KRED: News Recommendation System Investigation and Implementations

1st Matteo Garbarino
Politecnico di Torino
Data Science and Engineering
Turin, Italy
s265386@studenti.polito.it

2nd Micol Rosini
Politecnico di Torino
Data Science and Engineering
Turin, Italy
micol.rosini@studenti.polito.it

3rd Gaia Sabbatini
Politecnico di Torino
Data Science and Engineering
Turin, Italy
s291532@studenti.polito.it

***Abstract*—This paper presents an enhanced online news recommendation system that incorporates KRED[2], a knowledge graph-based recommendation model. KRED utilizes a knowledge graph system, taking into account information from the entity's neighbourhood and refining the entity embedding with dynamic insights based on the news context. In addition, we propose two extensions to the baseline model: data enrichment and movie recommendations. The first extension enables the system to exploit news reading time to model users' behaviour. The second extension expands the system's functionality to include movie recommendations, offering a more comprehensive and personalized user experience beyond news content. By integrating the KRED model with the aforementioned techniques, the enhanced recommendation system delivers accurate and context-aware recommendations. It leverages the knowledge graph's interdependencies and dynamic insights to refine entity embeddings, resulting in improved performance and user satisfaction. The code is available at:**
https://github.com/micolrosini/KRED-Recommendation-System

I. Problem Statement

In today's digital age, the consumption of online news has become an integral part of people's daily lives. With the abundance of news sources and articles available, personalized recommendation systems play a huge role in helping users navigate through this vast sea of information. This paper focuses on the development and extension of a recommendation system for online news, utilizing the Knowledge-Aware Representation Enhancement model for news Documents (KRED).

KRED is a powerful framework designed to enhance the representation of news articles by incorporating knowledge entities mentioned within the articles. This knowledge-aware approach provides valuable insights that can improve the accuracy and relevance of recommendations. The versatility of KRED is showcased by its ability to perform various recommendation tasks, including item-to-item and user-to-item recommendations, news popularity prediction, news category prediction, and local news detection.

Due to limited computational resources, this paper primarily focuses on the user-to-item recommendation task within the KRED framework. By analyzing users' reading history and comparing it with the characteristics of news articles, the goal is to recommend news articles that are most likely to be of interest to individual users.

In addition to the baseline KRED model, this paper proposes two extensions that further enhance its capabilities.

- **Data enrichment**: by extending the model to accommodate a dataset containing information about the reading time spent by each user on each news article clicked. The recommendation system can therefore learn from a more informative dataset leading to better user preference modelling.
- **Domain adaptation for Movies recommendations**: by leveraging Mind Reader dataset, that contains film-related informations and entities, the extended KRED model can recommend movies that are highly likely to be enjoyed by users based on their preferences and interests.

The contributions of this paper lie in the adaptation and extension of the KRED model for user-to-item recommendations in the online news domain. By incorporating data enrichment and movie recommendations, the enhanced model strives to provide a more tailored and personalized user experience. The following sections will delve into the methodology, experimental setup, and evaluation results to demonstrate the effectiveness and benefits of these extensions in the recommendation system for online news using the KRED model.

II. Methodology

In this paper, we propose a Knowledge-aware Representation Enhancement model for news documents (KRED) that utilizes the red backbone to create a recommendation system for news and other extensions. The methodology of our approach consists of three key layers: an entity representation layer, a context embedding layer, and an information distillation layer.

1) **Entity Representation Layer**: the assumption used is that entities in news articles can be linked to corresponding entities in a knowledge graph. TransE is used to learn embedding vectors for each entity and relationship in the knowledge graph. To represent an entity, we employ the Knowledge Graph Attention (KGAT) Network, which considers both the entity's own embeddings and its neighbours' representations.
2) **Context Embedding Layer**: to reduce computational costs, the conclusive information of entities is extracted from the original document. There are three context

embedding features: position encoding, frequency encoding, and category encoding. The position encoding distinguishes between entities appearing in the title and body of the news, while the frequency encoding captures the importance of entities based on their occurrence frequency. The category encoding helps the model understand content more accurately by revealing the category of entities.

