DKN: Deep Knowledge-Aware Network for News Recommendation

Hongwei Wang[1,2], Fuzheng Zhang[2], Xing Xie[2], Minyi Guo[1]
[1]Shanghai Jiao Tong University, Shanghai, China
[2]Microsoft Research Asia, Beijing, China
wanghongwei55@gmail.com,{fuzzhang,xingx}@microsoft.com,guo-my@cs.sjtu.edu.cn

ABSTRACT

Online news recommender systems aim to address the information explosion of news and make personalized recommendation for users. In general, news language is highly condensed, full of knowledge entities and common sense. However, existing methods are unaware of such external knowledge and cannot fully discover latent knowledge-level connections among news. The recommended results for a user are consequently limited to simple patterns and cannot be extended reasonably. Moreover, news recommendation also faces the challenges of high time-sensitivity of news and dynamic diversity of users' interests. To solve the above problems, in this paper, we propose a *deep knowledge-aware network* (DKN) that incorporates knowledge graph representation into news recommendation. DKN is a content-based deep recommendation framework for click-through rate prediction. The key component of DKN is a multi-channel and word-entity-aligned *knowledge-aware convolutional neural network* (KCNN) that fuses semantic-level and knowledge-level representations of news. KCNN treats words and entities as multiple channels, and explicitly keeps their alignment relationship during convolution. In addition, to address users' diverse interests, we also design an *attention* module in DKN to dynamically aggregate a user's history with respect to current candidate news. Through extensive experiments on a real online news platform, we demonstrate that DKN achieves substantial gains over state-of-the-art deep recommendation models. We also validate the efficacy of the usage of knowledge in DKN.

KEYWORDS

News recommendation; knowledge graph representation; deep neural networks; attention model

1 INTRODUCTION

With the advance of the World Wide Web, people's news reading habits have gradually shifted from traditional media such as newspapers and TV to the Internet. Online news websites, such as Google News[1] and Bing News[2], collect news from various sources and provide an aggregate view of news for readers. A notorious problem with online news platforms is that the volume of articles can be overwhelming to users. To alleviate the impact of information overloading, it is critical to help users target their reading interests and make personalized recommendations [2, 27, 29, 34, 36, 41].

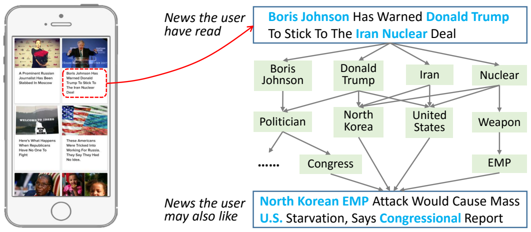

News the user have read
Boris Johnson Has Warned Donald Trump To Stick To The Iran Nuclear Deal
Boris Johnson
Donald Trump
Iran
Nuclear
Politician
North Korea
United States
Weapon
......
Congress
EMP
News the user may also like
North Korean EMP Attack Would Cause Mass U.S. Starvation, Says Congressional Report
A Prominent Russian Journalist Has Been Stabbed In Moscow
Boris Johnson Has Warned Donald Trump To Stick To The Iran Nuclear Deal
Here's What Happens When Republicans Have No One To Fight
These Americans Were Tricked Into Working For Russia. They Say They Had No Idea.

Figure 1: Illustration of two pieces of news connected through knowledge entities.

Generally, news recommendation is quite difficult as it poses three major challenges. First, unlike other items such as movies [10] and restaurants [14], news articles are highly time-sensitive and their relevance expires quickly within a short period (see Section 5.1). Out-of-date news are substituted by newer ones frequently, which makes traditional ID-based methods such as collaborative filtering (CF) [43] less effective. Second, people are topic-sensitive in news reading as they are usually interested in multiple specific news categories (see Section 5.5). How to dynamically measure a user's interest based on his diversified reading history for current candidate news is key to news recommender systems. Third, news language is usually highly condensed and comprised of a large amount of knowledge entities and common sense. For example, as shown in Figure 1, a user clicks a piece of news with the title "Boris Johnson Has Warned Donald Trump To Stick To The Iran Nuclear Deal" that contains four knowledge entities: "Boris Johnson", "Donald Trump", "Iran" and "Nuclear". In fact, the user may also be interested in another piece of news with the title "North Korean EMP Attack Would Cause Mass U.S. Starvation, Says Congressional Report" with high probability, which shares a great deal of contextual knowledge and is strongly connected with the previous one in terms of commonsense reasoning. However, traditional semantic models [32] or topic models [3] can only find their relatedness based on co-occurrence or clustering structure of words, but are hardly able to discover their latent knowledge-level connection. As a result, a user's reading pattern will be narrowed down to a limited circle and cannot be reasonably extended based on existing recommendation methods.

To extract deep logical connections among news, it is necessary to introduce additional *knowledge graph* information into news recommendations. A knowledge graph is a type of directed heterogeneous graph in which nodes correspond to *entities* and edges correspond to *relations*. Recently, researchers have proposed several academic knowledge graphs such as NELL[3] and DBpedia[4], as

[1] https://news.google.com/
[2] https://www.bing.com/news
[3] http://rtw.ml.cmu.edu/rtw/
[4] http://wiki.dbpedia.org/

WWW'18, April 23–27, 2018, Lyon, France.
2018. 978-x-xxxx-xxxx-x/YY/MM…$15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

well as commercial ones such as Google Knowledge Graph[5] and Microsoft Satori[6]. These knowledge graphs are successfully applied in many scenarios such as machine reading[51], text classification[46], and word embedding[49], to name just a few.

Considering the above challenges in news recommendation and inspired by the wide success of leveraging knowledge graphs, in this paper we propose a novel framework that takes advantage of external knowledge for news recommendation, namely the *deep knowledge-aware network* (DKN). Different from CF-based methods, DKN is a content-based model for click-through rate (CTR) prediction, which takes one piece of candidate news and one user's click history as input, and outputs the probability of the user clicking the news. Specifically, for a piece of input news, we first enrich its information by associating each word in the news content with a relevant entity in the knowledge graph. We also search and use the set of contextual entities of each entity (i.e., its immediate neighbors in the knowledge graph) to provide more complementary and distinguishable information. Then we design a key component in DKN, namely *knowledge-aware convolutional neural networks* (KCNN), to fuse the word-level and knowledge-level representations of news and generate a knowledge-aware embedding vector. Distinct from existing work [46], KCNN is: 1) *multi-channel*, as it treats word embedding, entity embedding, and contextual entity embedding of news as multiple stacked channels just like colored images; 2) *word-entity-aligned*, as it aligns a word and its associated entity in multiple channels and applies a transformation function to eliminate the heterogeneity of the word embedding and entity embedding spaces. The intuitive understanding of the superiority of KCNN is that it maintains the alignment of multiple representations for a word and explicitly bridges different embedding spaces.

Using KCNN, we obtain a knowledge-aware representation vector for each piece of news. To get a dynamic representation of a user with respect to current candidate news, we use an *attention* module to automatically match candidate news to each piece of clicked news, and aggregate the user's history with different weights. The user's embedding and the candidate news' embedding are finally processed by a deep neural network (DNN) for CTR prediction.

Empirically, we apply DKN to a real-world dataset from Bing News with extensive experiments. The results show that DKN achieves substantial gains over state-of-the-art deep-learning-based methods for recommendation. Specifically, DKN significantly outperforms baselines by 2.8% to 17.0% on F1 and 2.6% to 16.1% on AUC with a significance level of 0.1. The results also prove that the usage of knowledge and an attention module can bring additional 3.5% and 1.4% in improvement, respectively, in the DKN framework. Moreover, we present a visualization result of attention values to intuitively demonstrate the efficacy of the usage of the knowledge graph in Section 5.5.

