Graph Enhanced BERT for Query Understanding

Juanhui Li
Michigan State University, USA
lijuanh1@msu.edu

Wei Zeng
Baidu Inc., China
zengwei04@baidu.com

Suqi Cheng
Baidu Inc., China
chengsuqi@baidu.com

Yao Ma
New Jersey Institute of Technology
USA
yao.ma@njit.edu

Jiliang Tang
Michigan State University, USA
tangjili@msu.edu

Shuaiqiang Wang
Baidu Inc., China
shqiang.wang@gmail.com

Dawei Yin
Baidu Inc., China
yindawei@acm.org

ABSTRACT

Query understanding plays a key role in exploring users' search intents. However, it is inherently challenging since it needs to capture semantic information from short and ambiguous queries and often requires massive task-specific labeled data. In recent years, pre-trained language models (PLMs) have advanced various natural language processing tasks because they can extract general semantic information from large-scale corpora. However, directly applying them to query understanding is sub-optimal because existing strategies rarely consider to boost the search performance. On the other hand, search logs contain user clicks between queries and urls that provide rich users' search behavioral information on queries beyond their content. Therefore, in this paper, we aim to fill this gap by exploring search logs. In particular, we propose a novel graph-enhanced pre-training framework, GE-BERT, which leverages both query content and the query graph to capture both semantic information and users' search behavioral information of queries. Extensive experiments on offline and online tasks have demonstrated the effectiveness of the proposed framework.

CCS CONCEPTS

• **Information systems** → **Information retrieval query processing**; **Query representation**; **Retrieval models and ranking**.

KEYWORDS

Query understanding, BERT, Graph neural networks, KL-divergence

1 INTRODUCTION

Query understanding [15] plays a crucial role in information retrieval. It aims to learn the intentions of a search query, and provides useful information to advance downstream applications such as document retrieval and ranking [6]. Many types of tasks have been developed to facilitate query understanding such as query classification [2, 14], query matching [8], and query expansion [21]. The challenges for these tasks are mainly from two aspects. First, search queries are often short and ambiguous; as a result, it is hard to capture the semantic meanings of queries only relying on their content. Second, the majority of existing solutions [1, 2] often train models from scratch in a supervised way, which require a large number of task-specific labeled data. However, obtaining such labeled data is labour intensive and usually costs tremendous money.

Recent years we have witnessed immense efforts in developing pre-trained language models (PLMs) [5, 13] such as BERT [4] and its variants [11, 16, 19]. These PLMs are trained on large-scale unlabeled corpora where they can extract general semantic information. Thus, they pave a way to help specific NLP tasks even when only a small amount of task-specific labeled data is available. However, directly applying these PLMs to query understanding can be suboptimal. First, the goal of query understanding is to boost the search performance but the current PLMs seldom incorporate this goal into their pre-training strategies. Second, queries are different from normal natural languages because they are usually short.

To enhance the PLMs towards query understanding, one natural direction is to design domain-adaptive pre-training strategies with domain data [3, 10, 12, 21]. The search log is a commonly used domain data for query understanding, which is often denoted as a query-url bipartite click graph [6]. The graph encodes user search behaviors and provides rich information about queries beyond their content. Thus, it is appealing to explore this click graph to advance PLMs. However, BERT and its variants cannot directly process such graph data and dedicated efforts are desired.

In this paper, we propose a novel graph-based domain adaptive pre-training framework, named Graph Enhanced BERT(GE-BERT), which enjoys both advantages of the PLMs and the domain knowledge, i.e. click graph, for query understanding. To evaluate
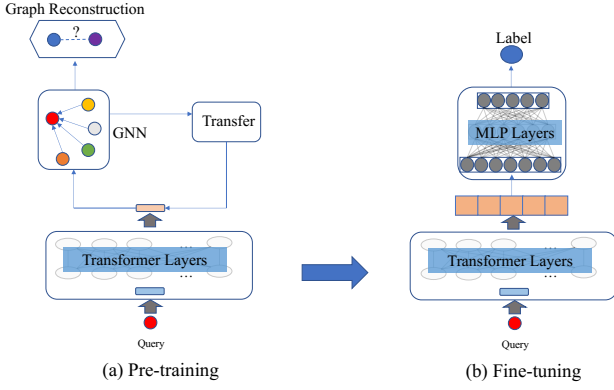
Figure 1: Illustration of pre-training and fine-tuning process.

its effectiveness, we first fine-tune GE-BERT and compare it with representative baselines on two offline tasks, i.e., query classification and query clustering. Then we demonstrate the superiority of the fine-tuned GE-BERT on the online deployment based on the the medical query searching.

