KRED model: ablation study and new dataset application

Jiahao Zhang
DAUIN
Politecnico di Torino
Torino, Italy
s301185@studenti.polito.it

Tianming Qu
DAUIN
Politecnico di Torino
Torino, Italy
s302398@studenti.polito.it

Alessandra Borzomi
DAUIN
Politecnico di Torino
Torino, Italy
s302523@studenti.polito.it

***Abstract*—This paper investigates two extensions to the KRED model, focusing on block-wise analysis and domain adaptation. The first extension involves a comprehensive analysis of the model's performance by selectively disabling one of the three main model's architecture blocks, respectively: the entity representation layer, the context embedding layer, and the information distillation layer. Through this research, we aim to understand the relative contributions and interplay of these blocks in entity understanding. The second extension explores the model's applicability to different domains, specifically evaluating its performance on the Amazon book dataset. Our study sheds light on the versatility and effectiveness of the KRED model, providing valuable insights for knowledge graph-based entity representation and adaptation across different domains.**
The code is available at the following link :
KRED model: ablation study and new database application

I. INTRODUCTION

News recommendation systems have become essential for helping users navigate the overwhelming amount of digital news articles and discover relevant content. State-of-the-art models in news recommendation typically rely on manual feature extraction or extract latent representations using neural network models. However, these approaches often overlook the significance of entities within articles, which can provide valuable context and improve recommendation accuracy. Previous research also explored how to incorporate knowledge graphs into news recommendations and the most related work is the DKN [1]. While DKN demonstrated the potential of incorporating knowledge graphs, its focus on news titles limits its flexibility, making more challenging the integration of more advanced models such as Transformers. Additionally, recent advancements in natural language understanding, exemplified by BERT, have opened up new avenues for enhancing recommendation systems, leveraging the power of pre-training and fine-tuning. By combining the strengths of these state-of-the-art techniques and addressing their limitations it was proposed the Knowledge-aware Representation Enhancement model for news Documents (KRED). This framework leverages entity representations from a knowledge graph, refines them using the context of surrounding entities through the concept of Knowledge Graph Attention (KGAT), and incorporates various contextual information such as position, category, and frequency in a context embedding layer. Finally, an attention mechanism is employed to merge all entities into a fixed-length embedding vector, enhancing the original document representation. This enriched representation extends beyond personalized item recommendation to tasks like item-to-item recommendation, news popularity prediction, news category classification, and local news detection. These tasks exhibit shared data patterns, where users with similar interests tend to engage with news articles of similar topics, and news articles of the same category attract similar users. By jointly training these applications in a multi-task framework, we harness the complementary nature of their data and enhance the models' performance. The KRED model's multi-task learning approach facilitates the seamless integration of various news recommendation tasks. With a shared backbone model and task-specific predictors, it can leverage the power of collaborative signals from user-item interactions to enhance the learning of news category classification. By minimizing task-specific loss functions tailored to each task's objective, it is possible to effectively optimize the model's performance across personalized recommendations, item-to-item recommendations, news popularity prediction, news category classification, and local news detection.

II. RELATED WORKS

A. *News Recommendation Systems*

Different methods and techniques are proposed for personalized news recommendations. One such approach is the embedding-based method, which effectively utilizes distributed representations of articles and users. This method incorporates a variant of a denoising auto-encoder to refine article representations and employs a recurrent neural network (RNN) to capture user preferences and generate user representations [2]. Another notable technique focuses on learning informative representations by leveraging multiple perspectives of news, including titles, bodies, and topic categories. This approach utilizes an attentive multi-view learning model with word-level and view-level attention mechanisms, enabling the selection of crucial words and views for comprehensive news representations [3]. To enhance representation learning, a deep fusion model (DFM) has been proposed. By integrating an inception module and an attention mechanism, the DFM

