnsider the latest 60 interactions as the user's engagement history. For the MIND dataset, we apply a negative sampling ratio of 4, limit news titles to 32 tokens, and restrict news abstracts to 72 tokens. Including approximately 20 additional tokens for the news category and template, each user's engaged history in MIND can total up to 7,440 tokens. In the Goodreads dataset, we set the negative sampling ratio to 2, limit book titles to 24 tokens, and constrain book descriptions to 85 tokens. Consequently, a user in the Goodreads dataset can have up to 7,740 tokens for their engagement history.

4 Results

Main Results. Table 1 shows the test results of nine baselines and our EmbSum. We can find that models initialized with PLMs obtained much better performance than baselines trained from scratch. We observe that EmbSum outperforms previous SoTA AUC scores given by UNBERT [33]. Compared to UNBERT, EmbSum achieves an improvement of 0.22 and 0.24 AUC on the MIND and Goodreads datasets, respectively. Notably, EmbSum uses only T5-small as the backbone, which has 61M parameters, significantly fewer than

MIND				
	AUC	MRR	nDCG@5	nDCG@10
Ours	**71.95**	38.58	**36.75**	**42.97**
wo CPE	68.17	36.49	33.72	40.20
wo grouping	71.34	38.29	36.41	42.55
wo UPE	71.41	**38.64**	36.70	42.90
wo $\mathcal{L}_{sum}$	71.43	38.42	36.39	42.60
Goodreads				
Ours	**61.64**	**73.75**	**54.86**	**69.08**
wo CPE	60.97	72.94	54.39	68.53
wo grouping	61.39	73.53	54.79	68.86
wo UPE	61.35	73.55	54.67	68.81
wo $\mathcal{L}_{sum}$	61.50	73.55	54.74	68.91

Table 2: Result of ablation study for EmbSum. The best results are highlighted in bold.

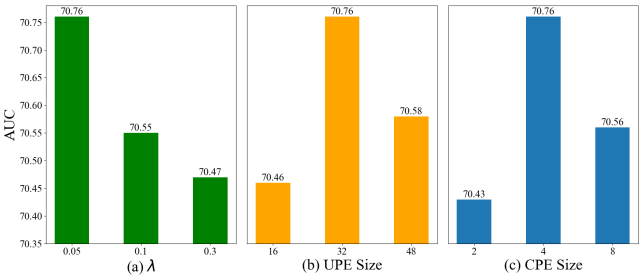

Figure 3: Influence of different hyperparameters.

	ROUGE 1	ROUGE 2	ROUGE L
MIND	51.42	30.33	39.12
Goodreads	46.99	20.86	28.16

Table 3: Evaluation on EmbSum-generated user summary.

the 125M parameters of BERT-based methods (e.g., UNBERT and MINER). On other ranking-based metrics, EmbSum achieves the best MRR and nDCG@10 scores on both datasets. EmbSum outperforms UniTRec which also uses an encoder-decoder architecture but cannot produce standalone user and candidate embeddings.

Ablation Studies. To better understand the effectiveness of our framework, we conduct ablation studies on both datasets, the results of which are presented in Table 2. We first remove the CPE for the candidate item and use only the encoder hidden state of the `[SOS]` token to represent the candidate content. This alteration consistently results in the largest performance drop on both datasets, decreasing the AUC scores by 3.78 and 0.67 on the MIND and Goodreads datasets, respectively. These results suggest the effectiveness of utilizing multiple embeddings to represent candidate content and enhance user-item interactions. Removing session-based grouping and encoding each content separately leads to AUC drops of 0.61 and 0.25 on the MIND and Goodreads datasets, respectively. When we reduce the codebook size of the UPE layer to 1, meaning each user is represented by a single vector, it results in a performance decrease of 0.54 and 0.29 AUC on the MIND and Goodreads datasets, respectively. This also justifies the efficacy of using multiple embeddings in EmbSum. Furthermore, we investigate the efficacy of using LLM-generated user-interest summaries by removing $\mathcal{L}_{sum}$. Without $\mathcal{L}_{sum}$, we only provide the `[SOS]` token as the decoder input and take the hidden state of the `[SOS]` token as the global representation. As Table 2 shows, this change results in a performance drop on both datasets (0.52 on MIND and 0.14 on Goodreads).