2 THE PROPOSED FRAMEWORK

In this section, we present the details of the proposed framework. Before that, we firstly introduce some key concepts and notations. The query graph is built from the query-url bipartite graph [6], where two queries are connected if they share at least one url in the bipartite graph. Formally, we denote the query graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, ..., v_n\}$ is the set of $|\mathcal{V}| = n$ query nodes, and $\mathcal{E}$ is the set of edges connecting nodes among $\mathcal{V}$. Furthermore, the query graph can also be described by an adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, where $\mathbf{A}_{ij} = 1$ if the nodes v_i and v_j are connected and $\mathbf{A}_{ij} = 0$ otherwise. Each query node v_i in the graph represents a query $q_i = \{c_1^i, ..., c_t^i\}$, which consists of a sequence of tokens. Here c_j^i indicates the j-th token in q_i, and t is the number of tokens.

Based on the query graph, we propose a pre-training framework GE-BERT to integrate the graph information into BERT, which can be further fine-tuned for downstream query understanding tasks. An illustration is shown in Figure 1. In particular, the pre-training framework in Figure 1(a) consists of three modules – the Transformer module, the GNN module and the Transfer module. The *graph semantics* captured from the GNN module is transferred through the Transfer module to the Transformer module. An illustration of the downstream task is presented in Figure 1(b).

Notably, we inject the query graph information into the Transformer module to perform downstream tasks, instead of using both the Transformer module and the GNN module to build the downstream model due to two reasons: 1) First, the GNN module is only trained on the existing query graph and thus it cannot handle new coming queries without any connection to the existing graph. The Transformer module, on the other hand, has the flexibility to generate embeddings for new queries. 2) Second, the complexity of the GNN module for fine-tuning and online deployment is significantly higher than that of the Transformer module, while the Transformer module provides more practical flexibility. Next, we introduce details of the three modules in the proposed pre-training framework and discuss pre-training strategies.

2.1 The Transformer Module

In the Transformer module, we adopt the Transformer encoder of the BERT model to learn representations of queries. The encoder is pre-trained on a large scale corpus and thus has strong ability to capture semantic information in natural language. The Transformer encoder takes a sequence of tokens as input and outputs a sequence of their corresponding final embeddings, i.e., one embedding for each token. To feed the queries into the Transformer encoder, we tokenize each query into a sequence of tokens based on the Word-Piece method [18]. Following the original BERT [4], for each query, we add a special token [CLS] at the beginning of the token sequence; thus the query q_i is denoted as: $q_i = \{[CLS], c_1^i, ..., c_t^i\}$. We specifically denote the input embedding and output embedding for a token c_j^i in query q_i as $\mathbf{E}_{c_j^i}$ and $\mathbf{E}'_{c_j^i}$, respectively. Formally, we summarize the aforementioned process into the following formulation:

$$\{\mathbf{E}'_{CLS^i}, \mathbf{E}'_{c_1^i}, ..., \mathbf{E}'_{c_t^i}\} = \text{Transformer}(\{\mathbf{E}_{CLS^i}, \mathbf{E}_{c_1^i}, ..., \mathbf{E}_{c_t^i}\}). \quad (1)$$

Finally, we use the output embedding of the [CLS] token $\mathbf{E}'_{CLS^i}$ as the final embedding of the entire query. For convenience, we denote the final embedding output from the Transformer module for query q_i as $\mathbf{H}_i = \mathbf{E}'_{CLS^i}$.