2 PRELIMINARIES

In this section, we present several concepts and models related to this work, including knowledge graph embedding and convolutional neural networks for sentence representation learning.

2.1 Knowledge Graph Embedding

A typical knowledge graph consists of millions of entity-relation-entity triples (h, r, t), in which h, r and t represent the head, the relation, and the tail of a triple, respectively. Given all the triples in a knowledge graph, the goal of knowledge graph embedding is to learn a low-dimensional representation vector for each entity and relation that preserves the structural information of the original knowledge graph. Recently, translation-based knowledge graph embedding methods have received great attention due to their concise models and superior performance. To be self-contained, we briefly review these translation-based methods in the following.

- **TransE** [5] wants $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ when (h, r, t) holds, where $\mathbf{h}$, $\mathbf{r}$ and $\mathbf{t}$ are the corresponding representation vector of h, r and t. Therefore, TransE assumes the score function
$$f_r(h,t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2 \tag{1}$$
is low if (h, r, t) holds, and high otherwise.
- **TransH** [48] allows entities to have different representations when involved in different relations by projecting the entity embeddings into relation hyperplanes:
$$f_r(h,t) = \|\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\|_2^2, \tag{2}$$
where $\mathbf{h}_\perp = \mathbf{h} - \mathbf{w}_r^\top \mathbf{h} \mathbf{w}_r$ and $\mathbf{t}_\perp = \mathbf{t} - \mathbf{w}_r^\top \mathbf{t} \mathbf{w}_r$ are the projections of $\mathbf{h}$ and $\mathbf{t}$ to the hyperplane $\mathbf{w}_r$, respectively, and $\|\mathbf{w}_r\|_2 = 1$.
- **TransR** [28] introduces a projection matrix $\mathbf{M}_r$ for each relation r to map entity embeddings to the corresponding relation space. The score function in TransR is defined as
$$f_r(h,t) = \|\mathbf{h}_r + \mathbf{r} - \mathbf{t}_r\|_2^2, \tag{3}$$
where $\mathbf{h}_r = \mathbf{h}\mathbf{M}_r$ and $\mathbf{t}_r = \mathbf{t}\mathbf{M}_r$.
- **TransD** [20] replaces the projection matrix in TransR by the product of two projection vectors of an entity-relation pair:
$$f_r(h,t) = \|\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\|_2^2, \tag{4}$$
where $\mathbf{h}_\perp = (\mathbf{r}_p \mathbf{h}_p^\top + \mathbf{I})\mathbf{h}$, $\mathbf{t}_\perp = (\mathbf{r}_p \mathbf{t}_p^\top + \mathbf{I})\mathbf{t}$, $\mathbf{h}_p$, $\mathbf{r}_p$ and $\mathbf{t}_p$ are another set of vectors for entities and relations, and $\mathbf{I}$ is the identity matrix.

To encourage the discrimination between correct triples and incorrect triples, for all the methods above, the following margin-based ranking loss is used for training:
$$\mathcal{L} = \sum_{(h,r,t)\in\Delta} \sum_{(h',r,t')\in\Delta'} \max\left(0, f_r(h,t) + \gamma - f_r(h',t')\right), \tag{5}$$
where γ is the margin, Δ and Δ' are the set of correct triples and incorrect triples.

2.2 CNN for Sentence Representation Learning

Traditional methods [1, 44] usually represent sentences using the bag-of-words (BOW) technique, i.e., taking word counting statistics as the feature of sentences. However, BOW-based methods ignore word orders in sentences and are vulnerable to the sparsity problem, which leads to poor generalization performance. A more effective way to model sentences is to represent each sentence in a given corpus as a distributed low-dimensional vector. Recently, inspired by the success of applying convolutional neural networks (CNN) in the filed of computer vision [25], researchers have proposed

[5] https://www.google.com/intl/bn/insidesearch/features/search/knowledge.html
[6] https://searchengineland.com/library/bing/bing-satori

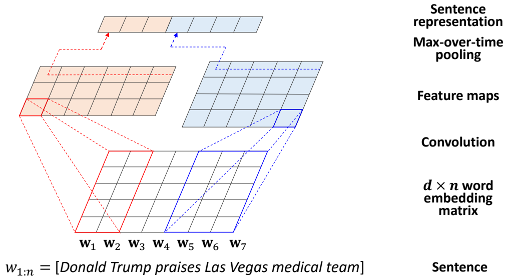


Figure 2: A typical architecture of CNN for sentence representation learning [22].

many CNN-based models for sentence representation learning [8, 21, 22, 53] [7]. In this subsection, we introduce a typical type of CNN architecture, namely Kim CNN [22].

Figure 2 illustrates the architecture of Kim CNN. Let $w_{1:n}$ be the raw input of a sentence of length n, and $\mathbf{w}_{1:n} = [\mathbf{w}_1\ \mathbf{w}_2\ ...\ \mathbf{w}_n] \in \mathbb{R}^{d \times n}$ be the word embedding matrix of the input sentence, where $\mathbf{w}_i \in \mathbb{R}^{d \times 1}$ is the embedding of the i-th word in the sentence and d is the dimension of word embeddings. A convolution operation with filter $\boldsymbol{h} \in \mathbb{R}^{d \times l}$ is then applied to the word embedding matrix $\mathbf{w}_{1:n}$, where l $(l \leq n)$ is the window size of the filter. Specifically, a feature c_i is generated from a sub-matrix $\mathbf{w}_{i:i+l-1}$ by

$$c_i = f(\boldsymbol{h} * \mathbf{w}_{i:i+l-1} + b), \tag{6}$$

where f is a non-linear function, $*$ is the convolution operator, and $b \in \mathbb{R}$ is a bias. After applying the filter to every possible position in the word embedding matrix, a feature map

$$\mathbf{c} = [c_1, c_2, ..., c_{n-l+1}] \tag{7}$$

is obtained, then a max-over-time pooling operation is used on feature map $\mathbf{c}$ to identify the most significant feature:

$$\tilde{c} = \max\{\mathbf{c}\} = \max\{c_1, c_2, ..., c_{n-l+1}\}. \tag{8}$$

One can use multiple filters (with varying window sizes) to obtain multiple features, and these features are concatenated together to form the final sentence representation.

3 PROBLEM FORMULATION

We formulate the news recommendation problem in this paper as follows. For a given user i in the online news platform, we denote his click history as $\{t_1^i, t_2^i, ..., t_{N_i}^i\}$, where t_j^i $(j = 1, ..., N_i)$ is the title[8] of the j-th news clicked by user i, and N_i is the total number of user i's clicked news. Each news title t is composed of a sequence of words, i.e., $t = [w_1, w_2, ...]$, where each word w may be associated with an entity e in the knowledge graph. For example, in the title "Trump praises Las Vegas medical team", "Trump" is linked with the entity "Donald Trump", while "Las" and "Vegas" are linked with the entity "Las Vegas". Given users' click history as well as the connection between words in news titles and entities in the knowledge graph, we aim to predict whether user i will click a candidate news t_j that he has not seen before.

4 DEEP KNOWLEDGE-AWARE NETWORK

In this section, we present the proposed DKN model in detail. We first introduce the overall framework of DKN, then discuss the process of knowledge distillation from a knowledge graph, the design of knowledge-aware convolutional neural networks (KCNN), and the attention-based user interest extraction, respectively.