3) **Information Distillation Layer**: the importance of an entity is not solely determined by its own message but also influenced by other co-occurring entities and the topics of the article. An attentive mechanism merges all entities' information into one output vector. The attention weight is computed using a two-layer fully connected neural network, and the key and value are entity representations obtained from TransE. The attention mechanism allows for information propagation from neighbour nodes to the current entity.

Finally, the entity vector and the original document vector are concatenated and fed into a fully-connected feed-forward network to obtain the Knowledge-aware Document Vector (KDV). By employing the KRED model, we can generate Knowledge-enhanced Document Vectors (KDV) that can be consumed by various downstream applications, including recommendation systems for news. Our framework has the advantage of accommodating different document representation methods, utilizing all available data in news documents, and achieving computational efficiency. The KRED model is highly effective for news recommendations due to its ability to cover a broad range of applications. It goes beyond personalized recommendations and includes item-to-item recommendation, news popularity prediction, new category prediction, and local news detection. This wide coverage makes KRED a versatile and comprehensive solution for various aspects of news recommendation systems. Overall, the proposed methodology of the KRED model enables the development of other extensions related to recommendation systems. Fig.2 shows the proposed KRED model.

A. First extension: Domain adaptation with Movies recommendations

The Mind-reader dataset used in the training process contains movie entities and related entities such as actors, companies, film categories, release years, and directors. Additionally, there is a rating file where users have provided positive or negative ratings for movies or entities related to movies. The dataset also includes a knowledge graph that connects entities with specific relations.

To recreate the files necessary for training the recommender system, the following steps were performed:

1) **Manual analysis and mapping of Wikidata IDs**: Since not all entities in the dataset had associated Wikidata IDs, a manual analysis was conducted on the Wikidata database to find the corresponding IDs for those entities.

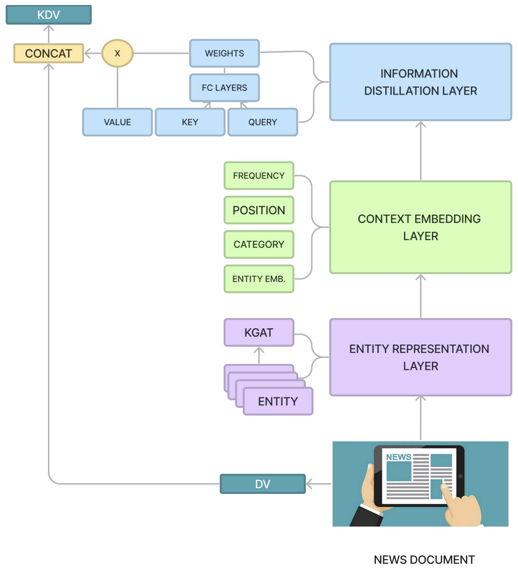


Fig. 1: An overview of the proposed KRED model. FC-LAYERS indicates fully connected layers. DV indicates the (original) document vector. KDV indicates the knowledge-enhanced document vector produced by KRED.

This mapping process ensured that each entity had a unique identifier.

2) **Querying Wikidata for entity embeddings**: the textual description of the entity ID is retrieved from the Wikidata database. Then it is passed to the BERT model to obtain the entity embedding.

3) **Creating movie embeddings**: to create movie embeddings and capture all relevant information, we gathered entities associated with each movie from the knowledge graph. This allowed us to enrich the movie embeddings with details about actors, directors, year of publication, and genre. Additionally, we encoded additional information such as position, frequency, and category for each entity, following the approach used in the original training. The entity category was determined by analyzing the relations present in the knowledge graph. For example, if two distinct entities were connected to other entities using the same relation, such as *Starred by*, it indicated that these second linked entities shared the category of *actor*. By leveraging these relationships, we extracted valuable insights about entity categories. Since we didn't have an abstract or summary available for the movies, we treated all entities as part of the "title."

4) **User history creation**: The user history was generated by selecting movies that were positively rated by each user. These movies were considered part of the user's history since they served as a reflection of the user's preferences and viewing experiences. The user history would later be used to personalize recommendations.