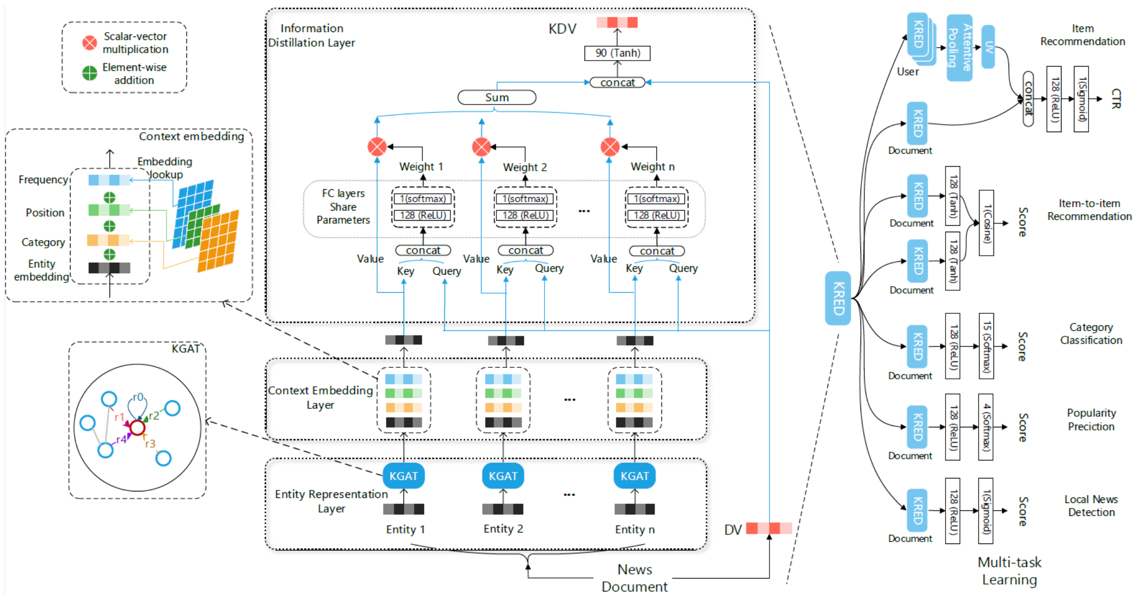

Fig. 1. The Framework of KRED.

improves candidate retrieval and item re-ranking in deep recommendation systems [4]. Lastly, a state-of-the-art technique, the neural news recommendation model with personalized attention (NPA), addresses the varying informativeness of words and news articles for different users. This approach leverages a convolutional neural network (CNN) for news representation and incorporates a personalized attention network, resulting in tailored recommendations that align with individual user interests and preferences [5].

B. *Knowledge-Aware Recommendation Systems*

A knowledge graph is a structured representation of real-world entities, their attributes, and the relationships between them. It captures rich and interconnected information from diverse sources, enabling a comprehensive understanding of the underlying domain. In the context of recommendation systems, knowledge graphs provide valuable insights into the relationships among users, items, and contextual factors. Knowledge-aware recommendation systems have emerged as a powerful approach to enhance the accuracy and explainability of recommendation algorithms. By incorporating knowledge graphs into the recommendation framework, researchers have significantly improved various domains, including news recommendation systems.

The work by Wu et al. [6] proposes a knowledge-aware recommendation framework that integrates a knowledge graph to enhance the accuracy of news recommendations. The authors demonstrate that by leveraging the rich relationships encoded in the knowledge graph, their model outperforms traditional recommendation methods. Similarly, Li et al. [7] propose a knowledge-aware news recommendation system that utilizes a knowledge graph to improve accuracy and explainability. Their results show that incorporating the knowledge graph significantly enhances recommendation quality while providing transparent and interpretable explanations.

III. ARCHITECTURE

The KRED (Knowledge-aware Representation Enhancement model) for news documents is illustrated in Figure 1. Its objective is to enhance an arbitrary Document Vector (DV) using knowledge graph information, resulting in a Knowledge-enhanced Document Vector (KDV) that can be utilized in downstream applications. The model consists of three key components: an entity representation layer, a context embedding layer, and an information distillation layer which will be briefly described in the following subsections.

