KENRM : Knowledge Enhanced Neural Relevance Matching in Online Tourism Industry

Feng Wei*
Alibaba Group
Hangzhou, China
ethan.wf@alibaba-inc.com

Shihao Li
Alibaba Group
Hangzhou, China
shihao.lsh@alibaba-inc.com

Dekun Yang
Alibaba Group
Hangzhou, China
dekun.ydk@alibaba-inc.com

Bufeng Zhang
Alibaba Group
Hangzhou, China
feitong@alibaba-inc.com

ABSTRACT

For an online tourism platform like Fliggy.com [1], search is one of the most critical channels for engaging customers. Recently, neural relevance models for information retrieval are becoming increasingly popular. They provide effective approaches for product search systems due to their competitive advantages in semantic matching. However, these techniques suffer from the data sparseness problem especially in the domain of online tourism. To address this issue, this paper proposes a novel **K**nowledge **E**nhanced **N**eural **R**elevance **M**atching model called KENRM, which incorporates knowledge graph as additional relevance signals to enhance the performance of neural matching model in an end-to-end fashion. The proposed approach can not only help to overcome the long-tail problem of click-through data, but also incorporate external heterogeneous information to improve search results. Extensive experiments on a real-world online tourism dataset demonstrate significant improvement achieved by our proposed approach over multiple strong baselines.

CCS CONCEPTS

• **Information systems** → **Retrieval models and ranking**.

KEYWORDS

Product Search, Neural Relevance Matching, Knowledge Graph

ACM Reference Format:
Feng Wei, Shihao Li, Dekun Yang, and Bufeng Zhang. 2020. KENRM : Knowledge Enhanced Neural Relevance Matching in Online Tourism Industry. In *Proceedings of ACM SIGIR Workshop on eCommerce (SIGIR eCom'20).* ACM, New York, NY, USA, 8 pages.

*Corresponding Author.
[1]Fliggy.com is a Chinese online tourism platform run by Alibaba Group.

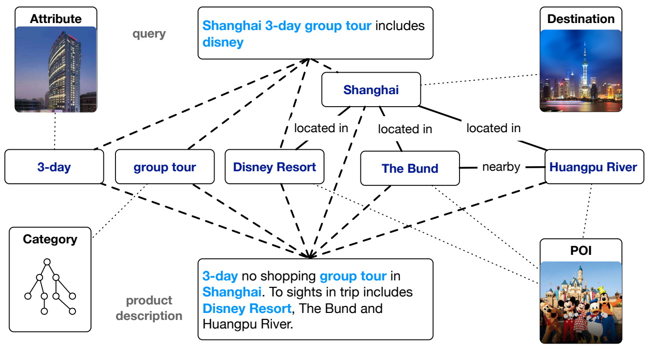

Attribute
query
Shanghai 3-day group tour includes disney
Destination
Shanghai
located in
located in
located in
3-day
group tour
Disney Resort
The Bund
nearby
Huangpu River
Category
POI
product description
3-day no shopping group tour in Shanghai. To sights in trip includes Disney Resort, The Bund and Huangpu River.

Figure 1: Illustration of knowledge graph enhanced search engine systems. The knowledge graph provides fruitful facts and connections between query and product descriptions.

1 INTRODUCTION

Nowadays, online tourism has become increasingly popular, people rely on search engines to find their desired products like booking a hotel or buying tickets in a scenic spot. Similar to the web search, one of the biggest challenges to retrieve relevant products for a query is the lexical gap problem, when users and sellers use different vocabularies to express the same meaning. For example, the Canton Tower is often called "Slim Waist" by the local residents, because the tower resembles the figure of a female body. The lexical gap between queries and product descriptions makes it difficult to exploit such traditional information retrieval models as LSI [3], BM25 [12], which focus more on exact lexical matching in this application scenario.

In recent years, neural relevance models provide a good solution to this problem with their advantages in semantic matching [5]. For example, distributed word and phrase representations such as word2vec [9] and glove [11] provide a promising basis to overcome the longstanding vocabulary mismatch problem in ranking, which refers to the phenomenon where queries and documents describe the same concept with different words. DRMM [4] points out that relevance matching, which is the core problem in information retrieval, has different characteristics from the semantic matching problem that many NLP models are designed for, which