5) **Positive and negative lists**: Using the rating file, positive and negative lists were created for each user. These

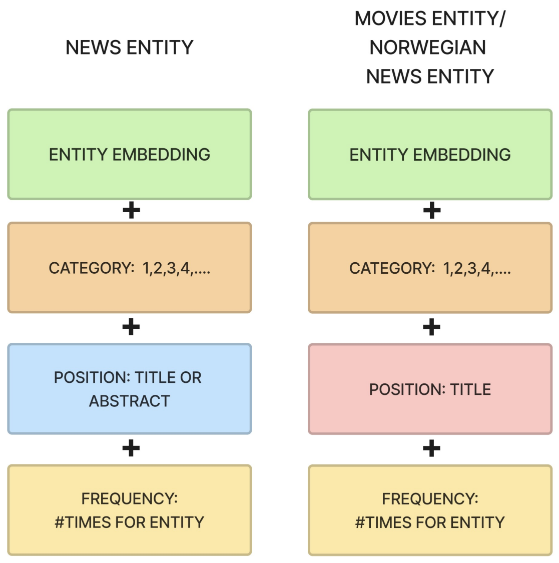
lists contained movies and entities that were liked or disliked by the user. By including entities like actors or directors in the lists, the model would be able to recommend movies featuring those entities to users. (By incorporating entities such as actors or directors in the recommendation lists, our model goes beyond suggesting movies alone. It has the ability to provide recommendations based on user preferences for specific directors or actors. This means that users can not only discover movies they might enjoy but also explore films directed by their preferred directors or featuring their favorite actors. Considering these additional dimensions, our system aims to enhance the user's movie-watching experience by offering a more comprehensive and tailored set of recommendations.)

Following these steps, all the necessary files were recreated to train the recommender system. The movie and entity embeddings, user history, and positive/negative lists allowed the model to understand user preferences better and make personalized recommendations based on their interests.

B. Second extension: Data enrichment

The reason for developing this extension is to model more accurately the user preferences. In the news context, the assumption that a click on an article always expresses a real interest may be an oversimplification. Users may quickly close an article due to a "clickbait" title or lose interest while reading. A richer dataset containing reading time information is assumed to be helpful to better understand the true user preferences. Unfortunately, it wasn't possible to generate realistic synthetic data to enrich the original *MIND dataset*[5, 3], therefore a new one naturally containing the needed information was selected, the *Adressa Norwegian News Dataset*[1]. The goal is to train the KRED model to perform the same task using a richer dataset exploiting user's article reading time to better model their behaviors.
To recreate the files necessary for training the recommender system, the following steps were performed:

1) **Data loading**: The *Adressa dataset* was originally a chronologically ordered log of user clicks on news articles (structure described in section III.A.3). Given the vast dimension (16GB) the dataset and its irregular schema, a considerable effort was done in this step to load the data is a suitable data structure. Albeit using an ad-hoc environment (e.g. Google Colab), the disk space and RAM were limiting factors, thus it was necessary to accurately choose data structures and perform memory management operations to obtain a Pandas DataFrame[4] to be later processed.
2) **Dataset preprocessing - user modelling**: The next step was to obtain a suitable input for the main model. For the generation of the *behaviours.tsv* file the key point was to extract the user history and the user behaviours (i.e. positive and negative clicks). While the user history is a subset of the positive clicks, conversely the definitions of positive and negative clicks had to be renewed to consider the reading time. For each user, the mean and standard deviation of their reading time of all their clicked news was computed. Then, if the reading time was within 1 standard deviation from the mean user reading time for each user click, it was classified as a positive click, otherwise as negative. The interpretation is that if a user closes too quickly or takes too long w.r.t. their habits (e.g. loses interest in the article and leaves the page open) it is safe to assume there was no interest in that article.
3) **Dataset preprocessing - news modelling**: The entities embeddings, knowledge graph and adjacency matrix files were not provided by the dataset and had to be generated. Due to the unavailability of the news corpora, the extraction of entities was performed using the metadata present in the dataset, including news title, category, URL, keywords, and the "profile" field, which contained a list of potential entities. These extracted entities were then mapped to corresponding Wikidata IDs, as outlined in steps II.A.1 and II.A.2. To construct the Knowledge Graph from the ground up, requests were made to the Wikidata API, and an iterative process was carried out using the previously obtained list of Wikidata ID entities. This process aimed to explore possible relationships among the entities and establish connections within the Knowledge Graph. Finally, the Adjacency Matrix was generated based on the established relationships.