Influence of Hyperparameters. We investigate the model sensitivity to the weight for summarization loss (λ) and the size of CPE and UPE. We randomly sampled 20% Train data of MIND dataset for training and validated performance on the Dev set. As Figure 3a shows, the (λ) values of 0.05, 0.1, and 0.3 yielded Dev performances of 70.76, 70.55, and 70.47, respectively. The CPE size values of 2, 4, and 8 resulted in Dev performances of 70.43, 70.76, and 70.56, respectively. The UPE size values of 16, 32, and 48 achieved Dev performances of 70.46, 70.76, and 70.58, respectively. These results indicate that our EmbSum model is not highly sensitive to hyperparameters. For CPE size, our findings suggest that increasing the number of context codes on the candidate item side does not improve model performance. We believe that excessively numerous codebooks for a single item might introduce superfluous parameters, thereby negatively impacting performance.

EmbSum-Generated Summary. We also evaluate the quality of the summaries generated by our model and report the ROUGE scores in Table 3. Since CTR is our main task, we generate summaries using the checkpoint that achieves the best AUC score on the Dev set. We select the test users who are not included in the training set and then generate their interest summaries based on their engagement history. We use the summaries generated by `Mixtral-8x22B-Instruct` as references for calculating ROUGE scores. For the MIND and Goodreads datasets, our model achieves ROUGE-L scores of 39.12 and 28.16, respectively. More concrete examples of the EmbSum-generated summaries can be found in the Appendix, demonstrating that EmbSum is capable of accurately capturing the user's diverse interests.

5 Conclusion

We present a novel framework EmbSum for content-based recommendation. EmbSum utilize encoder-decoder architecture and poly-attention modules to learn independent user and candidate content embeddings as well as generate user-interest summaries based on long user engagement histories. Through our experiments on two benchmark datasets, we have demonstrated that our framework achieves SoTA performance while using fewer parameters, and being able to generate a user-interest summary which can be used for recommendation explainability/transparency.

References

[1] Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Slav Petrov, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy P. Lillicrap, Angeliki

Lazaridou, Orhan Firat, James Molloy, Michael Isard, Paul Ronald Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, George Tucker, Enrique Piqueras, Maxim Krikun, Iain Barr, Nikolay Savinov, Ivo Danihelka, Becca Roelofs, Anaïs White, Anders Andreassen, Tamara von Glehn, Lakshman Yagati, Mehran Kazemi, Lucas Gonzalez, Misha Khalman, Jakub Sygnowski, and et al. 2023. Gemini: A Family of Highly Capable Multimodal Models. *ArXiv preprint* abs/2312.11805 (2023).

[2] Nicki Skafte Detlefsen, Jirí Borovec, Justus Schock, Ananya Harsh Jha, Teddy Koker, Luca Di Liello, Daniel Stancl, Changsheng Quan, Maxim Grechkin, and William Falcon. 2022. TorchMetrics - Measuring Reproducibility in PyTorch. *J. Open Source Softw.* 7, 69 (2022), 4101. https://doi.org/10.21105/JOSS.04101

[3] Tom Fawcett. 2006. An introduction to ROC analysis. *Pattern Recognit. Lett.* 27, 8 (2006), 861–874. https://doi.org/10.1016/J.PATREC.2005.10.010

[4] Youyang Gu, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Learning to refine text based recommendations. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, 2103–2108. https://doi.org/10.18653/v1/D16-1227

[5] Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2019. Poly-encoders: Transformer Architectures and Pre-training Strategies for Fast and Accurate Multi-sentence Scoring.

[6] Andreea Iana, Goran Glavaš, and Heiko Paulheim. 2023. NewsRecLib: A PyTorch-Lightning Library for Neural News Recommendation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Yansong Feng and Els Lefever (Eds.). Association for Computational Linguistics, Singapore, 296–310.

[7] Gautier Izacard and Edouard Grave. 2021. Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Association for Computational Linguistics, Online, 874–880. https://doi.org/10.18653/v1/2021.eacl-main.74

[8] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.* 20, 4 (2002), 422–446. https://doi.org/10.1145/582415.582418

[9] Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of Experts. *ArXiv preprint* abs/2401.04088 (2024).

[10] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, 7871–7880. https://doi.org/10.18653/v1/2020.acl-main.703

[11] 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*. Association for Computational Linguistics, Dublin, Ireland, 343–352. https://doi.org/10.18653/v1/2022.findings-acl.29

[12] Qijiong Liu. 2023. Legommenders: A Modular Framework for Recommender Systems.

[13] Qijiong Liu, Nuo Chen, Tetsuya Sakai, and Xiao-Ming Wu. 2023. ONCE: Boosting Content-based Recommendation with Both Open- and Closed-source Large Language Models.

[14] Rui Liu, Bin Yin, Ziyi Cao, Qianchen Xia, Yong Chen, and Dell Zhang. 2023. PerCoNet: News Recommendation with Explicit Persona and Contrastive Learning.