2.2 The GNN Module

To capture the graph information in the query graph, we propose the GNN module which takes the query graph as input. The node's initial embedding is represented by the output embedding from the Transformer module for the specific query. The GNN module consists of several GNN layers, where each layer updates the node embedding by aggregating information from its neighborhood. Specifically, the GNN model is able to capture the K-hop neighborhood information after K layers. Formally, the k-th GNN layer can be generally defined as follows:

$$\tilde{\mathbf{H}}^k = \text{GNN}(\mathbf{A}, \tilde{\mathbf{H}}^{k-1}) \quad (2)$$

where $\text{GNN}(\cdot)$ denotes a general GNN layer, $\mathbf{A}$ is the adjacency matrix and $\tilde{\mathbf{H}}^k \in \mathbb{R}^{n \times d_k}$ is the output embedding of all nodes in the k-th layer. We use $\tilde{\mathbf{H}}_i \in \mathbb{R}^{d_K}$ to denote the final output embedding of the node v_i after a total of K iterations, i.e., $\tilde{\mathbf{H}}_i = \tilde{\mathbf{H}}_i^K$. Specifically, the node embedding is initialized with the output from the Transformer module, i.e., $\tilde{\mathbf{H}}_i^0 = \mathbf{H}_i$.

In this paper, we adopt two representative GNN models, i.e., GCN [7] and GAT [17], to capture the query graph information. To preserve the graph information and train the GNN parameters, we use the output node embeddings from the GNN module to reconstruct the query graph. In particular, given two nodes v_i and v_j, we define a probability score $s_{ij} = \text{sigmoid}(\tilde{\mathbf{H}}_i \cdot \tilde{\mathbf{H}}_j)$ to predict if they have a link in the query graph where $\text{sigmoid}(\cdot)$ is the sigmoid function. The loss of the graph reconstruction is defined as follows:

$$\mathcal{L}_{GNN} = -\sum_i \sum_{j \in (\mathcal{N}_i \cup \mathcal{N}'_i)} \mathbf{A}_{ij} \log s_{ij} + (1 - \mathbf{A}_{ij}) \log(1 - s_{ij}) \quad (3)$$

where $\mathcal{N}_i$ is the set of neighbors of v_i, $\mathcal{N}'_i$ is the set of negative samples that are not connected with node v_i, and $|\mathcal{N}'_i| = |\mathcal{N}_i|$.

2.3 The Transfer Module

The Transfer module is proposed to fuse the graph information captured by the GNN module to the Transformer module. In the literature, minimizing the KL divergence between prior and posterior distributions is demonstrated to be an effective way for knowledge transfer [9, 20]. Inspired by this, we define the prior distribution based on the Transformer module since it only utilizes query content: $p(\mathbf{A}_{ij}|\mathbf{H}_i, \mathbf{H}_j) = sigmoid(\mathbf{H}_i \cdot \mathbf{H}_j)$. The posterior distribution is obtained from the GNN module as it involves additional information from users' search behaviors: $\tilde{p}(\mathbf{A}_{ij}|\tilde{\mathbf{H}}_i, \tilde{\mathbf{H}}_j) = sigmoid(\tilde{\mathbf{H}}_i \cdot \tilde{\mathbf{H}}_j)$. The loss to minimize the distance between these two distributions is defined as follows:

$$
\mathcal{L}_{KL} = \sum_i \sum_{j \in (\mathcal{N}_i \cup \mathcal{N}_i')} \left(\tilde{p}(\mathbf{A}_{ij}|\tilde{\mathbf{H}}_i, \tilde{\mathbf{H}}_j) \log \frac{\tilde{p}(\mathbf{A}_{ij}|\tilde{\mathbf{H}}_i, \tilde{\mathbf{H}}_j)}{p(\mathbf{A}_{ij}|\mathbf{H}_i, \mathbf{H}_j)} \right.
$$
$$
\left. + (1 - \tilde{p}(\mathbf{A}_{ij}|\tilde{\mathbf{H}}_i, \tilde{\mathbf{H}}_j)) \log \frac{1 - \tilde{p}(\mathbf{A}_{ij}|\tilde{\mathbf{H}}_i, \tilde{\mathbf{H}}_j)}{1 - p(\mathbf{A}_{ij}|\mathbf{H}_i, \mathbf{H}_j)} \right)
$$

The prior and posterior distributions essentially predict if two nodes are connected. Optimizing this loss potentially drives the Transformer module to achieve similar graph reconstruction result with the GNN module, i.e., the Transformer module could naturally capture the graph information guided by the GNN module.