NEWS ENTITY
MOVIES ENTITY/ NORWEGIAN NEWS ENTITY
ENTITY EMBEDDING
+
CATEGORY: 1,2,3,4,....
+
POSITION: TITLE OR ABSTRACT
+
FREQUENCY: #TIMES FOR ENTITY
ENTITY EMBEDDING
+
CATEGORY: 1,2,3,4,....
+
POSITION: TITLE
+
FREQUENCY: #TIMES FOR ENTITY
Fig. 2: Difference of the original news entity embedding and the movies/addressa entity embedding: the only position considered is title, there is no abstract's entities.

III. Experiments

A. Dataset Description

1) Microsoft News Dataset (MIND): The KRED framework utilizes the Microsoft News Dataset (MIND), a large-scale collection of anonymized behavior logs from the Microsoft News website. MIND consists of 1 million users and over 160,000 English news articles. Additionally, a smaller version called MINDsmall is available, which includes a random sample of 50,000 users and their corresponding behavior logs.

Each news article in the dataset is associated with several features, including:

- News ID
- News Category and Sub-Category
- Title and a concise abstract
- URL of the news article
- List of entities mentioned in the title, and in the abstract along with the corresponding wikidataID
- List of entities mentioned in the abstract

Each user's log is associated with the following features:

- User-Id
- Time stamp of the log
- User's news reading history
- News-Id+label: 1 if the news shown to the user was clicked on, and 0 otherwise.

In the provided dataset, along with the news articles, the embeddings of entities and relations are included. Additionally, a knowledge graph is provided that establishes connections between entities through relations.

2) Mind Reader Dataset: MindReader is a novel dataset providing explicit user ratings over a knowledge graph within the movie domain, it contains 218,794 ratings from 2,316 users over 12,206 entities, and an associated knowledge graph consisting of 18,133 movie-related entities. The dataset is collected from an online movie recommendation game, MindReader, where users are pseudo-randomly asked to provide preferences for both movie- and non-movie entities (e.g., genres, actors, and directors). For each entity, users can either like it, dislike it, or state that they do not know it. It contains the file

- `ratings.csv` containing for each item the UserId, the WikidataId of the entity, a boolean indicator that states if the item is a film or not, the value of the rating(+1/-1/0)
- `entities.csv`: containing for each entity the wikidataId, the name of the entity, and the category.
- `knowledgegraphtriplets.csv`: the knowledge graph.

In the provided dataset, entity and relations embeddings were not explicitly given.

3) Adressa Norwegian News Dataset:: Adressa is an anonymized user behavior log collected from Adresseavisen, a local newspaper company in Trondheim (Norway), for Recommendation Technology research purposes. The dataset version used for the experiments is AdressaSMALL, collected over a full week (Jan 1st to Jan 7th 2017) and consists of over 280.000 users, 40.000 news articles and 10.000.000 clicks. The most relevant Adressa's features are:

- userId: unique user identifier
- timestamp: timestamp of the click
- URL: clicked news URL, also used as news ID
- News title, category, keywords, profile: news metadata used to extract entities
- activeTime: reading time of the user for the article

The dataset's missing values have been dropped, except for the reading times where the missing value meant less than 1 second. The embeddings, Knowledge Graph and Adj. Matrix were not included and had to be recomputed from scratch.

B. Experimental Design

Considering the limitations in memory space and processing time, both on our local machines and Google's Colab platform with the Tesla T4 GPU (16GB memory), it became apparent that processing the entire MIND dataset was not feasible. Consequently, we decided to work exclusively on the MINDsmall dataset (50000 users). During the preprocessing stage, which encompassed the majority of data preparation for both the original task and its extension, we primarily relied on our local machines. However, when it came to the training phase, we leveraged the powerful computational resources offered by the Google Colab platform.