4.1 DKN Framework

The framework of DKN is illustrated in Figure 3. We introduce the architecture of DKN from the bottom up. As shown in Figure 3, DKN takes one piece of candidate news and one piece of a user's clicked news as input. For each piece of news, a specially designed KCNN is used to process its title and generate an embedding vector. KCNN is an extension of traditional CNN that allows flexibility in incorporating symbolic knowledge from a knowledge graph into sentence representation learning. We will detail the process of knowledge distillation in Section 4.2 and the KCNN module in Section 4.3, respectively. By KCNN, we obtain a set of embedding vectors for a user's clicked history. To get final embedding of the user with respect to the current candidate news, we use an attention-based method to automatically match the candidate news to each piece of his clicked news, and aggregate the user's historical interests with different weights. The details of attention-based user interest extraction are presented in Section 4.4. The candidate news embedding and the user embedding are concatenated and fed into a deep neural network (DNN) to calculate the predicted probability that the user will click the candidate news.

4.2 Knowledge Distillation

The process of knowledge distillation is illustrated in Figure 4, which consists of four steps. First, to distinguish knowledge entities in news content, we utilize the technique of *entity linking* [33, 38] to disambiguate mentions in texts by associating them with predefined entities in a knowledge graph. Based on these identified entities, we construct a sub-graph and extract all relational links among them from the original knowledge graph. Note that the relations among identified entities only may be sparse and lack diversity. Therefore, we expand the knowledge sub-graph to all entities within one hop of identified ones. Given the extracted knowledge graph, a great many knowledge graph embedding methods, such as TransE [5], TransH [48], TransR [28], and TransD [20], can be utilized for entity representation learning. Learned entity embeddings are taken as the input for KCNN in the DKN framework.

It should be noted that though state-of-the-art knowledge graph embedding methods could generally preserve the structural information in the original graph, we find that the information of learned embedding for a single entity is still limited when used in subsequent recommendations. To help identify the position of entities in the knowledge graph, we propose extracting additional contextual information for each entity. The "context" of entity e is defined as

[7] Researchers have also proposed other types of neural network models for sentence modeling such as recurrent neural networks [42], recursive neural networks [40], and hybrid models [26]. However, CNN-based models are empirically proven to be superior than others [17], since they can detect and extract specific local patterns from sentences due to the convolution operation. To keep our presentation focused, we only discuss CNN-based models in this paper.

[8] In addition to title, it is also viable to use abstracts or snippets of news. In this paper, we only take news titles as input, since a title is a decisive factor affecting users' choice of reading. But note that our approach can be easily generalized to any sort of news-related texts.

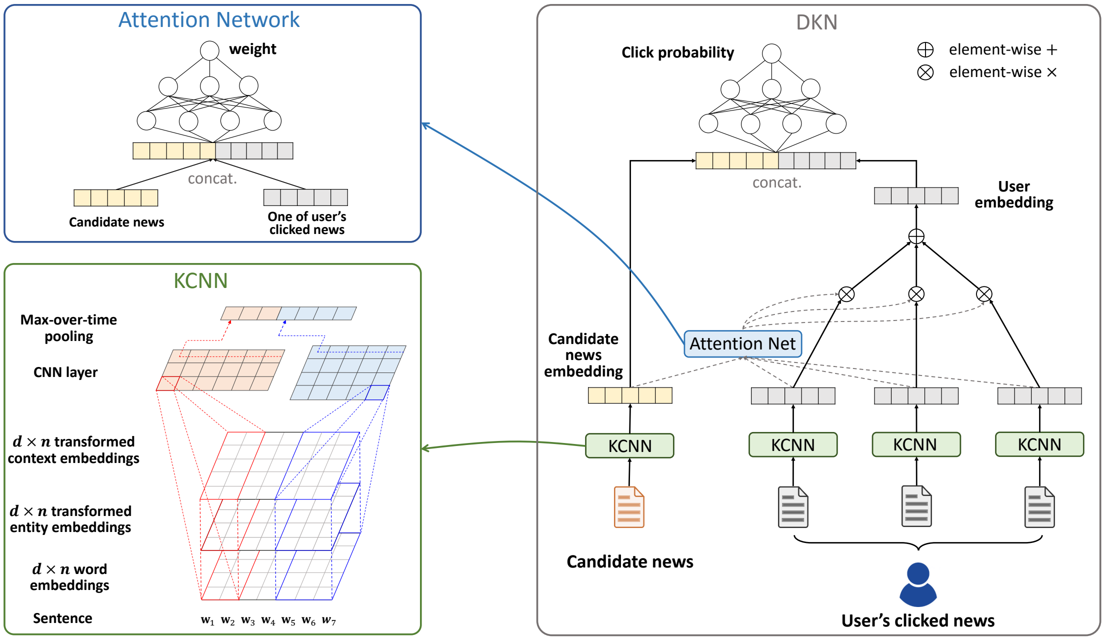
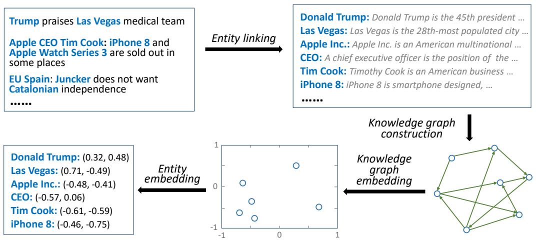


Figure 3: Illustration of the DKN framework.



Trump praises Las Vegas medical team

Apple CEO Tim Cook: iPhone 8 and Apple Watch Series 3 are sold out in some places

EU Spain: Juncker does not want Catalonian independence

......

Donald Trump: *Donald Trump is the 45th president ...*
Las Vegas: *Las Vegas is the 28th-most populated city ...*
Apple Inc.: *Apple Inc. is an American multinational ...*
CEO: *A chief executive officer is the position of the ...*
Tim Cook: *Timothy Cook is an American business ...*
iPhone 8: *iPhone 8 is smartphone designed, ...*
......

Donald Trump: (0.32, 0.48)
Las Vegas: (0.71, -0.49)
Apple Inc.: (-0.48, -0.41)
CEO: (-0.57, 0.06)
Tim Cook: (-0.61, -0.59)
iPhone 8: (-0.46, -0.75)

Figure 4: Illustration of knowledge distillation process.

the set of its immediate neighbors in the knowledge graph, i.e.,

$$context(e) = \{e_i \mid (e, r, e_i) \in \mathcal{G} \text{ or } (e_i, r, e) \in \mathcal{G}\}, \tag{9}$$

where r is a relation and $\mathcal{G}$ is the knowledge graph. Since the contextual entities are usually closely related to the current entity with respect to semantics and logic, the usage of context could provide more complementary information and assist in improving the identifiability of entities. Figure 5 illustrates an example of context. In addition to use the embedding of "Fight Club" itself to represent the entity, we also include its contexts, such as "Suspense" (genre), "Brad Pitt" (actor), "United States" (country) and "Oscars" (award), as its identifiers. Given the context of entity e, the *context embedding* is calculated as the average of its contextual entities:

$$\bar{\mathbf{e}} = \frac{1}{|context(e)|} \sum_{e_i \in context(e)} \mathbf{e}_i, \tag{10}$$

where $\mathbf{e}_i$ is the *entity embedding* of e_i learned by knowledge graph embedding. We empirically demonstrate the efficacy of context embedding in the experiment section.

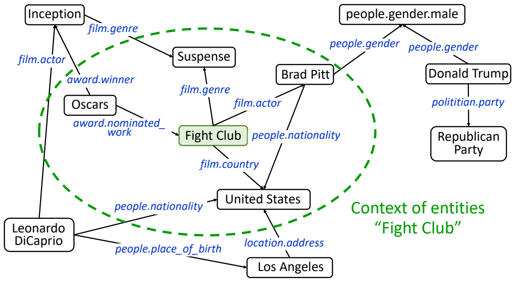


Figure 5: Illustration of context of an entity in a knowledge graph.