2.4 Pre-training Strategies

We propose two strategies for the pre-training process, i.e., stage-by-stage and joint training. 1) In the **stage-by-stage strategy**, we separate the training processes of the Transformer module and the GNN module. In the stage of training the GNN module, we fix the parameters of the Transformer module and update the GNN module by the loss $\mathcal{L}_{GNN}$. In the stage of training the Transformer Module, we fix the GNN module and transfer the information captured by the GNN module into the Transformer module by the loss $\mathcal{L}_{KL}$. 2) In the **joint training strategy**, the proposed modules are trained simultaneously. The loss function is defined by combining two losses: $\mathcal{L}_J = \mathcal{L}_{GNN} + \lambda \mathcal{L}_{KL}$, where λ is a pre-defined parameter to balance the two losses. For both strategies, the trained Transformer module can be used to build models for downstream tasks.

3 EXPERIMENT

In this section, we aim to validate the effectiveness of GE-BERT. We first fine-tune it on two **offline** query understanding tasks, i.e., query classification and query matching tasks. Then, we further demonstrate the effectiveness of the fine-tuned model in the **online** search system based on the medical query searching.

3.1 Datasets

We use different datasets according to the tasks. 1) For the **pre-training**, the model is trained on the query graph which is built on a large query log data generated by the Baidu search engine[1] with millions of users. We collected the initial search log data within a week to construct the query graph which consists of 92, 413, 932 query nodes and 957, 210, 956 edges. 2) In the **offline query classification** task, we predict the label for the input query. Each query

[1]https://www.baidu.com/

Table 1: Performance of the offline query classification task.

Methods		ACC	Precision	Recall	F1
BERT		0.6728	0.5941	0.5604	0.5722
BERT+Q		0.6862	0.6162	0.5708	0.5863
GAT-based	GE-BERT-J	0.6897	0.6215	0.5702	0.5874
	GE-BERT-S	0.7033	**0.6308**	0.5835	0.5997
GCN-based	GE-BERT-J	0.6945	0.6292	0.5758	0.5923
	GE-BERT-S	**0.7034**	0.6227	**0.5978**	**0.6034**

is associated with a label such as music and medical care. There are 100, 887 queries with 30 classes in total. 3) In the **offline query matching** task, we predict if the input query pairs have similar semantic meaning or not. Each query pair is associated with a label which is 1 when the two queries have similar semantic meanings, and 0 otherwise. There are 108, 615 query pairs with 2 classes in total. We adopt 60%, 20%, 20% for the training/validation/test split for offline tasks.

3.2 Offline tasks

3.2.1 Baselines and Model Variants. We consider two baselines based on the BERT model [4]. The first one directly uses BERT model in the downstream task, denoted as **BERT**. The second one further trains on the query graph under the graph reconstruction task, and we denote it as **BERT+Q**. Namely, different from the **BERT**, **BERT+Q** further trains the BERT parameters on the query graph by the loss function which has the same form as Eq.(3). **BERT+Q** is a baseline to evaluate the effectiveness of the GNN module to capture query graph information. Two variants are **GE-BERT-J** which jointly trains all proposed modules and **GE-BERT-S** which uses the stage-by-stage pre-training strategy.

3.2.2 Experimental Settings. For pre-training tasks, the number of stages in GE-BERT-S is set to be 4, and λ in GE-BERT-J is set to be 1. For the two downstream tasks, we utilize the Transformer module to build models. Specifically, for the query classification task, given a query, we feed the query embedding from the Transformer Module into a Multi-Layer Perceptron (MLP) for classification. For the query matching task, given a query pair, we feed their embeddings into a MLP and then calculate the inner product followed by the sigmoid function to generate the score for final prediction.

3.2.3 Performance . The experiment results are shown in Table 1 and Table 2 respectively. From these tables, we can make the following observations: 1) BERT+Q outperforms BERT consistently which demonstrates that incorporating query graph information can enhance Transformer to perform better. 2) In general, the proposed GE-BERT variants achieve better performance than BERT+Q, which indicates that the GNN module can capture the query graph information more effectively than pre-training with simple graph reconstruction task, which facilitates the performance. It also suggests that KL-divergence can indeed facilitate the process of transfer the information from the GNN module to the Transformer module.

3.3 Online task

In this subsection, we conduct an online A/B testing to validate the effectiveness of our models in the Baidu search engine. It is based

Table 2: Performance of the offline query matching task.

