QUEEN: Neural Query Rewriting in E-commerce

Yaxuan Wang*
yaxuanw@amazon.com
Amazon.com Inc.
Seattle, Washington, USA

Hanqing Lu*
luhanqin@amazon.com
Amazon.com Inc.
Palo Alto, California, USA

Yunwen Xu
yunwenx@amazon.com
Amazon.com Inc.
Palo Alto, California, USA

Rahul Goutam
rgoutam@amazon.com
Amazon.com Inc.
Palo Alto, California, USA

Yiwei Song
ywsong@amazon.com
Amazon.com Inc.
Palo Alto, California, USA

Bing Yin
alexbyin@amazon.com
Amazon.com Inc.
Palo Alto, California, USA

ABSTRACT

Query rewriting (QR), which aims to improve the shopping experience by reformulating ambiguous customer input queries into well-formed queries, is a critical component of modern e-commerce search engines. In this work, we present a practical deep learning solution, named as Query Understanding EnhancEd mechaNism (QUEEN), to the large-scale query rewriting problem in e-commerce search engines. QUEEN incorporates query annotations, the by-product of query processing pipelines in most e-commerce search engines, to model ambiguous product search queries. The empirical study is based on 38.5 million anonymous product search queries. Compared to other SOTA baselines, QUEEN improves the sentence-level recall by 6% (relatively).

CCS CONCEPTS

• **Computing methodologies** → **Information extraction**.

KEYWORDS

product search, query processing, information retrieval

ACM Reference Format:
Yaxuan Wang, Hanqing Lu, Yunwen Xu, Rahul Goutam, Yiwei Song, and Bing Yin. 2021. QUEEN: Neural Query Rewriting in E-commerce. In *Proceedings of ACM The Web Conference Workshop on Knowledge Management in e-Commerce (WWW KMEcommerce'21).* ACM, New York, NY, USA, 4 pages.

1 INTRODUCTION

Product search comes with fewer feedback signals than general web search because actions are much sparser. Although the rich customer feedback can help understand queries that are frequently searched, the feedback data of infrequent queries is relatively limited. Due to the context shortage or vocabulary gap between the query and its targeted product document, an infrequent query usually yields less relevant search results.

In this work, we try to answer the question: can we train a deep learning model to reformulate ambiguous and infrequent queries which derive unsatisfying results? Following this motivation, we analyzed one month of anonymous product search logs from a leading e-commerce marketplace. We found that, in many sessions, customers iteratively clarify their shopping intents by reformulating search queries. For instance, a customer comes to an e-commerce site and enters a query *"4xlt big and tall,"* the back-end system does not understand it and does not return any results. Then s/he reformulates the query into *"xxxxl tshirt big,"* which is more expressive. This solution allows the back-end system to find plenty of related products. Then the customer begins to engage with the search results by clicking on or purchasing them. This series of behaviors can be extracted as a reformulation pair *(source: 4xlt big and tall, target: xxxxl tshirt big, action: purchase)*, and a deep learning model can be trained to mimic this behavior and automatically do the reformulation.

In this paper, we propose QUEEN, a transformer-based model that is trained on reformulation pairs, as a solution to the query rewriting problem in e-commerce. To overcome the limitations of context shortage and erroneous semantics, we use query annotations, a by-product of the query understanding pipeline in most e-commerce search engines, to guide the learning process. Query annotations demonstrate the functional role of each token, which introduces important domain knowledge to the model. Since the query annotation is one output of the query understanding pipeline, it does not generate any additional cost. Beyond that, introducing the annotation actually reduces its number of parameters when comparing to the baseline deep learning models.

The rest of the paper is organized as follows: in Section 2, we introduce some related works, in Section 3, we describe the model architecture of QUEEN, in Section 4, we show the empirical results on anonymous product search logs, and we present our conclusion in Section 5.

2 RELATED WORKS

Query rewriting (QR) is a conventional topic in Information Retrieval (IR) area. Some research studies are based on the idea of utilizing the relevance between queries and documents. For example, [14] presents a solution to combine the word-level context features of queries and documents, but the retrieval phase is too expensive to be applied in the large scale e-commerce search engine. On the other hand, some researchers focus on the query information.

*Both authors contributed equally to this paper.

WWW KMEcommerce'21, April 2021, Virtual Event

For instance, [2] presents the first probabilistic query expansion method based on query click rate, and [1, 3] are developed based on the relevance of queries such as co-click similarity. Given limited context and customer behavior, either more advanced technologies or domain knowledge is required.

Embedding approaches which enhance the representation of queries have shown strong power in the query rewriting task. [10] aims to generate semantic representation for queries and phrases by embedding both query and document into specific matrices. In [4], the authors propose a context-aware embedding approach that considers all the queries in a specific session. Recently, seq2seq models show remarkable reliability and flexibility in the query rewriting task In [5], a 2-step query rewriting framework is presented. To help preserve important keywords of the original query, [7] introduces additional decoders. Besides, [13] integrates the user search log, an important information source, with basic seq2seq model. In our paper, we collect training data from the anonymous user search log, which contains positive feedback such as clicks or purchases. Moreover, our method adopts domain knowledge to tackle common issues in tail queries, such as insufficiency of query context and customer feedback.

3 METHODOLOGY

In this section, we present QUEEN, a novel embedding mechanism that aims to provide a query-to-query reformulation solution by integrating the keyword annotation from domain knowledge. It is applicable to most seq2seq and transformer models that have encoder-decoder structures.