4.3 Knowledge-aware CNN

Following the notations used in Section 2.2, we use $t = w_{1:n} = [w_1, w_2, ..., w_n]$ to denote the raw input sequence of a news title t of length n, and $\mathbf{w}_{1:n} = [\mathbf{w}_1\ \mathbf{w}_2\ ...\ \mathbf{w}_n] \in \mathbb{R}^{d \times n}$ to denote the word embedding matrix of the title, which can be pre-learned from a large corpus or randomly initialized. After the knowledge distillation introduced in Section 4.2, each word w_i may also be associated with an entity embedding $\mathbf{e}_i \in \mathbb{R}^{k \times 1}$ and the corresponding context embedding $\bar{\mathbf{e}}_i \in \mathbb{R}^{k \times 1}$, where k is the dimension of entity embedding.

Given the input above, a straightforward way to combine words and associated entities is to treat the entities as "pseudo words" and concatenate them to the word sequence [46], i.e.,

$$\mathbf{W} = [\mathbf{w}_1\ \mathbf{w}_2\ ...\ \mathbf{w}_n\ \mathbf{e}_{t_1}\ \mathbf{e}_{t_2}\ ...], \tag{11}$$

where $\{\mathbf{e}_{t_j}\}$ is the set of entity embeddings associated with this news title. The obtained new sentence $\mathbf{W}$ is fed into CNN [22] for further processing. However, we argue that this simple concatenating strategy has the following limitations: 1) The concatenating strategy breaks up the connection between words and associated entities and is unaware of their alignment. 2) Word embeddings and entity embeddings are learned by different methods, meaning it is not suitable to convolute them together in a single vector space. 3) The concatenating strategy implicitly forces word embeddings and entity embeddings to have the same dimension, which may not be optimal in practical settings since the optimal dimensions for word and entity embeddings may differ from each other.

Being aware of the above limitations, we propose a *multi-channel* and *word-entity-aligned* KCNN for combining word semantics and knowledge information. The architecture of KCNN is illustrated in the left lower part of Figure 3. For each news title $t = [w_1, w_2, ..., w_n]$, in addition to use its word embeddings $\mathbf{w}_{1:n} = [\mathbf{w}_1\ \mathbf{w}_2\ ...\ \mathbf{w}_n]$ as input, we also introduce the *transformed entity embeddings*

$$g(\mathbf{e}_{1:n}) = [g(\mathbf{e}_1)\ g(\mathbf{e}_2)\ ...\ g(\mathbf{e}_n)] \tag{12}$$

and *transformed context embeddings*

$$g(\overline{\mathbf{e}}_{1:n}) = [g(\overline{\mathbf{e}}_1)\ g(\overline{\mathbf{e}}_2)\ ...\ g(\overline{\mathbf{e}}_n)] \tag{13}$$

as source of input[9], where g is the transformation function. In KCNN, g can be either linear

$$g(\mathbf{e}) = \mathbf{Me} \tag{14}$$

or non-linear

$$g(\mathbf{e}) = \tanh(\mathbf{Me} + \mathbf{b}), \tag{15}$$

where $\mathbf{M} \in \mathbb{R}^{d \times k}$ is the trainable transformation matrix and $\mathbf{b} \in \mathbb{R}^{d \times 1}$ is the trainable bias. Since the transformation function is continuous, it can map the entity embeddings and context embeddings from the entity space to the word space while preserving their original spatial relationship. Note that word embeddings $\mathbf{w}_{1:n}$, transformed entity embeddings $g(\mathbf{e}_{1:n})$ and transformed context embeddings $g(\overline{\mathbf{e}}_{1:n})$ are the same size and serve as the multiple channels analogous to colored images. We therefore align and stack the three embedding matrices as

$$\mathbf{W} = \big[[\mathbf{w}_1\ g(\mathbf{e}_1)\ g(\overline{\mathbf{e}}_1)]\ [\mathbf{w}_2\ g(\mathbf{e}_2)\ \overline{g}(\mathbf{e}_2)]\ ...\ [\mathbf{e}_n\ g(\mathbf{e}_n)\ g(\overline{\mathbf{e}}_n)]\big] \in \mathbb{R}^{d \times n \times 3}. \tag{16}$$

After getting the multi-channel input $\mathbf{W}$, similar to Kim CNN [22], we apply multiple filters $\boldsymbol{h} \in \mathbb{R}^{d \times l \times 3}$ with varying window sizes l to extract specific local patterns in the news title. The local activation of sub-matrix $\mathbf{W}_{i:i+l-1}$ with respect to $\boldsymbol{h}$ can be written as

$$c_i^h = f(\boldsymbol{h} * \mathbf{W}_{i:i+l-1} + b), \tag{17}$$

and we use a max-over-time pooling operation on the output feature map to choose the largest feature:

$$\tilde{c}^h = \max\{c_1^h, c_2^h, ..., c_{n-l+1}^h\}. \tag{18}$$

All features $\tilde{c}^{h_i}$ are concatenated together and taken as the final representation $\mathbf{e}(t)$ of the input news title t, i.e.,

$$\mathbf{e}(t) = [\tilde{c}^{h_1}\ \tilde{c}^{h_2}\ ...\ \tilde{c}^{h_m}], \tag{19}$$

where m is the number of filters.

4.4 Attention-based User Interest Extraction

Given user i with clicked history $\{t_1^i, t_2^i, ..., t_{N_i}^i\}$, the embeddings of his clicked news can be written as $\mathbf{e}(t_1^i)$, $\mathbf{e}(t_2^i)$, ..., $\mathbf{e}(t_{N_i}^i)$. To represent user i for the current candidate news t_j, one can simply average all the embeddings of his clicked news titles:

$$\mathbf{e}(i) = \frac{1}{N_i} \sum_{k=1}^{N_i} \mathbf{e}(t_k^i). \tag{20}$$

However, as discussed in the introduction, a user's interest in news topics may be various, and user i's clicked items are supposed to have different impacts on the candidate news t_j when considering whether user i will click t_j. To characterize user's diverse interests, we use an attention network [47, 54] to model the different impacts of the user's clicked news on the candidate news. The attention network is illustrated in the left upper part of Figure 3. Specifically, for user i's clicked news t_k^i and candidate news t_j, we first concatenate their embeddings, then apply a DNN $\mathcal{H}$ as the attention network and the softmax function to calculate the normalized impact weight:

$$s_{t_k^i, t_j} = \text{softmax}\Big(\mathcal{H}\big(\mathbf{e}(t_k^i), \mathbf{e}(t_j)\big)\Big) = \frac{\exp\Big(\mathcal{H}\big(\mathbf{e}(t_k^i), \mathbf{e}(t_j)\big)\Big)}{\sum_{k=1}^{N_i} \exp\Big(\mathcal{H}\big(\mathbf{e}(t_k^i), \mathbf{e}(t_j)\big)\Big)}. \tag{21}$$

The attention network $\mathcal{H}$ receives embeddings of two news titles as input and outputs the impact weight. The embedding of user i with respect to the candidate news t_j can thus be calculated as the weighted sum of his clicked news title embeddings:

$$\mathbf{e}(i) = \sum_{k=1}^{N_i} s_{t_k^i, t_j} \mathbf{e}(t_k^i). \tag{22}$$

Finally, given user i's embedding $\mathbf{e}(i)$ and candidate news t_j's embedding $\mathbf{e}(t_j)$, the probability of user i clicking news t_j is predicted by another DNN $\mathcal{G}$:

$$p_{i, t_j} = \mathcal{G}\big(\mathbf{e}(i), \mathbf{e}(t_j)\big). \tag{23}$$

We will demonstrate the efficacy of the attention network in the experiment section.