Methods		ACC	Precision	Recall	F1
BERT		0.6232	0.7825	0.6282	0.5661
BERT+Q		0.6254	0.7842	0.6303	0.5691
GAT-based	GE-BERT-J	0.6388	0.7887	0.6436	0.5891
	GE-BERT-S	0.6370	0.7881	0.6418	0.5865
GCN-based	GE-BERT-J	**0.6425**	**0.7899**	**0.6472**	**0.5944**
	GE-BERT-S	0.6312	0.7861	0.6317	0.5779

on the final ranked results for medical query searching, which is an important search scenario in the Baidu search engine. A crucial step for the medical query searching is to identify medical queries first. To achieve this goal, we treat it as a classification task and directly utilize the **fine-tuned** model (including the MLP layer) from the offline query classification task to perform the classification. In particular, given an input query, the fine-tuned model is able to tell whether it is a medical query or not. We have designed many strategies for medical queries to facilitate their ranking performance. Thus, different fine-tuned models might have different classification performance, which leads to different final ranked results on the search engine. We evaluate the model performance by comparing their ranked results. Due to the high cost of deploying multiple models, we evaluate the fine-tuned GE-BERT-S with GCN (**GE-BERT-S+F**) and make a comparison with the fine-tuned BERT+Q model (**BERT+Q+F**)

3.3.1 Experimental Settings and Result. We randomly sample a certain number of queries from the search log, and we apply GE-BERT-S+F and BERT+Q+F to identify the medical queries. Then we obtain 6, 268 queries which are predicted as medical queries by as least one of the two models. Among these queries, they make different predictions on 964 queries, i.e., the difference ratio is $964/6268 = 15.38\%$.

To compare the performance of GE-BERT-S+F and BERT+Q+F online, we compare their ranked results obtained from the search engine for these 964 queries, and use the (Good vs. Same vs. Bad) GSB [22] as the metric. More specifically, given a query, the annotators are provided with a pair $(result_1, result_2)$ where $result_1$ is the ranked result from GE-BERT-S+F and $result_2$ is the ranked result from BERT+Q+F. The result consists of multiple URLs. The annotators who don't know which model the result is generated are asked to rate the result independently: Good ($result_1$ is better), Bad ($result_2$ is better), and Same (they are equally good or bad) by considering the relevance between the ranked result and the given query. To quantify the human evaluation, the ΔGSB is defined by combining these three indicators:

$$\Delta GSB = \frac{\#Good - \#Bad}{\#Good + \#Same + \#Bad} \tag{4}$$

The statistic for computing the ΔGSB is shown in Table 3. We can observe that GE-BERT-S+F brings in substantial improvement and the advantage of GSB is 3.20%, which shows the superiority of our model to advance the online medical query searching.

3.3.2 Case Study. In this section, we further analyze the fine-tuned model by practical cases in query classification. They are presented in Table 4 where the second column is the true label of the query, the third and the forth column are predicted labels from GE-BERT-S+F

Table 3: Results of the online A/B testing.

diff_ratio	# Good	# Same	# Bad	ΔGSB
15.38%	57	881	26	3.20%

Table 4: Case study of query classification for the online task.

Query	Label	GE-BERT-S+F	BERT+Q+F
What's the weight of Nini	people	people	medical care
Dead Cells the fisherman	games	games	medical care
Which eyedrops is good for long-term use for students	medical care	medical care	product
What are the pros and cons of drinking soy milk	medical care	medical care	parenting

and BERT+Q+F respectively. We have the following observations: 1) For the first two queries, BERT+Q+F wrongly predicts them as medical queries. The potential reason might come from two misleading words *weight* and *Cells* because they are related to medical care. However, the first query is more about the word *Nini* (a famous actress in China) and *Dead Cells* in the second query is a video game. 2) For the third and the forth query, BERT+Q+F wrongly predicts them as other categories instead of the medical query. However, the word *eyedrops* in the third query is not simply a product but more related to the medical care. In the forth query, the word *milk* might be misleading so BERT+Q+F predicts it as *parenting*. In fact, the user cares more about the function of the soy milk so it's a medical query. Generally, these queries are difficult ones containing the misleading words. GE-BERT-S+F is able to make the right predictions. The potential reason is that the GNN utilizes the information of neighboring queries which can help to distinguish the misleading words.