A. *Entity Representation Layer*

Entities in news articles can be linked to corresponding entities in a knowledge graph, which is represented as a collection of entity-relation-entity triples denoted as: $G = \{(h, r, t) | h, t \in E, r \in R\}$, where E and R represent the set of entities and relations respectively, (h, r, t) represents that there is a relation r from the head entity h to the tail entity t. To enhance the representation of entity embeddings, we leverage information from the knowledge graph by considering the neighboring entities and the relations connecting them. Let N_h denote the set of triplets where h is the head entity. An entity is represented by:

$$e_{N_h} = \text{ReLU}\left(W_0\left(e_h \oplus \sum_{(h,r,t)\in N_h} \pi(h,r,t)\cdot e_t\right)\right)$$

where $\oplus$ denotes vector concatenation, e_h and e_t are the entity vectors learned of respectively the head entity and the tail entity and $\pi(h, r, t)$ is the attention weight that controls how

TABLE I
ABLATION STUDY

	User-to-item		Item-to-item	Category Classification	
	AUC	NDCG@10	AUC	ACCURACY	F1
Baseline Val	0.6390		0.8259	0.6937	
Baseline Test	**0.6404**	**0.3457**	**0.8867**	**0.6601**	**0.3229**
Time for one epoch	17 minutes		5 minutes	1 minutes	
w/o KGAT Val	0.6125		0.8221	0.6515	
w/o KGAT Test	0.5831	0.353	0.8806	0.6025	0.2606
Time for one epoch	**2.5 minutes**		**2 seconds**	**1 seconds**	
w/o Context Val	0.6074		0.8221	0.6515	
w/o Context Test	0.5875	0.355	0.8786	0.6582	0.3244
Time for one epoch	23 minutes		5 minutes	1 minutes	
w/o Distillation Val	0.6284		0.8351	0.6304	
w/o Distillation Test	0.6285	0.3377	0.8888	0.5752	0.19
Time for one epoch	23 minutes		5 minutes	1 minutes	

much information the neighbor node needs to propagate to the current entity.

B. Context Embedding Layer

Entities in documents may have different positions, and frequencies or can be associated with different categories. The dynamic context significantly impacts the significance and pertinence of an entity within an article. To account for this, three types of context embedding features are incorporated into the entity embeddings created by the Entity Representation Layer respectively a position-biased vector, a frequency encoding vector, and a category encoding vector:

$$e_{I_h} = e_{N_h} + C^{(1)}_{p_h} + C^{(2)}_{f_h} + C^{(3)}_{t_h}$$

C. Information Distillation Layer

The final step includes using an attentive mechanism to merge all entities' information into one output vector. The original document vector v_d serves as the query. Both key and value are entity representation e_{I_h}.

$$e_{O_h} = \prod_{h \in E_v} \pi(h, v) \cdot e_{I_h} \tag{1}$$

The final entity vector and the original document vector v_d are then concatenated and passed to one fully-connected feed-forward network to obtain v_k, the Knowledge-Aware document vector.

$$v_k = \tanh(W_3(e_{Oh} \oplus v_d) + b_3$$

IV. EXPERIMENTS AND RESULTS

In this section, we will briefly discuss some possible extensions to better explain how the KRED model works.

A. Datasets

1) MIND Dataset: The MIND dataset for news recommendation was collected from anonymized behavior logs of the Microsoft News website. It is formed by data containing information about 1 million users who had at least 5 news clicks during 6 weeks from October 12 to November 22, 2019. All the clicks are formatted into impression logs and a small version of the dataset, the MIND-small, the one used in this ablation study, was created by randomly sampling 50,000 users and their behavior logs. Both training and validation dataset folders contain four different files, respectively: behaviors.tsv containing the impression logs and users' news click histories, news.tsv containing detailed information on news articles involved in the behaviors.tsv file and both entity_embedding.vec and relation_embedding.vec contain 100-dimensional embeddings of the entities and relations learned from the WikiData knowledge graph.