5 EXPERIMENTS

In this section, we present our experiments and the corresponding results, including dataset analysis and comparison of models. We also give a case study about user's reading interests and make discussions on tuning hyper-parameters.

5.1 Dataset Description

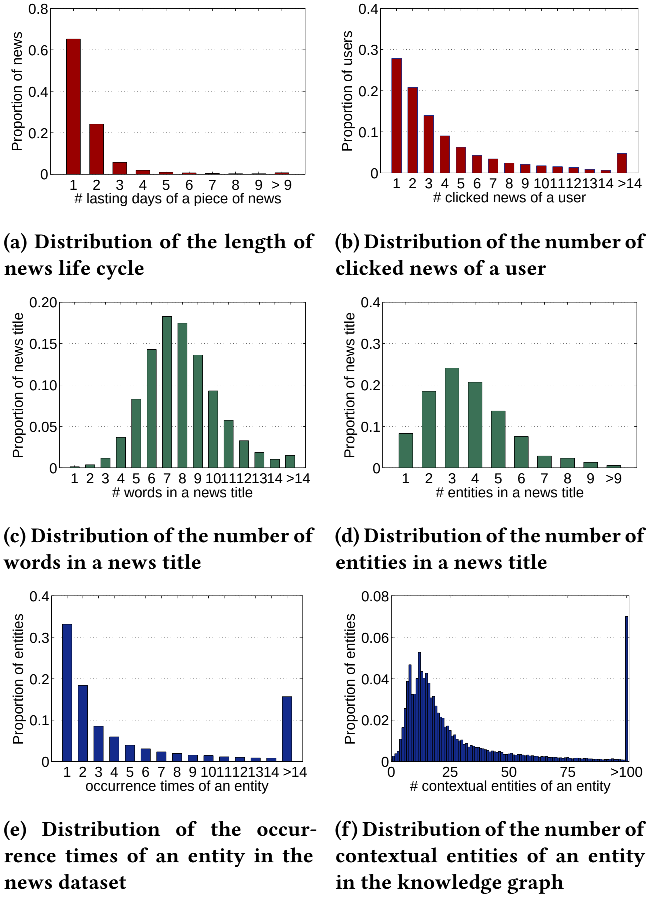
Our dataset comes from the server logs of Bing News. Each piece of log mainly contains the timestamp, user id, news url, news title, and click count (0 for no click and 1 for click). We collect a randomly sampled and balanced dataset from October 16, 2016 to June 11, 2017 as the training set, and from June 12, 2017 to August 11, 2017 as the test set. Additionally, we search all occurred entities in the dataset as well as the ones within their one hop in the Microsoft Satori knowledge graph, and extract all edges (triples) among them with confidence greater than 0.8. The basic statistics and distributions of the news dataset and the extracted knowledge graph are shown in Table 1 and Figure 6, respectively.

[9] $\mathbf{e}_i$ and $\overline{\mathbf{e}}_i$ are set as zero if w_i has no corresponding entity.

Table 1: Basic statistics of the news dataset and the extracted knowledge graph.

# users	141,487	# triples	7,145,776
# news	535,145	avg. # words per title	7.9
# logs	1,025,192	avg. # entities per title	3.7
# entities	336,350	avg. # contextual entities per entity	42.5
# relations	4,668		

"#" denotes "the number of".



(a) Distribution of the length of news life cycle

(b) Distribution of the number of clicked news of a user

(c) Distribution of the number of words in a news title

(d) Distribution of the number of entities in a news title

(e) Distribution of the occurrence times of an entity in the news dataset

(f) Distribution of the number of contextual entities of an entity in the knowledge graph

Figure 6: Illustration of statistical distributions in news dataset and extracted knowledge graph.

Figure 6a illustrates the distribution of the length of the news life cycle, where we define the life cycle of a piece of news as the period from its publication date to the date of its last received click. We observe that about 90% of news are clicked within two days, which proves that online news is extremely time-sensitive and are substituted by newer ones with high frequency. Figure 6b illustrates the distribution of the number of clicked pieces of news for a user. 77.9% of users clicked no more than five pieces of news, which demonstrates the data sparsity in the news recommendation scenario.

Figures 6c and 6d illustrate the distributions of the number of words (without stop words) and entities in a news title, respectively. The average number per title is 7.9 for words and 3.7 for entities, showing that there is one entity in almost every two words in news titles on average. The high density of the occurrence of entities also empirically justifies the design of KCNN.

Figures 6e and 6f present the distribution of occurrence times of an entity in the news dataset and the distribution of the number of contextual entities of an entity in extracted knowledge graph, respectively. We can conclude from the two figures that the occurrence pattern of entities in online news is sparse and has a long tail (80.4% of entities occur no more than ten times), but entities generally have abundant contexts in the knowledge graph: the average number of context entities per entity is 42.5 and the maximum is 140, 737. Therefore, contextual entities can greatly enrich the representations for a single entity in news recommendation.

5.2 Baselines

We use the following state-of-the-art methods as baselines in our experiments:

- **LibFM** [37] is a state-of-the-art feature-based factorization model and widely used in CTR scenarios. In this paper, the input feature of each piece of news for LibFM is comprised of two parts: TF-IDF features and averaged entity embeddings. We concatenate the feature of a user and candidate news to feed into LibFM.
- **KPCNN** [46] attaches the contained entities to the word sequence of a news title and uses Kim CNN to learn representations of news, as introduced in Section 4.3.
- **DSSM** [18] is a deep structured semantic model for document ranking using word hashing and multiple fully-connected layers. In this paper, the user's clicked news is treated as the query and the candidate news are treated as the documents.
- **DeepWide** [7] is a general deep model for recommendation, combining a (wide) linear channel with a (deep) non-linear channel. Similar to LibFM, we use the concatenated TF-IDF features and averaged entity embeddings as input to feed both channels.
- **DeepFM** [15] is also a general deep model for recommendation, which combines a component of factorization machines and a component of deep neural networks that share the input. We use the same input as in LibFM for DeepFM.
- **YouTubeNet** [9] is proposed to recommend videos from a large-scale candidate set in YouTube using a deep candidate generation network and a deep ranking network. In this paper, we adapt the deep raking network to the news recommendation scenario.
- **DMF** [50] is a deep matrix factorization model for recommender systems which uses multiple non-linear layers to process raw rating vectors of users and items. We ignore the content of news and take the implicit feedback as input for DMF.

Note that except for LibFM, other baselines are all based on deep neural networks since we aim to compare our approach with state-of-the-art deep learning models. Additionally, except for DMF which is based on collaborative filtering, other baselines are all content-based or hybrid methods.

Table 2: Comparison of different models.

Models*	F1	AUC	p-value**
DKN	**68.9 ± 1.5**	**65.9 ± 1.2**	–
LibFM	61.8 ± 2.1 (-10.3%)	59.7 ± 1.8 (-9.4%)	$< 10^{-3}$
LibFM(-)	61.1 ± 1.9 (-11.3%)	58.9 ± 1.7 (-10.6%)	$< 10^{-3}$
KPCNN	67.0 ± 1.6 (-2.8%)	64.2 ± 1.4 (-2.6%)	0.098
KPCNN(-)	65.8 ± 1.4 (-4.5%)	63.1 ± 1.5 (-4.2%)	0.036
DSSM	66.7 ± 1.8 (-3.2%)	63.6 ± 2.0 (-3.5%)	0.063
DSSM(-)	66.1 ± 1.6 (-4.1%)	63.2 ± 1.8 (-4.1%)	0.045
DeepWide	66.0 ±1.2 (-4.2%)	63.3 ± 1.5 (-3.9%)	0.039
DeepWide(-)	63.7 ± 0.9 (-7.5%)	61.5 ± 1.1 (-6.7%)	0.004
DeepFM	63.8 ± 1.5 (-7.4%)	61.2 ± 2.3 (-7.1%)	0.014
DeepFM(-)	64.0 ± 1.9 (-7.1%)	61.1 ± 1.8 (-7.3%)	0.007
YouTubeNet	65.5 ± 1.2 (-4.9%)	63.0 ± 1.4 (-4.4%)	0.025
YouTubeNet(-)	65.1 ± 0.7 (-5.5%)	62.1 ± 1.3 (-5.8%)	0.011
DMF	57.2 ± 1.2 (-17.0%)	55.3 ± 1.0 (-16.1%)	$< 10^{-3}$

* "(-)" denotes "without input of entity embeddings".