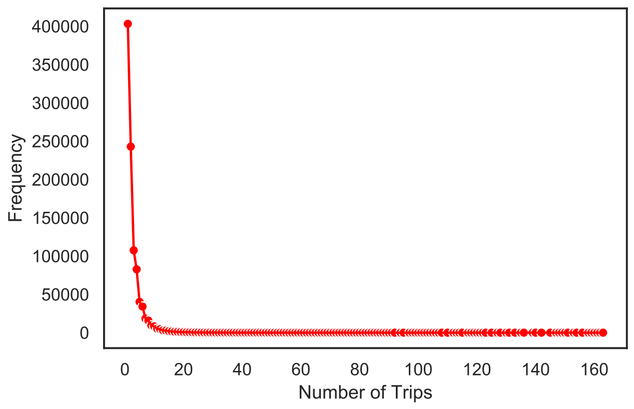


Figure 2: The frequency distribution of travel in the search logs from Fliggy.com. Most people only travel once or twice a year.

is essentially to model how semantically close two pieces of texts are, such as paraphrase detection [14] and question answer [15].

However, training a state-of-the-art deep neural network model usually requires a large amount of labeled data which is not always readily available. As shown in Figure 2, travel is a low frequency activity for most people, which causes the data sparseness problem in the domain of online tourism. And also it is infeasible to obtain enough training data for the long-tailed queries and products. Analysis of the search logs in Fliggy.com, in additional to the click through data, we found that queries and product descriptions often share some common entities like destinations, point of interest (POI) and category and so on, which can be regarded as the additional connections between queries and products. For example, as shown in Figure 1, we utilize all types of entities and relations to construct a knowledge graph (KG), which can be incorporated to alleviate the problem of sparsity to a certain extent.

Inspired by the recent advances in graph embedding techniques [20], this paper introduces a novel relevance model named KENRM (**K**nowledge **E**nhanced **N**eural **R**elevance **M**atching model), which integrates a hybrid structure of knowledge graph into a neural relevance matching model. We firstly combine query-product bipartite graph with entity-based knowledge graph to build a hybrid structure of knowledge graph using the search logs in Fliggy.com. Then we design an entity embedding propagation layer to aggregate multi-hop neighbors representations. Meanwhile, we process the textual information in queries and product descriptions with multi-head self-attention layer in the manner of interaction-based model. Moreover, we fuse the heterogeneous information with a Tensor Fusion Layer to combine textual representation and KG representation. Finally, we evaluate our model on a real-world dataset of online tourism search, experimental results demonstrate that our model substantially outperforms state-of-the-art baselines.

To conclude, the major contributions of this work are as follows:

- We design a unified neural framework to incorporate a hybrid structure of knowledge graph to an interaction-based neural retrieval model in an end to end fashion.
- Our proposed model can effectively deal with the long tail problem and incorporate external heterogeneous information to improve search results.
- We conduct experiments on our online tourism dataset, and the results demonstrate our proposed model can achieve significant improvement in retrieval performance compared with the existing work.

The following of the paper is organized as follows. Firstly, we first discuss related work in Section 2. Then we introduce the preliminaries in Section 3. In Section 4, we present our proposed model. Section 5 describes the experimental methodology and evaluation. Finally we conclude and discuss future work in Section 6.

2 RELATED WORK

Deep learning has achieved great success in many NLP and information retrieval applications. Early attempts at neural information retrieval mainly focus on representation-based modeling between query and document such as DSSM [6] and CDSSM [13]. DSSM is an early model for web search that directly maps word sequences to character-level trigrams using a word hashing layer, and then feeds the dense hashed features to a multi-layer perceptron for similarity learning representations. CDSSM extends this idea by replacing the multi-layer perceptron in DSSM with a CNN to capture local contextual signals from neighboring character trigrams. Recently, interaction-based approaches have demonstrated increased effectiveness in many ranking tasks, which learn word-level interaction patterns from query-document pairs. DRMM [4] uses histogram to summarize the word-level similarities into ranking models. K-NRM [18] and Conv-KNRM [2] use kernels to summarize word-level interactions with word embeddings and provide soft match signals for learning to rank. Interaction-based models and representation-based models can also be combined for further improvements like DUET [10]. In this paper, we process the textual information in queries and products descriptions in the manner of interaction-based model.

Knowledge graph embedding aims at embedding entities and relations of a knowledge graph into low-dimensional continuous vector space in which the inherent structure of the knowledge graph is preserved. Recently, due to the superior performance, translation based embedding methods have received great attention, in which entities are usually represented as vectors and relations are typically abstracted as operations on the entity vectors. As the most representative translation based method, TransE [1] represents both entities and relations as vectors in the same space so that the embedded entities h and t can be connected by relation r when triple (h, r, t) holds, i.e., $h + r \approx t$. However, TransE has difficulty in dealing with 1-to-N relations. To this end, TransH [17] allows an entity to have distinct representations when involved in different relations by introducing relation-specific hyperplanes. Moreover, TransR [8] first projects entities from entity space to corresponding relation space and then builds translations between projected entities. In this paper, to facilitate appropriate characteristics of an entity when involved in different relations, we apply the loss function of TransR to our final loss and optimize our proposed model in a co-training method.