To assess the performance of the model, various metrics were employed based on the specific tasks at hand. However, due to computational power limitations, we chose to concentrate solely on the user-to-item task. Consequently, the evaluation metric utilized in this work is the **AUC (Area Under the Curve)** score, commonly employed in binary classification tasks. It measures the performance of a classifier by calculating the area under the Receiver Operating Characteristic (ROC) curve. The ROC curve plots the true positive rate (sensitivity) against the false positive rate (1-specificity) at various classification thresholds. The AUC score represents the overall performance of the classifier across all possible threshold values, it quantifies how well a classifier can discriminate between positive and negative instances. This metric ranges from 0 to 1, where a score of 1 indicates a perfect classifier. TAB.I displayed our best configuration of hyper-parameters.

Hyper-parameters	**Values**
Epoch	10
Batch size	64
Learning rate	$2e^{-5}$
Learning rate decay	$1e^{-6}$
Optimizer	Adam
Number of workers	4

TABLE I: Hyper-parameters set

C. Execution Times

Due to the substantial time required to complete each epoch, which averaged around 20 minutes, we decided to limit the training process to 10 epochs. This choice was primarily motivated by the constraints imposed by limited computational resources. By reducing the number of epochs, we aimed to

Configuration	**AUC score**
KRED for NEWS recommendation	0.6201
KRED for MOVIES recommendation	0.7143
KRED for ENRICHED NEWS recommendation	0.619

TABLE II: AUC score for the different models

strike a balance between optimizing the model's performance and efficiently managing the available resources.

D. Results

The performance evaluation of the experiments is displayed in Table II, it showcases the three configurations proposed and the AUC score results for the most successful models. The selected hyper-parameters set is shown in TableI. Sharing the same set of hyper-parameters among the experiments allows for more comparable results. The metrics are lower than the ones reported in the original paper because we use a smaller dataset and fewer epochs. Overall the most successful experiment appears to be the Domain Adaptation Extension, while instead, surprisingly, the data enrichment hasn't shown a significant improvement in the model performances, but neither a significant worsening, meaning that it could probably be further optimized.

IV. Conclusions and Takeaways

Starting from KRED as a baseline model, which benefits from knowledge graphs to enrich its embeddings we developed two extensions with different purposes. The performances of the baseline model are slightly lower reasonably due to the usage of the MIND Small version of the dataset, but it was not possible to confirm this assumption due to a lack of computational resources to utilize the full MIND dataset. The first extension, domain adaptation to movies recommendation, scored a higher performance w.r.t. the baseline model, thus proving its versatility and the effectiveness of its innovations. The second extension, data enrichment through the usage of reading time, is aimed at further boosting the performances of the baseline model on the same task. However, the extension scored a lower performance, which could be explained by a number of involved factors. The introduction of a new dataset was necessary to introduce the reading time information, but might not allow a fair comparison with the baseline, also because the entities had to be extracted from the metadata and the knowledge graph had to be rebuilt, which has been one of the main challenges together with computational resources management. Furthermore, the usage of the reading time by itself, even if it is a correct procedure since it was compared with users' own mean reading time, would have been a much more informative datapoint if combined also with the document length, which was not an available information. Surely this last consideration could be used as inspiration and starting point for future further research.

References

[1] Jon Atle Gulla et al. "The Adressa Dataset for News Recommendation". In: *Proceedings of the International Conference on Web Intelligence*. ACM. Aug. 2017, pp. 1042–1048.

[2] Danyang Liu et al. "Fast and Accurate Knowledge-Aware Document Representation Enhancement for News Recommendations". In: *CoRR* abs/1910.11494 (2019). arXiv: 1910.11494. URL: http://arxiv.org/abs/1910.11494.

[3] Microsoft News Dataset. *MSNews Website*. https://msnews.github.io/. 2023.

[4] The pandas development team. *pandas-dev/pandas: Pandas*. Version v2.0.3. If you use this software, please cite it as below. June 2023. DOI: 10.5281/zenodo.8092754. URL: https://doi.org/10.5281/zenodo.8092754.

[5] Fangzhao Wu et al. "MIND: A Large-scale Dataset for News Recommendation". In: *Proceedings of the Association for Computational Linguistics (ACL)*. 2020.