** p-value is the probability of no significant difference with DKN on AUC by t-test.

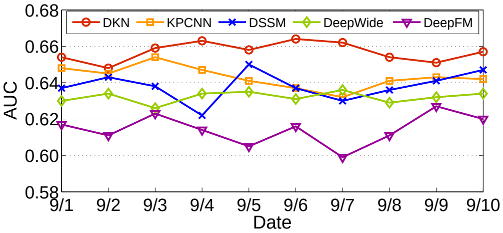


Figure 7: AUC score of DKN and baselines over ten days (Sep. 01-10, 2017).

5.3 Experiment Setup

We choose TransD [20] to process the knowledge graph and learn entity embeddings, and use the non-linear transformation function in Eq. (15) in KCNN. The dimension of both word embeddings and entity embeddings are set as 100. The number of filters are set as 100 for each of the window sizes 1, 2, 3, 4. We use Adam [23] to train DKN by optimizing the log loss. We will further study the variants of DKN and the sensitivity of key parameters in Sections 5.4 and 5.6, respectively. To compare DKN with baselines, we use *F1* and *AUC* value as the evaluation metrics.

The key parameter settings for baselines are as follows. For KPCNN, the dimensions of word embeddings and entity embeddings are both set as 100. For DSSM, the dimension of semantic feature is set as 100. For DeepWide, the final representations for deep and wide components are both set as 100. For YouTubeNet, the dimension of final layer is set as 100. For LibFM and DeepFM, the dimensionality of the factorization machine is set as $\{1, 1, 0\}$. For DMF, the dimension of latent representation for users and items is set as 100. The above settings are for fair consideration. Other parameters in the baselines are set as default. Each experiment is repeated five times, and we report the average and maximum deviation as results.

5.4 Results

In this subsection, we present the results of comparison of different models and the comparison among variants of DKN.

Table 3: Comparison among DKN variants.

Variants	F1	AUC
DKN with entity and context emd.	**68.8 ± 1.4**	**65.7 ± 1.1**
DKN with entity emd. only	67.2 ± 1.2	64.8 ± 1.0
DKN with context emd. only	66.5 ± 1.5	64.2 ± 1.3
DKN without entity nor context emd.	66.1 ±1.4	63.5 ± 1.1
DKN + TransE	67.6 ± 1.6	65.0 ± 1.3
DKN + TransH	67.3 ± 1.3	64.7 ± 1.2
DKN + TransR	67.9 ± 1.5	65.1 ± 1.5
DKN + TransD	**68.8 ± 1.3**	**65.8 ± 1.4**
DKN with non-linear mapping	**69.0 ± 1.7**	**66.1 ± 1.4**
DKN with linear mapping	67.1 ± 1.5	64.9 ± 1.3
DKN without mapping	66.7 ± 1.6	63.7 ± 1.6
DKN with attention	**68.7 ± 1.3**	**65.7 ± 1.2**
DKN without attention	67.0 ± 1.0	64.8 ± 0.8

5.4.1 Comparison of different models. The results of comparison of different models are shown in Table 2. For each baseline in which the input contains entity embedding, we also remove the entity embedding from input to see how its performance changes (denoted by "(-)"). Additionally, we list the improvements of baselines compared with DKN in brackets and calculate the p-value of statistical significance by t-test. Several observations stand out from Table 2:

- The usage of entity embedding could boost the performance of most baselines. For example, the AUC of KPCNN, DeepWide, and YouTubeNet increases by 1.1%, 1.8% and 1.1%, respectively. However, the improvement for DeepFM is less obvious. We try different parameter settings for DeepFM and find that if the AUC drops to about 0.6, the improvement brought by the usage of knowledge could be up to 0.5%. The results show that FM-based method cannot take advantage of entity embedding stably in news recommendation.
- DMF performs worst among all methods. This is because DMF is a CF-based method, but news is generally highly time-sensitive with a short life cycle. The result proves our aforementioned claim that CF methods cannot work well in the news recommendation scenario.
- Except for DMF, other deep-learning-based baselines outperform LibFM by 2.0% to 5.2% on F1 and by 1.5% to 4.5% on AUC, which suggests that deep models are effective in capturing the non-linear relations and dependencies in news data.
- The architecture of DeepWide and YouTubeNet is similar in the news recommendation scenario, thus we can observe comparable performance of the two methods. DSSM outperforms DeepWide and YouTubeNet, the reason for which might be that DSSM models raw texts directly with word hashing.
- KPCNN performs best in all baselines. This is because KPCNN uses CNN to process input texts and can better extract the specific local patterns in sentences.
- Finally, compared with KPCNN, DKN can still have a 1.7% AUC increase. We attribute the superiority of DKN to its two properties: 1) DKN uses word-entity-aligned KCNN for sentence representation learning, which could better preserve the relatedness between words and entities; 2) DKN uses an attention network to treat users' click history discriminatively, which better captures users' diverse reading interests.

Table 4: Illustration of training and test logs for a randomly sampled user (training logs with label 0 are omitted).

	No.	Date	News title	Entities	Label	Category
training	1	12/25/2016	Elon Musk teases huge upgrades for Tesla's supercharger network	Elon Musk; Tesla Inc.	1	Cars
	2	03/25/2017	Elon Musk offers Tesla Model 3 sneak peek	Elon Musk; Tesla Model 3	1	Cars
	3	12/14/2016	Google fumbles while Tesla sprints toward a driverless future	Google Inc.; Tesla Inc.	1	Cars
	4	12/15/2016	Trump pledges aid to Silicon Valley during tech meeting	Donald Trump; Silicon Valley	1	Politics
	5	03/26/2017	Donald Trump is a big reason why the GOP kept the Montana House seat	Donald Trump; GOP; Montana	1	Politics
	6	05/03/2017	North Korea threat: Kim could use nuclear weapons as "blackmail"	North Korea; Kim Jong-un	1	Politics
	7	12/22/2016	Microsoft sells out of unlocked Lumia 950 and Lumia 950 XL in the US	Microsoft; Lumia; United States	1	Other
	8	12/08/2017	6.5 magnitude earthquake recorded off the coast of California	earthquake; California	1	Other
			......			
test	1	07/08/2017	Tesla makes its first Model 3	Tesla Inc; Tesla Model 3	1	Cars
	2	08/13/2017	General Motors is ramping up its self-driving car: Ford should be nervous	General Motors; Ford Inc.	1	Cars
	3	06/21/2017	Jeh Johnson testifies on Russian interference in 2016 election	Jeh Johnson; Russian	1	Politics
	4	07/16/2017	"Game of Thrones" season 7 premiere: how you can watch	Game of Thrones	0	Other

Figure 7 presents the AUC score of DKN and baselines for additional ten test days. We can observe that the curve of DKN is consistently above baselines over ten days, which strongly proves the competitiveness of DKN. Moreover, the performance of DKN is also with low variance compared with baselines, which suggests that DKN is also robust and stable in practical application.