Our work mainly focuses on exploring the effectiveness of entities representation and relations representation in a knowledge graph to enhance the performance of neural information retrieval models.

3 PRELIMINARY

This section first introduces the preliminaries and notations used in this paper, and then describes the hybrid structure of knowledge graph we constructed.

3.1 Notations

Like other neural retrieval models, we use click-through data to train our models. The click-through data used in our study consists of queries and products with their co-clicked information. We denote the set of queries as $\mathcal{Q}$ and each query is represented by a collection of terms $q = \{t_q^1, \cdots, t_q^m\}$. Similarly, the set of products is denoted by $\mathcal{P}$ and each product is represented by a collection of terms $p = \{t_p^1, \cdots, t_p^n\}$.

3.2 Knowledge Graph Construction

In addition to the textual information from query q and product p, we construct a hybrid structure of knowledge graph, which is incorporated into our proposed model as additional relevance signals. The details of construction are shown as follows.

Query-Product Bipartite Graph. In a search scenario, we typically have historical query-product interactions from the click-through data. Here we represent each interaction data as a query-product bipartite graph $\mathcal{G}_B$, which is defined as $\{(q, B_{qp}, p) \mid q \in \mathcal{Q}, p \in \mathcal{P}\}$, where a link $B_{qp} = 1$ indicates that there is an observed interaction between query q and product p, otherwise $B_{qp} = 0$.

Entity-Based Knowledge Graph. In addition to the query-product bipartite graph, there are some external knowledge can be linked to query and product descriptions. In this work, we build a standard knowledge graph $\mathcal{G}_K$ with entities and relations by harvesting the search logs in Fliggy.com, which is presented as $\{(h, r, t) \mid h, t \in \mathcal{E}, r \in \mathcal{R}\}$, where $\mathcal{E}$ denotes the set of entities, $\mathcal{R}$ denotes the set of relations and each triplet in $\mathcal{G}_K$ describes that there is a relationship r from head entity h to tail entity t. Specifically, the set of entities $\mathcal{E}$ is derived from our internal knowledge base in the production system of Fliggy.com. We employ an entity linking system to extract entities from queries and product descriptions. There are four key types of entities in the domain of tourism, as follows.

- **Destinations** are the basic elements shared by all products in our online tourism platform. Meanwhile, queries are often expressed by a combination of a destination name and the keyword "tourism" such as "Beijing tourism".
- **Point of Interest**s, or POIs, are serval specific point locations that someone may find useful or interesting. An example is a point on the Earth representing the location of West Lake, which is a famous scenic spot in Hangzhou.
- **Hotel Brands** are often found in queries, which can be used to represent a preference for a particular type of hotel like Marriott. Moreover, the hotel brand is a basic attribute of hotels.
- **Category Taxonomy Tree** is built by our platform. Products delivered by sellers are classified into a suitable leaf category. The similarity between products with same category is much higher than that of different categories.

Then we regard these four key types of entities as relations to connect queries and products, and obtain the entity-based knowledge graph $\mathcal{G}_K$.

Hybrid Knowledge Graph. Here we define our final knowledge graph, which encodes query-product bipartite graph $\mathcal{G}_B$ and entity-based knowledge graph $\mathcal{G}_K$ as a unified relational graph. We first represent each query-product interaction in $\mathcal{G}_B$ as a triplet $(q, Interact, p)$, where $B_{qp} = 1$ is represented as an additional relation $Interact$ between query q and product p. Then based on the entity-based knowledge graph, $\mathcal{G}_B$ can be seamlessly integrated with $\mathcal{G}_K$ as a unified graph $\mathcal{G} = \{(h, r, t) \mid h, t \in \hat{\mathcal{E}}, r \in \hat{\mathcal{R}}\}$, where $\hat{\mathcal{E}} = \mathcal{E} \cup \mathcal{Q} \cup \mathcal{P}$ and $\hat{\mathcal{R}} = \mathcal{R} \cup \{Interact\}$. Finally, We integrate this knowledge graph $\mathcal{G}$ into neural retrieval models in the next section.

4 OUR PROPOSED MODEL

Our goal is to combine the advantages of both graph embedding methods and neural retrieval approaches. The Framework of our proposed model is illustrated in Figure 3. Our model takes a query q, a product p and their corresponding sub-graph as input, and outputs the relevance score between a query and product.