2) Amazon Book Dataset: The Amazon Book Dataset is made of two distinct files: the "Reviews" file and the "Books Details" file. The "Reviews" file contains a vast collection of product reviews and metadata sourced from Amazon, consisting of 142.8 million reviews gathered between May 1996 and July 2014 whereas the "Books Details" file provides comprehensive information about 212,404 unique books and is built by using the Google Books API. The content of the first file comprehends various features associated with the reviews, including the book title, price, user ID of the reviewer, and additional details such as the time of the review, a summary of its content, the full text of the review, and a score ranging from 0 to 5 representing the rating given to the book. The second file delves into extensive attributes covering diverse aspects of books, such as the book title, description, authors, ratings (average ratings provided by different users), publisher, and genre (category) of the books.

B. Baseline and Ablation study

Before the experiment, we present the details of our experimental setup:

- **Hyper-parameters**: for fine-tuning the model, we select *Adam* as optimizer with *learning rate* 2×10^{-5}, and *weight decay* 1×10^{-6}. Because of the limitation of computing resources, we set the *batch size* equal to 512.

- **Baselines**: To test the performance of the model, we consider the following tasks:
  User-to-item Recommendation: The primary aim of this task is to provide personalized news recommendations to users, leveraging their past click history. The performance of our model is assessed using the *Area Under the Curve (AUC)* and *Normalized Discounted Cumulative Gain (NDCG@10)* metrics.
  Item-to-item Recommendation: This task aims to accurately predict a collection of positive news pairs. A positive pair is defined as two news articles that have been clicked on by over 100 common users. The performance of our model under this task is evaluated using the *Area Under the Curve (AUC)* metric.
  Article Category Classification: Every news article in our project is assigned a specific category, such as health, sports, weather, and more. For classification purposes, we have classified the news into 15 distinct categories. Since this constitutes a multi-class classification problem, we utilize *Accuracy* and *F1 score* as the evaluation metrics.

- **Ablation study**: As stated in Section III, the KRED model comprises three components: the *Entity Representation Layer*, *Context Embedding Layer*, and *Information Distillation Layer*. These three layers are integrated into the news embedding layer. To gain a deeper understanding of their construction and performance, we remove one layer at a time and fine-tune the model across multiple tasks, as an extensive study, we also record the training time.

The results are presented in Table I. Notably, removing the entity representation layer has the most significant impact on performance, resulting in a drop of approximately **9%**. This layer incorporates information about the knowledge graph and the intricate relationships between entities. However, in the item-to-item recommendation task, which does not rely on user-item relations, the performance is relatively less affected compared to other tasks. Furthermore, the omission of position, frequency, and category information leads to an approximate **8%** decline in performance.

Importantly, the entity representation layer involves training another neural network, namely KGAT. Consequently, removing this layer can substantially reduce the computational resource requirements. As shown in the table, the training time is reduced by 85%.

Regarding the distillation layer, although its impact on performance is relatively smaller compared to other layers, we observed during training that it significantly reduces the training cost. This reduction is attributed to the attention mechanism within the distillation layer, which effectively reduces the size of the training data required.

Collectively, these three layers work together to enhance the model's performance. Removing any of these layers would result in performance degradation or increased computational resource requirements. Therefore, each layer has its unique and essential contribution to the model's effectiveness.

C. New dataset application

Recommendation systems' application does not only limit to news recommendations. Another possible application is related to e-commerce. In this section, we will briefly describe the main key steps we have done to manually adapt the Amazon book dataset [8] for the KRED usage, in particular, how to retrieve the knowledge graph based on Wikidata (this approach can be extended to most of the datasets) and how the model performs when we give input data from different domains. The main steps could be summarized as follows:

1 Creation of the entities
2 Extraction of the relationship between entities
3 Extraction of user's behavior

1) Creation of the entities: We can create the entities starting from the book's details. In particular, we will consider the following fields from the dataset: *title*, *description*, *author*, and *categories*. For the first two, we apply POS tagging [9] and then a Named Entity Recognition [10] module to retrieve the most important words. For the latter two fields, we simply extract the words. After obtaining the relevant words, we use the Wikidata API to obtain the wikidata_ids and the wikidata descriptions. Passing the description to a Sentence Transformer (a pre-trained model available in the Hugging Face API [11]), we obtain a sentence embedding of the entity description. Since the KRED model requires the embedding to have a vector dimension of 100, we perform PCA to extract the most relevant features.