5.4.2 Comparison among DKN variants. Further, we compare among the variants of DKN with respect to the following four aspects to demonstrate the efficacy of the design of the DKN framework: the usage of knowledge, the choice of knowledge graph embedding method, the choice of transformation function, and the usage of an attention network. The results are shown in Table 3, from which we can conclude that:

- The usage of entity embedding and contextual embedding can improve AUC by 1.3% and 0.7%, respectively, and we can achieve even better performance by combining them together. This finding confirms the efficacy of using a knowledge graph in the DKN model.
- DKN+TransD outperforms other combinations. This is probably because, as presented in Section 2.1, TransD is the most complicated model among the four embedding methods, which is able to better capture non-linear relationships among the knowledge graph for news recommendation.
- DKN with mapping is better than DKN without mapping, and the non-linear function is superior to the linear one. The results prove that the transformation function can alleviate the heterogeneity between word and entity spaces by self learning, and the non-linear function can achieve better performance.
- The attention network brings a 1.7% gain on F1 and 0.9% gain on AUC for the DKN model. We will give a more intuitive demonstration on the attention network in the next subsection.

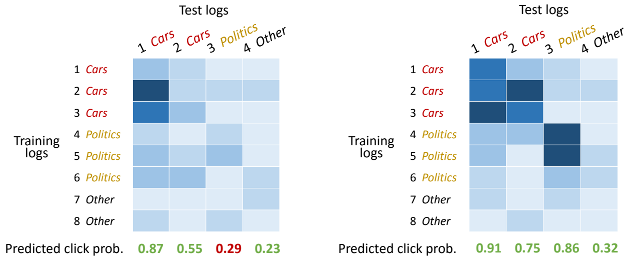
5.5 Case Study

To intuitively demonstrate the efficacy of the usage of the knowledge graph as well as the the attention network, we randomly sample a user and extract all his logs from the training set and the test set (training logs with label 0 are omitted for simplicity). As shown in Table 4, the clicked news clearly exhibits his points of interest: No. 1-3 concern cars and No. 4-6 concern politics (categories are not contained in the original dataset but manually tagged by us). We use the whole training data to train DKN with full features and DKN without entity nor context embedding, then feed each possible pair of training logs and test logs of this user to the two trained models and obtain the output value of their attention networks. The results are visualized in Figure 8, in which the darker shade of blue indicates larger attention values. From Figure 8a we observe that, the first title in test logs gets high attention values with "Cars" in the training logs since they share the same word "Tesla", but the results for the second title are less satisfactory, since the second title shares no explicit word-similarity with any title in the training set, including No. 1-3. The case is similar for the third title in test logs. In contrast, in Figure 8b we see that the attention network precisely captures the relatedness within the two categories "Cars" and "Politics". This is because in the knowledge graph, "General Motors" and "Ford Inc." share a large amount of context with "Tesla Inc." and "Elon Musk", moreover, "Jeh Johnson" and "Russian" are also highly connected to "Donald Trump". The difference in the response of the attention network also affects the final predicted results: DKN with knowledge graph (Figure 8b) accurately predicts all the test logs, while DKN without knowledge graph (Figure 8a) fails on the third one.

5.6 Parameter Sensitivity

DKN involves a number of hyper-parameters. In this subsection, we examine how different choices of hyper-parameters affect the performance of DKN. In the following experiments, expect for the parameter being tested, all other parameters are set as introduced in Section 5.3.

5.6.1 Dimension of word embedding d and dimension of entity embedding k. We first investigate how the dimension of word embedding d and dimension of entity embedding k affect performance by testing all combinations of d and k in set $\{20, 50, 100, 200\}$. The results are shown in Figure 9a, from which we can observe that, given dimension of entity embedding k, performance initially improves with the increase of dimension of word embedding d. This is because more bits in word embedding can encode more useful information of word semantics. However, the performance drops when d further increases, as a too large d (e.g., $d = 200$) may introduce noises which mislead the subsequent prediction. The case is similar for k when d is given.



(a) without knowledge graph

(b) with knowledge graph

Figure 8: Attention visualization for training logs and test logs for a randomly sampled user.

5.6.2 Window sizes of filters and the number of filters m. We further investigate the choice of windows sizes of filters and the number of filters for KCNN in the DKN model. As shown in Figure 9b, given windows sizes, the AUC score generally increases as the number of filters m gets larger, since more filters are able to capture more local patterns in input sentences and enhance model capability. However, the trend changes when m is too large ($m = 200$) due to probable overfitting. Likewise, we can observe similar rules for window sizes given m: a small window size cannot capture long-distance patterns in sentences, while a too large window size may easily suffer from overfitting the noisy patterns.

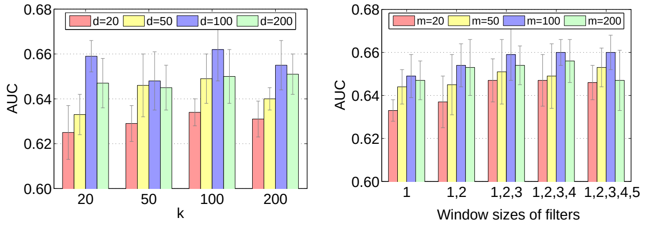
6 RELATED WORK

6.1 News Recommendation

News recommendation has previously been widely studied. Non-personalized news recommendation aims to model relatedness among news [31] or learn human editors' demonstration [47]. In personalized news recommendation, CF-based methods [43] often suffer from the cold-start problem since news items are substituted frequently. Therefore, a large amount of content-based or hybrid methods have been proposed [2, 24, 29, 36, 41]. For example, [36] proposes a Bayesian method for predicting users' current news interests based on their click behavior, and [41] proposes an explicit localized sentiment analysis method for location-based news recommendation. Recently, researchers have also tried to combine other features into news recommendation, for example, contextual-bandit [27], topic models [30], and recurrent neural networks [34]. The major difference between prior work and ours is that we use a knowledge graph to extract latent knowledge-level connections among news for better exploration in news recommendation.

6.2 Knowledge Graph

Knowledge graph representation aims to learn a low-dimensional vector for each entity and relation in the knowledge graph, while preserving the original graph structure. In addition to translation-based methods [5, 20, 28, 48] used in DKN, researchers have also proposed many other models such as Structured Embedding [6], Latent Factor Model [19], Neural Tensor Network [39] and Semantic Matching Energy [4]. Recently, the knowledge graph has also been used in many applications, such as movie recommendation[52], top-N recommendation [35], machine reading[51], text classification[46] word embedding[49], and question answering [11]. To the best of our knowledge, this paper is the first work that proposes leveraging knowledge graph embedding in news recommendation.



(a) AUC score w.r.t dimension of entity embedding k and dimension of word embedding d

(b) AUC score w.r.t window sizes of filters and the number of filters m

Figure 9: Parameter sensitivity of DKN.

6.3 Deep Recommender Systems

Recently, deep learning has been revolutionizing recommender systems and achieves better performance in many recommendation scenarios. Roughly speaking, deep recommender systems can be classified into two categories: using deep neural networks to process the raw features of users or items, or using deep neural networks to model the interaction among users and items. In addition to the aforementioned DSSM [18], DeepWide [7], DeepFM [15], YouTubeNet [9] and DMF [50], other popular deep-learning-based recommender systems include Collaborative Deep Learning [45], Hybrid CF-SDAE [12], Multi-view Deep Learning [13], and Neural Collaborative Filtering [16]. The major difference between these methods and ours is that DKN specializes in news recommendation and could achieve better performance than other generic deep recommender systems.