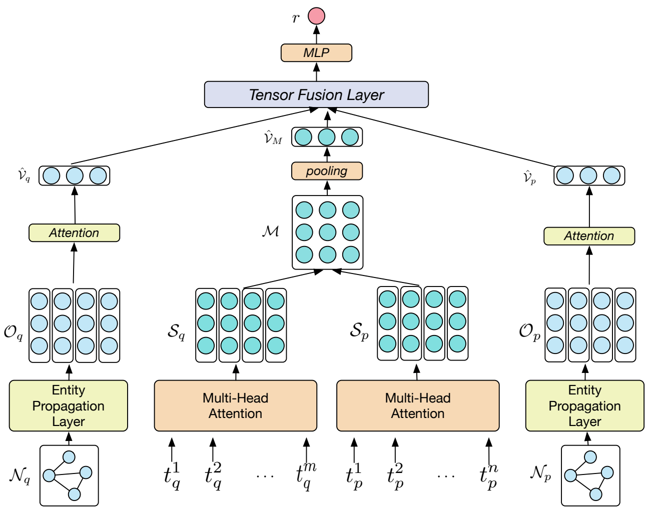


Figure 3: Illustration of the proposed KENRM model.

4.1 Entity Propagation

For the input corresponding sub-graph, we treat query q (or product p) as the seed in the knowledge graph $\mathcal{G}$, and extend along the links to form multi-hop neighbors N_q^k, $k = (1, 2, \cdots, H)$. The hop neighbors set N_q^k is the set of knowledge triples that are k-hop away from the seed q. These multi-hop sets are used to interact with the embedding of entity q iteratively. To be more specific, the set of k-hop relevant entities for query q is defined as:

$$\mathcal{E}_q^k = \{t \mid (h, r, t) \in \mathcal{G} \text{ and } h \in \mathcal{E}_q^{k-1}\}, \tag{1}$$

where $\mathcal{E}_q^0 = \{q\}$.

The k-hop neighbors set of query q is defined as the set of knowledge triples in Equation (2):

$$\mathcal{N}_q^k = \{(h, r, t) \mid (h, r, t) \in \mathcal{G} \text{ and } h \in \mathcal{E}_q^{k-1}\}. \tag{2}$$

In practice, we sample a fixed-size set of neighbors instead of using a full set to reduce the computation overhead.

4.2 Knowledge Graph Embedding

Knowledge graph embedding is an effective way to parameterize entities and relations as vector representations, while preserving the graph structure. Specifically, each input q is associated with an entity embedding $\mathcal{V}_q \in \mathcal{R}^d$, where d is the dimension of entity embeddings. Given the entity embedding $\mathcal{V}_q$ and the k-hop neighbors set $\mathcal{N}_q^k$ of query q, each triple (h_i, r_i, t_i) in $\mathcal{N}_q^k$ is assigned a relevance probability by comparing query q to the head h_i and the relation r_i in this triple:

$$\alpha_i = \frac{exp(\mathcal{V}_q^T \mathcal{R}_i \mathcal{V}_{h_i})}{\sum_{(h_j, r_j, t_j) \in \mathcal{N}_q^k} exp(\mathcal{V}_q^T \mathcal{R}_j \mathcal{V}_{h_j})}, \tag{3}$$

where $R_i \in \mathcal{R}^{d \times d}$ and $\mathcal{V}_{h_i} \in \mathcal{R}^d$ are the embeddings of relation r_i and head h_i, respectively. The relevance probability α_i can be regarded as the similarity of query q and the entity h_i measured in the space of relation R_i. After obtaining the relevance probabilities, we take the sum of tails in $\mathcal{N}_q^k$ weighted by the corresponding relevance probabilities, and the vector O_q^k is returned:

$$O_q^k = \sum_{(h_i, r_i, t_i) \in \mathcal{N}_q^k} \alpha_i \times \mathcal{V}_{t_i}, \tag{4}$$

where $\mathcal{V}_{t_i} \in \mathcal{R}^d$ is the embedding of tail t_i. The vector O_q^k can be seen as the k-order attention with respect to query q. The embedding of query q is calculated by combining all hop representations, which is formulated as follows:

$$\hat{\mathcal{V}}_q = \mathcal{V}_q + \sum_{k=0}^{H} O_q^k. \tag{5}$$

The same process is applied to the product p and then $\hat{\mathcal{V}}_p$ can be obtained.