[15] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv preprint* abs/1907.11692 (2019).

[16] Itzik Malkiel, Oren Barkan, Avi Caciularu, Noam Razin, Ori Katz, and Noam Koenigstein. 2020. RecoBERT: A Catalog Language Model for Text-Based Recommendations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. Association for Computational Linguistics, Online, 1704–1714. https://doi.org/10.18653/v1/2020.findings-emnlp.154

[17] 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)*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, Toronto, Canada, 1160–1170. https://doi.org/10.18653/v1/2023.acl-short.100

[18] Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. 2017. Embedding-based News Recommendation for Millions of Users. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*. ACM, 1933–1942. https://doi.org/10.1145/3097983.3098108

[19] OpenAI. 2023. GPT-4 Technical Report. *ArXiv preprint* abs/2303.08774 (2023).

[20] Tao Qi, Fangzhao Wu, Chuhan Wu, and Yongfeng Huang. 2022. News Recommendation with Candidate-aware User Modeling. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai (Eds.). ACM, 1917–1921. https://doi.org/10.1145/3477495.3531778

[21] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.* 21 (2020), 140:1–140:67.

[22] Ellen M. Voorhees. 1999. The TREC-8 Question Answering Track Report. In *Proceedings of The Eighth Text REtrieval Conference, TREC 1999, Gaithersburg, Maryland, USA, November 17-19, 1999 (NIST Special Publication, Vol. 500-246)*, Ellen M. Voorhees and Donna K. Harman (Eds.). National Institute of Standards and Technology (NIST).

[23] Mengting Wan and Julian J. McAuley. 2018. Item recommendation on monotonic behavior chains. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018*, Sole Pera, Michael D. Ekstrand, Xavier Amatriain, and John O'Donovan (Eds.). ACM, 86–94. https://doi.org/10.1145/3240323.3240369

[24] Rongyao Wang, Shoujin Wang, Wenpeng Lu, and Xueping Peng. 2022. News Recommendation Via Multi-Interest News Sequence Modelling. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022, Virtual and Singapore, 23-27 May 2022*. IEEE, 7942–7946. https://doi.org/10.1109/ICASSP43922.2022.9747149

[25] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-Instruct: Aligning Language Models with Self-Generated Instructions. In *Proceedings of ACL 2023 (Volume 1: Long Papers)*. ACL, 13484–13508. https://doi.org/10.18653/V1/2023.ACL-LONG.754

[26] Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. Neural News Recommendation with Attentive Multi-View Learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, Sarit Kraus (Ed.). ijcai.org, 3863–3869. https://doi.org/10.24963/ijcai.2019/536

[27] Chuhan Wu, Fangzhao Wu, Suyu Ge, Tao Qi, Yongfeng Huang, and Xing Xie. 2019. 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)*. Association for Computational Linguistics, Hong Kong, China, 6389–6394. https://doi.org/10.18653/v1/D19-1671

[28] Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021. Empowering News Recommendation with Pre-trained Language Models.

[29] Chuhan Wu, Fangzhao Wu, Tao Qi, Yongfeng Huang, and Xing Xie. 2021. Fastformer: Additive Attention Can Be All You Need.

[30] Chuhan Wu, Fangzhao Wu, Yang Yu, Tao Qi, Yongfeng Huang, and Qi Liu. 2021. NewsBERT: Distilling Pre-trained Language Model for Intelligent News Application. In *Findings of the Association for Computational Linguistics: EMNLP 2021*. Association for Computational Linguistics, Punta Cana, Dominican Republic, 3285–3295. https://doi.org/10.18653/v1/2021.findings-emnlp.280

[31] 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*. Association for Computational Linguistics, Online, 3597–3606. https://doi.org/10.18653/v1/2020.acl-main.331

[32] Liancheng Xu, Xiaoxiang Wang, Lei Guo, Jinyu Zhang, Xiaoqi Wu, and Xinhua Wang. 2023. Candidate-Aware Dynamic Representation for News Recommendation. In *Artificial Neural Networks and Machine Learning - ICANN 2023 - 32nd International Conference on Artificial Neural Networks, Heraklion, Crete, Greece, September 26-29, 2023, Proceedings, Part VII (Lecture Notes in Computer Science, Vol. 14260)*, Lazaros Iliadis, Antonios Papaleonidas, Plamen P. Angelov, and Chrisina Jayne (Eds.). Springer, 272–284. https://doi.org/10.1007/978-3-031-44195-0_23

[33] 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*. International Joint Conferences on Artificial Intelligence Organization, Montreal, Canada, 3356–3362. https://doi.org/10.24963/ijcai.2021/462