4 CONCLUSION

In this paper, we introduce GE-BERT, a novel pre-training framework preserving both the semantic information and query graph information, which is further fine-tuned in the query understanding tasks. Specifically, we devise three modules: a Transformer module for extracting the semantic information of query content, a GNN module for capturing the query graph information, and a Transfer module to transfer the graph information from the GNN module to the Transformer module. To train the GE-BERT, two strategies are proposed: stage-by-stage strategy by separating the training of the Transformer module and the GNN module, and joint training by training all modules simultaneously. Comprehensive experiments in offline and online tasks demonstrate the superiority of GE-BERT.

5 ACKNOWLEDGEMENTS

6 PRESENTER AND COMPANY BIOGRAPHY

Presenter: Juanhui Li is a PhD student in computer science at Michigan State University. She's mainly interested in the graph mining including graph neural networks, link prediction and knowledge graph.

Company: Baidu Inc. is one of the largest AI and internet companies in the world which is founded as a search engine platform in 2000. Baidu provides full AI stack services including infrastructure consisting of AI chips, deep learning framework and open AI platform to facilitate a wide range of applications and services.

REFERENCES

[1] Steven M Beitzel, Eric C Jensen, Ophir Frieder, David D Lewis, Abdur Chowdhury, and Aleksander Kolcz. 2005. Improving automatic query classification via semi-supervised learning. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*. IEEE, 8–pp.

[2] Huanhuan Cao, Derek Hao Hu, Dou Shen, Daxin Jiang, Jian-Tao Sun, Enhong Chen, and Qiang Yang. 2009. Context-aware query classification. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. 3–10.

[3] Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909* (2019).

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. 4171–4186.

[5] Chanwoo Jeong, Sion Jang, Eunjeong Park, and Sungchul Choi. 2020. A context-aware citation recommendation model with BERT and graph convolutional networks. *Scientometrics* 124 (2020), 1907–1922.

[6] Shan Jiang, Yuening Hu, Changsung Kang, Tim Daly Jr, Dawei Yin, Yi Chang, and Chengxiang Zhai. 2016. Learning query and document relevance from a web-scale click graph. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 185–194.

[7] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

[8] Hang Li and Jun Xu. 2014. Semantic matching in search. *Foundations and Trends in Information retrieval* 7, 5 (2014), 343–469.

[9] Rongzhong Lian, Min Xie, Fan Wang, Jinhua Peng, and Hua Wu. 2019. Learning to select knowledge for response generation in dialog systems. *arXiv preprint arXiv:1902.04911* (2019).

[10] Xiaowei Liu, Weiwei Guo, Huiji Gao, and Bo Long. 2020. Deep Search Query Intent Understanding. *CoRR* abs/2008.06759 (2020).

[11] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[12] Agnès Mustar, Sylvain Lamprier, and Benjamin Piwowarski. 2020. Using BERT and BART for Query Suggestion. In *Joint Conference of the Information Retrieval Communities in Europe*, Vol. 2621. CEUR-WS. org.

[13] Junyuan Shang, Tengfei Ma, Cao Xiao, and Jimeng Sun. 2019. Pre-training of graph augmented transformers for medication recommendation. *arXiv preprint arXiv:1906.00346* (2019).

[14] Dou Shen, Rong Pan, Jian-Tao Sun, Jeffrey Junfeng Pan, Kangheng Wu, Jie Yin, and Qiang Yang. 2006. Query enrichment for web-query classification. *ACM Transactions on Information Systems (TOIS)* 24, 3 (2006), 320–352.

[15] Giorgos Stoilos, Nikos Papasarantopoulos, Pavlos Vougiouklis, and Patrik Bansky. 2022. Type Linking for Query Understanding and Semantic Search. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3931–3940.

[16] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223* (2019).

[17] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

[18] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144* (2016).

[19] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237* (2019).

[20] Haolan Zhan, Hainan Zhang, Hongshen Chen, Lei Shen, Yanyan Lan, Zhuoye Ding, and Dawei Yin. 2020. User-Inspired Posterior Network for Recommendation Reason Generation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1937–1940.

[21] Zhi Zheng, Kai Hui, Ben He, Xianpei Han, Le Sun, and Andrew Yates. 2020. BERT-QE: contextualized query expansion for document re-ranking. *arXiv preprint arXiv:2009.07258* (2020).

[22] Lixin Zou, Shengqiang Zhang, Hengyi Cai, Dehong Ma, Suqi Cheng, Shuaiqiang Wang, Daiting Shi, Zhicong Cheng, and Dawei Yin. 2021. Pre-trained language model based ranking in Baidu search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 4014–4022.