4.3 Text Embedding

Given a query q and a product p, the textual matching component regards the query terms $q = \{t_q^1, \cdots, t_q^m\}$ and the product terms $p = \{t_p^1, \cdots, t_p^n\}$ as input. Specifically, We first employ an embedding layer to convert each term into a l-dimensional vector representation, generating a matrix representation for the query q and product q, where $\mathcal{S}_q \in \mathcal{R}^{m \times l}$ and $\mathcal{S}_p \in \mathcal{R}^{n \times l}$. In the following, we introduce our representation learning method with multi-head self-attentive neural network [16] in Equation (6):

$$\begin{aligned} \hat{\mathcal{S}}_q &= Multi - Head(\mathcal{S}_q) \\ \hat{\mathcal{S}}_p &= Multi - Head(\mathcal{S}_p), \end{aligned} \tag{6}$$

where $\hat{\mathcal{S}}_q \in \mathcal{R}^{m \times l}$ and $\hat{\mathcal{S}}_p \in \mathcal{R}^{n \times l}$. Then, to measure the similarity between the query and the descriptions of product, we match the query with the product by taking the dot product between the query representation matrix $\hat{\mathcal{S}}_q$ and the product representation matrix $\hat{\mathcal{S}}_p$:

$$\mathcal{M} = \hat{\mathcal{S}}_q \hat{\mathcal{S}}_p^T, \tag{7}$$

where $\mathcal{M} \in \mathcal{R}^{m \times n}$, and $\mathcal{M}_{i,j}$ can be considered as the similarity score by matching the query phrase vector $\mathcal{S}_{q,i}$ with the product phrase vector $\mathcal{S}_{p,j}$. The similarity matching matrix $\mathcal{M}$ is further normalized with range [0, 1] through a softmax function:

$$\mathcal{M}_{i,j} = softmax(\mathcal{M}_{i,j}) = \frac{e^{\mathcal{M}_{i,j}}}{\sum_{k=1}^{n} e^{\mathcal{M}_{i,k}}}. \tag{8}$$

Then we perform two pooling steps to generalize exact matches and soft matches in the similarity matrix $\mathcal{M}$ to obtain discriminative feature vectors. The first step is max pooling along query dimension:

$$\mathcal{V}_M = [max(\mathcal{M}_{1,:}), max(\mathcal{M}_{2,:}), \cdots, max(\mathcal{M}_{m,:})], \tag{9}$$

where $\mathcal{V}_M \in \mathcal{R}^m$. The second step is bin pooling to count the matches at different strength [4]:

$$\hat{\mathcal{V}}_M^k = log \sum_i \mathcal{I}(sd_k \le \mathcal{V}_M^i \le ed_k), \tag{10}$$

where $\mathcal{I}$ is the indicator function. sd_k and ed_k is the range for k bin. $\hat{\mathcal{V}}_M^k$ is the number of product terms whose scores fall into this bin.

The max-pooling matches each query term to its closest product term using embeddings, which is the exact match if one exists. Its score describes how closely related a product term is to the query. The bin-pooling counts the number of query term with different connection strength to the product. The bin with range [1, 1] counts the exact matches, and the other bins generate soft match signals. The two pooling steps together summarize the term matches to query-product ranking evidence.

4.4 Fusing Heterogeneous Information

We build a fusion layer called Tensor Fusion Layer [19] in our proposed model for further feature fusion, which is defined as the following vector field using a 3-fold Cartesian product:

$$\{(\hat{v}_q, \hat{v}_p, \hat{v}_M) \mid \hat{v}_q \in \begin{bmatrix} \hat{\mathcal{V}}_q \\ 1 \end{bmatrix}, \hat{v}_p \in \begin{bmatrix} \hat{\mathcal{V}}_p \\ 1 \end{bmatrix}, \hat{v}_M \in \begin{bmatrix} \hat{\mathcal{V}}_M \\ 1 \end{bmatrix}\}. \tag{11}$$

Each neural coordinate $(\hat{v}_q, \hat{v}_p, \hat{v}_M)$ can be seen as a 3-D point in the 3-fold Cartesian space. This Equation (11) is mathematically equivalent to a differentiable outer product between the query entity representation $\hat{\mathcal{V}}_q$, the product entity representation $\hat{\mathcal{V}}_p$, and the text matching representation $\hat{\mathcal{V}}_M$:

$$\mathcal{Z} = \begin{bmatrix} \hat{\mathcal{V}}_q \\ 1 \end{bmatrix} \otimes \begin{bmatrix} \hat{\mathcal{V}}_p \\ 1 \end{bmatrix} \otimes \begin{bmatrix} \hat{\mathcal{V}}_M \\ 1 \end{bmatrix}, \tag{12}$$

where $\otimes$ indicates the outer product between vectors and $\mathcal{Z} \in \mathcal{R}^{d \times d \times l}$. Then we employ a multi-layer perceptron to obtain the relevance score between the input query q and product p:

$$r = MLP(\mathcal{Z}). \tag{13}$$

4.5 Model Training

In the training stage, we use a pairwise ranking loss to train our neural retrieval model. Given a search query q along with a positive product p^+ and some negative products P^-, the ranking loss is defined as follows:

$$\mathcal{L}_{rank} = \sum_{q} \sum_{p^- \in \mathcal{P}^-} max(0, 1 - r^+ + r^-). \tag{14}$$