2) Extraction of the relationship between entities: With the wikidata_ids, we can extract also the relationship between the entities. We process the relationship description in the same way, by extracting its sentence embeddings and performing PCA [12] on it. Meanwhile, we also record the triplets (entity_id, relationship_id, entity_id) which is crucial to create the knowledge graph.

3) Extraction of user's behavior: User information can be extracted from the review dataset. In this case, we also extract the most important words by applying POS tagging + NER and pass the results to wikidata API to retrieve the wikidata_id. Based on the retrieved entities, we try to simulate the behavior by retrieving the correlated books both in terms of interacted books (by leaving a review) and also by random clicks. After obtaining the behaviors, we simply split the behaviors to obtain the training and validation dataset.

Following this approach, we obtain a database whose structure is very similar to the MIND dataset, and this allows us to directly use it to train a KRED model specific to the book recommendation system based on the Amazon Book dataset.

Using the same setup we have used in the baseline for the MIND database, we obtain the following result:

TABLE II
AMAZON BOOK EXPERIMENTS

Datasets	User-to-item AUC	NDCG@10
MIND Baseline	0.6404	0.3457
Amazon Books	**0.6346**	**0.3012**

We notice that we have a model that is comparable to the Baseline, which means that our approach works fine and this pipeline could be reused for different datasets that may require a model for a Recommendation system.

V. Conclusion

In conclusion, this paper shows two possible extensions of the KRED model. Firstly, a deeper ablation study has been performed on the architecture pointing out the advantages/disadvantages of each module and their relative cost and time efficiency related to the task we would like to tackle. Secondly, a systematic approach to creating a database has been introduced. This allows any user to create an input database that can be fed directly to the KRED architecture to create a model for the specific domain.

References

[1] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep Knowledge-Aware Network for News Recommendation. In Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018, Pierre-Antoine Champin, Fabien L. Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis (Eds.). ACM, 1835–1844. https://doi.org/10.1145/ 3178876.3186175

[2] 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

[3] 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

[4] Jianxun Lian, Fuzheng Zhang, Xing Xie, and Guangzhong Sun. 2018. Towards Better Representation Learning for Personalized News Recommendation: a Multi-Channel Deep Fusion Approach. In Proceedings of the TwentySeventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden, Jérôme Lang (Ed.). ijcai.org, 3805–3811. https: //doi.org/10.24963/ijcai.2018/529

[5] Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. NPA: Neural News Recommendation with Personalized Attention. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019, Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (Eds.). ACM, 2576–2584. https://doi.org/10.1145/3292500.3330665

[6] Wu, C., Wu, F., An, M., Huang, J., Huang, Y., & Xie, X. (2019). Knowledge-aware News Recommendation with Hierarchical Graph Convolutional Networks. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (pp. 1737–1746). https://doi.org/10.1145/3357384.3358056

[7] Li, J., Li, X., Wang, Y., & Gao, J. (2020). Explainable News Recommendation via Knowledge Graph-Aware Convolutional Sequence Model. ACM Transactions on Information Systems, 38(1), 1–29. https://doi.org/10.1145/3371739

[8] Amazon book review dataset https://www.kaggle.com/datasets/mohamedbakhet/amazon-books-reviews?select=Books_rating.csv

[9] POS tagging documentation https://www.nltk.org/api/nltk.tag.pos_tag.html

[10] Named Entity Recognition documentation https://www.nltk.org/api/nltk.chunk.ne_chunk.html

[11] Hugging Face API for Sentence Transformers https://huggingface.co/sentence-transformers

[12] Nathan Halko, Per-Gunnar Martinsson, Joel A. Tropp, Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions, https://arxiv.org/abs/0909.4061