7 CONCLUSIONS

In this paper, we propose DKN, a deep knowledge-aware network that takes advantage of knowledge graph representation in news recommendation. DKN addresses three major challenges in news recommendation: 1) Different from ID-based methods such as collaborative filtering, DKN is a content-based deep model for click-through rate prediction that are suitable for highly time-sensitive news. 2) To make use of knowledge entities and common sense in news content, we design a KCNN module in DKN to jointly learn from semantic-level and knowledge-level representations of news. The multiple channels and alignment of words and entities enable KCNN to combine information from heterogeneous sources and maintain the correspondence of different embeddings for each word. 3) To model the different impacts of a user's diverse historical interests on current candidate news, DKN uses an attention module to dynamically calculate a user's aggregated historical representation. We conduct extensive experiments on a dataset from Bing News. The results demonstrate the significant superiority of DKN compared with strong baselines, as well as the efficacy of the usage of knowledge entity embedding and the attention module. For future work, we plan to model the relationship of knowledge entities in news content more directly based on graph theory and graph algorithms.

REFERENCES

[1] Deepak Agarwal and Bee-Chung Chen. 2009. Regression-based latent factor models. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 19–28.

[2] Trapit Bansal, Mrinal Das, and Chiranjib Bhattacharyya. 2015. Content driven user profiling for comment-worthy recommendations of news and blog articles. In *Proceedings of the 9th ACM Conference on Recommender Systems*. ACM.

[3] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[4] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2014. A semantic matching energy function for learning with multi-relational data. *Machine Learning* 94, 2 (2014), 233–259.

[5] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*. 2787–2795.

[6] Antoine Bordes, Jason Weston, Ronan Collobert, Yoshua Bengio, and others. 2011. Learning Structured Embeddings of Knowledge Bases.. In *AAAI*, Vol. 6. 6.

[7] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, and others. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 7–10.

[8] Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. 2016. Very deep convolutional networks for natural language processing. *arXiv preprint arXiv:1606.01781* (2016).

[9] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 191–198.

[10] Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J Smola, Jing Jiang, and Chong Wang. 2014. Jointly modeling aspects, ratings and sentiments for movie recommendation (jmars). In *KDD*. ACM, 193–202.

[11] Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2015. Question Answering over Freebase with Multi-Column Convolutional Neural Networks.. In *ACL (1)*.

[12] Xin Dong, Lei Yu, Zhonghuo Wu, Yuxia Sun, Lingfeng Yuan, and Fangxi Zhang. 2017. A Hybrid Collaborative Filtering Model with Deep Structure for Recommender Systems.. In *AAAI*. 1309–1315.

[13] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. 2015. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 278–288.

[14] Yanjie Fu, Bin Liu, Yong Ge, Zijun Yao, and Hui Xiong. 2014. User preference learning with multiple information fusion for restaurant recommendation. In *Proceedings of the 2014 SIAM International Conference on Data Mining*. SIAM.

[15] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*.

[16] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*. International World Wide Web Conferences Steering Committee, 173–182.

[17] James Hong and Michael Fang. 2015. *Sentiment analysis with deeply learned distributed representations of variable length texts*. Technical Report. Technical report, Stanford University.

[18] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *CIKM*. ACM, 2333–2338.

[19] Rodolphe Jenatton, Nicolas L Roux, Antoine Bordes, and Guillaume R Obozinski. 2012. A latent factor model for highly multi-relational data. In *Advances in Neural Information Processing Systems*. 3167–3175.

[20] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge Graph Embedding via Dynamic Mapping Matrix. In *ACL*. 687–696.

[21] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188* (2014).

[22] Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*.

[23] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[24] Michal Kompan and Mária Bieliková. 2010. Content-Based News Recommendation. In *EC-Web*, Vol. 61. Springer, 61–72.

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[26] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent Convolutional Neural Networks for Text Classification.. In *AAAI*, Vol. 333. 2267–2273.

[27] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*. ACM, 661–670.

[28] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In *AAAI*.

[29] Jiahui Liu, Peter Dolan, and Elin Rønby Pedersen. 2010. Personalized news recommendation based on click behavior. In *Proceedings of the 15th international conference on Intelligent user interfaces*. ACM, 31–40.

[30] Tapio Luostarinen and Oskar Kohonen. 2013. Using topic models in content-based news recommender systems. In *Proceedings of the 19th Nordic Conference of Computational Linguistics*. Linköping University Electronic Press, 239–251.

[31] Yuanhua Lv, Taesup Moon, Pranam Kolari, Zhaohui Zheng, Xuanhui Wang, and Yi Chang. 2011. Learning to model relatedness for news recommendation. In *Proceedings of the 20th international conference on World wide web*. ACM, 57–66.

[32] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[33] David Milne and Ian H Witten. 2008. Learning to link with wikipedia. In *CIKM*. ACM, 509–518.

[34] Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. 2017. Embedding-based News Recommendation for Millions of Users. In *KDD*. ACM, 1933–1942.

[35] Enrico Palumbo, Giuseppe Rizzo, and Raphaël Troncy. 2017. entity2rec: Learning User-Item Relatedness from Knowledge Graphs for Top-N Item Recommendation. (2017).

[36] Owen Phelan, Kevin McCarthy, and Barry Smyth. 2009. Using twitter to recommend real-time topical news. In *Proceedings of the third ACM conference on Recommender systems*. ACM, 385–388.

[37] Steffen Rendle. 2012. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)* 3, 3 (2012), 57.

[38] Avirup Sil and Alexander Yates. 2013. Re-ranking for joint named-entity recognition and linking. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2369–2374.

[39] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*. 926–934.

[40] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*. 1631–1642.

[41] Jeong-Woo Son, A Kim, Seong-Bae Park, and others. 2013. A location-based news article recommendation with explicit localized semantic analysis. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 293–302.

[42] Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075* (2015).

[43] Chong Wang and David M Blei. 2011. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 448–456.

[44] Hongwei Wang, Jia Wang, Miao Zhao, Jiannong Cao, and Minyi Guo. 2017. Joint-Topic-Semantic-aware Social Recommendation for Online Voting. In *Proceedings of the 26th ACM International Conference on Conference on Information and Knowledge Management*. ACM, 347–356.

[45] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1235–1244.

[46] Jin Wang, Zhongyuan Wang, Dawei Zhang, and Jun Yan. 2017. Combining Knowledge with Deep Convolutional Neural Networks for Short Text Classification. In *Proceedings of the International Joint Conference on Artificial Intelligence*.

[47] Xuejian Wang, Lantao Yu, Kan Ren, Guanyu Tao, Weinan Zhang, Yong Yu, and Jun Wang. 2017. Dynamic Attention Deep Model for Article Recommendation by Learning Human Editors' Demonstration. In *KDD*. ACM.

[48] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. In *AAAI*. 1112–1119.

[49] Chang Xu, Yalong Bai, Jiang Bian, Bin Gao, Gang Wang, Xiaoguang Liu, and Tie-Yan Liu. 2014. Rc-net: A general framework for incorporating knowledge into word representations. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, 1219–1228.

[50] Hong-Jian Xue, Xin-Yu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. 2017. Deep Matrix Factorization Models for Recommender Systems. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*.

[51] Bishan Yang and Tom Mitchell. 2017. Leveraging knowledge bases in lstms for improving machine reading. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1.

[52] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *KDD*. ACM, 353–362.

[53] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS*. 649–657.

[54] Guorui Zhou, Chengru Song, Xiaoqiang Zhu, Xiao Ma, Yanghui Yan, Xingya Dai, Han Zhu, Junqi Jin, Han Li, and Kun Gai. 2017. Deep Interest Network for Click-Through Rate Prediction. *arXiv preprint arXiv:1706.06978* (2017).