At the same time, for each observed triple $(h, r, t) \in \mathcal{G}$, we follow the existing work called transR [8] and define the score function as:

$$f(h, r, t) = \|Rh + r - Rt\|^2, \tag{15}$$

the training of TransR considers the relative order between the valid triplets and broken ones, and encourages their discrimination through a pairwise ranking loss:

$$\mathcal{L}_{KG} = \sum -ln\ \delta(f(h, r, t') - f(h, r, t)), \tag{16}$$

where $(h, r, t') \notin \mathcal{G}$ is a new triplet by randomly replacing tail entity.

Finally, we have the objective function to learn Equations (14) and (16) jointly, which is formulated as follows:

$$\mathcal{L}_{KENRM} = \mathcal{L}_{rank} + \beta \cdot \mathcal{L}_{KG}, \tag{17}$$

where β is a tunable parameter. In our experiment, we just empirically set $\beta = 0.1$.

We apply Adam [7] method with mini-batches to optimize the loss function given in Equation (17). Meanwhile, the early stopping strategy is used to prevent over-fitting.

5 EXPERIMENT

This section describes the dataset, evaluation metrics, baselines, and implementation details of our experiments. Then, we evaluate our proposed method aiming to answer the following research questions:

- **RQ1** : How does our proposed model compare to the state-of-the-art neural retrieval models?
- **RQ2** : How does each module in our proposed model contribute to the final results?
- **RQ3** : How is our proposed approach dealing with the long-tail problem of click-through data?

5.1 Dataset And Metrics

To evaluate our proposed model, we collected a large query log dataset during a three month window from October 2019 to December 2019 on Fliggy.com. The dataset is composed of more than 7.5 million query and product pairs, of which there are more than 1.7 million unique queries and more than 1 million unique products. We randomly split the data into training and validation dataset with the ratio $99\% : 1\%$, and we make sure there are no shared query and product pairs in the training and validation dataset. All queries and products' descriptions are segmented by Alibaba Word Segmenter.

Our knowledge graph consists of query-product bipartite graph and entity-based knowledge graph. Besides the query-product bipartite graph, we need to construct a query and product entity-based knowledge graph for our dataset. We map queries and products into entities in our production systems via text matching if there is a mapping available. The details of our dataset are listed in Table 1.

Table 1: Dataset statistics.

Statistics	Value
# queries	1,785,091
# products	1,037,237
# vocabularies	92,100
# entities	2,848,384
# relations	5
# triplets	3,916,290

In our evaluation, we adopt Mean Average Precision (**MAP**) and Normalized Discounted Cumulative Gain (**NDCG@**k) as our evaluation metrics. **MAP** measures the mean of the average precision scores for each query in a set of queries. **NDCG@**k is used to capture multiple relevance levels. We regard the products returned by the our search engine in the SRP (**S**earch **R**esults **P**age) for each test query as candidates, and evaluate ranking performance of different models on those candidates. The products shown in the SRP for each test query are automatically labeled based on user-specific actions by three grades of relevance, irrelevant (0), somewhat relevant (1), very relevant (2), which correspond to not clicked, clicked, and clicked and purchased products, respectively.

5.2 Baselines

We compare our proposed KENRM model to a number of neural ranking models designed for standard ad hoc retrieval tasks. All comparison methods can be divided into three sets: representation-based models (DSSM, CDSSM), interaction-based models (DRMM, K-NRM, Conv-KNRM) and a combined model (DUET). The details of all baseline methods are shown as follows.

- **DSSM** [6] is an early neural model that directly maps word sequences to character-level trigrams using a word hashing layer, and then feeds the dense hashed features to a multi-layer perceptron for similarity learning representations.
- **CDSSM** [13] replaces the multi-layer perceptron in DSSM with a CNN layer to capture local contextual signals from neighboring character trigrams.
- **DRMM** [4] uses histogram to summarize the word-level similarities into ranking models.
- **K-NRM** [18] uses kernels to summarize word-level interactions with word embeddings and provide soft match signals for learning to rank.
- **Conv-KNRM** [2] uses convolutional neural networks to represent ngrams of various lengths and soft-matches them in a unified embedding space.
- **DUET** [10] combines interaction based method and representation based method with a global component for semantic matches and a local component for exact matches.

To enable fair comparisons with the baselines, we adopt the same training strategies in all our experiments, including embeddings, optimizer, and hyper parameter settings. We train word2vec [9] embedding with a learning rate of 0.025 and the SGD optimizer in our training corpus of Fliggy.com, which is used to initialize the word vectors embedding layer. The number of self-attention head is set to 2 in our proposed model. Term matching scores are binned into 5 bins: [0, 0.25), [0.25, 0.5), [0.5, 0.75), [0.75, 1), [1, 1], we discard the negative bins as negative cosine similarities.

5.3 Retrieval Performance

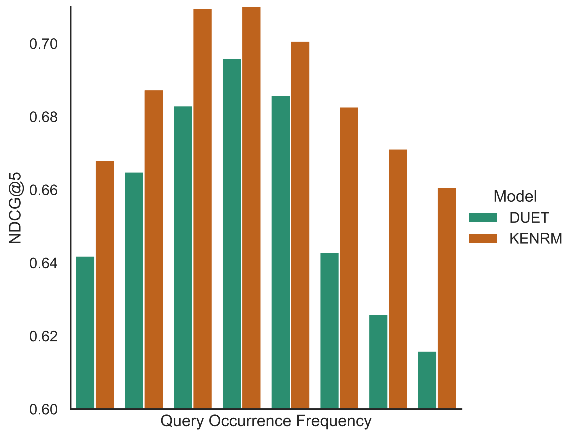
To answer **RQ1**, we conduct our main experiments and list the results of baselines and our proposed model in Table 2. We can see that the performance of representation-based models (DSSM, CDSSM) are consistently worse than the interaction-based models (DRMM, K-NRM, Conv-KNRM) in our dataset. In particular, CDSSM suffers more than DSSM, showing that a more complex model may lead to lower effectiveness. The best neural baseline is DUET, which demonstrates that combining two types of neural models can improve ranking performance.

In comparison, our proposed KENRM model achieves high effectiveness on all metrics, beating all neural baselines by a large margin. We believe that effectiveness gains mainly come from two aspects: 1) Our proposed model incorporates graph information to enhance the representations of query and product, which provides additional relevance signals in this retrieval task; 2) Tensor Fusion Layer provides a better fusion of graph embedding and text embedding.

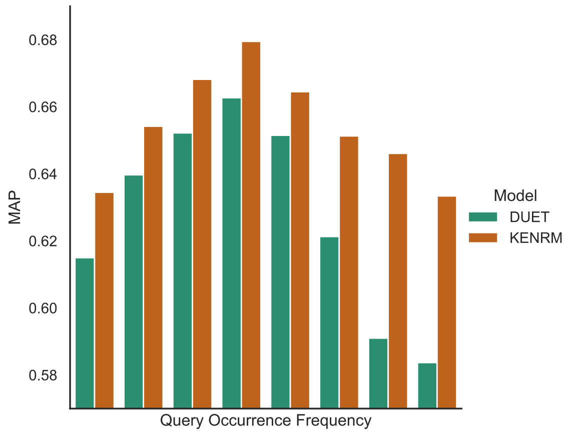
5.4 Ablation Study

To better understand the contribution of each module in our proposed KENRM model and answer **RQ2**, we conduct an ablation study by removing each component step by step. Here, we aim to study how the sub-graph of query and product, multi-head attention, max-bin pooling and tensor fusion modules contribute to our model effectiveness. The results are shown in Table 3 with each row denoting the removal of a specific module. For example, the row "– query graph" represents removing query sub-graph module and the row "– tensor fusion layer" means replacing tensor fusion layer with concatenation layer.

From the first three rows "– query/product/both graph(s)", we can see that removing query and product sub-graphs from knowledge graph leads to a significant effectiveness drop, which means query and product sub-graphs provide additional relevance signals to enhance the performance of neural retrieval model. Also, removing tensor fusion layer makes the results consistently and significantly worse, which confirms that this module can obtain a better fusion of heterogeneous information. Turning our attention to the last two rows, we observe that removing multi-head self attention layer or max-bin pooling layer both lead to significant drops in our datasets. This suggests that multi-head self attention layer is more suitable to process bag-of-word textual information, and max-bin pooling layer can combine exact match signals and soft match signals to provide a better performance.



(a) NDCG@5



(b) MAP

Figure 4: Retrieval performance regarding the long-tail problem in our dataset.

5.5 Long Tail Problem

To answer **RQ3**, we divide search queries into 8 groups according to their occurrence frequency ranks in our dataset and plot the averaged relative performance gains of our proposed KENRM model in each group with the best neural baseline DUET in Figure 4. All of their relative gains increase in the first 4 groups because the commercial search engine performs relatively better on popular queries. In the last 4 groups where increasingly fewer training samples are provided, the performance of DUET starts to decline dramatically, while our proposed model still remains robust. This suggests that neural retrieval models indeed suffer from overfitting problems on those rare queries in the training data and our proposed graph embedding-based approach provides an effective solution.

6 CONCLUSION

In this work, we propose a novel Knowledge Enhanced Neural Relevance Matching model (KENRM) for product search in the

Table 2: Main results on our dataset. The best results on each metric are bold.

Model	NDCG@1	NDCG@5	NDCG@10	NDCG@20	MAP
DSSM	0.3589	0.5925	0.6338	0.6525	0.5466
CDSSM	0.3307	0.5724	0.6167	0.6363	0.5255
DRMM	0.4502	0.6380	0.6768	0.6959	0.6041
K-NRM	0.4250	0.6269	0.6648	0.6840	0.5882
Conv-KNRM	0.4409	0.6418	0.6796	0.6961	0.6035
DUET	0.4805	0.6667	0.7012	0.7174	0.6313
KENRM	**0.5188**	**0.7002**	**0.7302**	**0.7431**	**0.6642**

Table 3: Ablation Study.

Model	NDCG@1	NDCG@5	NDCG@10	NDCG@20	MAP
KENRM	**0.5188**	**0.7002**	**0.7302**	**0.7431**	**0.6642**
- query graph	0.4502	0.6385	0.6768	0.6959	0.6041
- product graph	0.4435	0.6362	0.6740	0.6927	0.5999
- both graphs	0.4292	0.6307	0.6687	0.6872	0.5925
- tensor fusion layer	0.4537	0.6466	0.6834	0.7009	0.6103
- multi-head attention	0.5003	0.6894	0.7203	0.7335	0.6515
- max-bin pooling	0.4781	0.6751	0.7076	0.7217	0.6361

domain of online tourism, which integrates a hybrid structure of knowledge graph into a neural ranking model in an end to end fashion. We design an entity embedding propagation layer, which adaptively aggregates the embedding of multi-hop neighbors to update representation of query or product. Through extensive experiments on a real-world dataset of online tourism search, we demonstrate that our proposed approach can successfully combine both the advantages of graph embedding models and neural retrieval models. To be more specific, experimental results show that KENRM enables neural models to effectively deal with the long-tail sparsity problem and combine heterogeneous external information to improve search accuracy. We also conduct an ablation study to verify the importance of each model component.

This work explores the potential of incorporating a knowledge graph to enhance the performance of neural retrieval models in the search scenario. In the future, we will focus on the knowledge graph to better understand the query, user and products.

REFERENCES

[1] 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.

[2] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In *Proceedings of the eleventh ACM international conference on web search and data mining.* 126–134.

[3] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science* 41, 6 (1990), 391–407.

[4] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management.* 55–64.

[5] Jiafeng Guo, Yixing Fan, Liang Pang, Liu Yang, Qingyao Ai, Hamed Zamani, Chen Wu, W Bruce Croft, and Xueqi Cheng. 2019. A deep look into neural ranking models for information retrieval. *Information Processing & Management* (2019), 102067.

[6] 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 *Proceedings of the 22nd ACM international conference on Information & Knowledge Management.* 2333–2338.

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

[8] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence.*

[9] 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.

[10] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. 2017. Learning to match using local and distributed representations of text for web search. In *Proceedings of the 26th International Conference on World Wide Web.* 1291–1299.

[11] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP).* 1532–1543.

[12] Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2009), 333–389.

[13] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web.* 373–374.

[14] Richard Socher, Eric H Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Y Ng. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in neural information processing systems.* 801–809.

[15] Ming Tan, Cicero Dos Santos, Bing Xiang, and Bowen Zhou. 2016. Improved representation learning for question answer matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* 464–473.

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems.* 5998–6008.

[17] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Twenty-Eighth AAAI conference on artificial intelligence.*

[18] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR conference on research and development in information retrieval.* 55–64.

[19] Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2017. Tensor fusion network for multimodal sentiment analysis. *arXiv preprint arXiv:1707.07250* (2017).

[20] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and Maosong Sun. 2018. Graph Neural Networks: A Review of Methods and Applications. *arXiv preprint arXiv:1812.08434* (2018).