3.1 Notations and Problem Definition

Given the source query S and its target query T, we can pre-process S and T into two sub-word sequences, denoted by $\{s_1, s_2, ..., s_n\}$ and $\{t_1, t_2, ..., t_m\}$, where n and m are the length of S and T after word segmentation [9]. Meanwhile, we denote the annotation aligning the source query as $\{a_1, a_2, ..., a_n\}$. The annotation is generated by the our query understanding pipeline, and the details of this pipeline are beyond the scope of our paper. Our query-to-query generation model θ generates query candidate set $\boldsymbol{C}$ where each candidate $C_i = \{c_1, c_2, ..., c_l\}$ and c_j is a sub-word.

Formally, we define the problem of annotation integrated query rewriting as follows: given an input query S with its annotation A, our model θ predicts a set of query candidates $\boldsymbol{C} = \{C_1, ..., C_k\}$ and each candidate C_i is generated by maximizing the conditional probability distribution as follows:

$$\arg\max_{\theta} \prod_{i=1}^{l} P(c_i | c_{i-1}, c_{i-2}, ..., c_1, S, A; \theta) \tag{1}$$

3.2 Model Architecture

In general, our model consists of two parts: a hybrid encoder and a decoder.

Hybrid encoder. Given an input query token and its annotation, the hybrid encoder integrates the semantics and annotations. It comprises two parts: one embedding layer and a stack of encoding layers. The embedding layer maps discrete words into continuous word embedding $\{\vec{w_1}, \vec{w_2}, ..., \vec{w_n}\}$. Similarly with original transformer model[12], the hybrid encoder generates positional embedding which has the same number of dimension d_{model} with the word embedding. Then, the encoder generates continuous annotation embedding $\{\vec{a_1}, \vec{a_2}, ..., \vec{a_n}\}$ given input annotation sequence. Different from knowledge graph based studies that focus on recognizing the query entity, our method analyzes the role of each token based on the query context.The roles of a token can be something like <*brand*>, <*color*>, and <*unknown*>. Finally, the embedding layer combines the word embedding, positional embedding, and annotation embedding. There are two ways to combine those embeddings: *Addition* and *Concatenation*. The details are as follows:

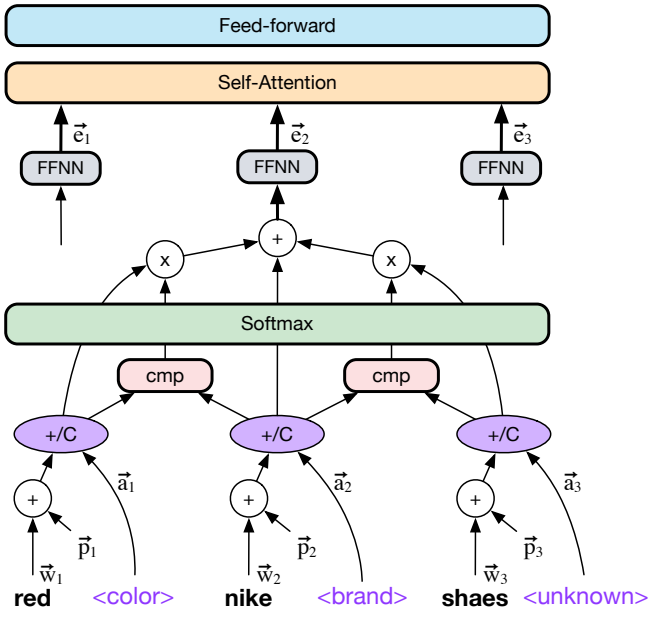

Figure 1: Hybrid transformer encoder which integrates word embedding w_i, positional embedding p_i, and annotation embedding a_i. In this example, "shaes" which can not be annotated by domain knowledge is a spelling error of "shoes".

- *Addition:* the final embedding is derived by adding word, positional, and annotation embedding :

$$\{\vec{e_1}, \vec{e_2}, ..., \vec{e_n}\} = \{[\vec{w_1} + \vec{p_1} + \vec{a_1}], ..., [\vec{w_n} + \vec{p_n} + \vec{a_n}]\}, e_i \in \mathbb{R}^{d_i}, d_i = d_w = d_a$$

- *Concatenation:* the final embedding is derived by concatenating the annotation embedding with the addition of word and positional embedding:

$$\{\vec{e_1}, \vec{e_2}, ..., \vec{e_n}\} = \{[(\vec{w_1} + \vec{p_1}) \oplus \vec{a_1}], ..., [(\vec{w_n} + \vec{p_n}) \oplus \vec{a_n}]\}, e_i \in \mathbb{R}^{d_i}, d_i = d_w + d_a$$

The extensive experiments show that concatenating the annotation may achieve higher accuracy but sufficient parameter tuning (such as how many dimensions of annotation/word embedding is optimal) is inevitable. In this work, we chose *Concatenation*-based model.

Decoder. Since predicted candidates are not required to have annotations, our model employs normal decoder structures.

4 EXPERIMENTS

We evaluate the impact of QUEEN by comparing the prediction performance of different models. In Section 4.1, we explain how to collect experimental data from anonymous e-commerce search logs. Then we introduce baseline models and parameter settings in Section 4.2. Next, we introduced evaluation metrics in Section 4.3. In Section 4.4 and 4.5, we analyze overall experimental results and the impact of different combination methods. Finally, we discuss the model complexity in Section 4.6.

4.1 Dataset Creation

We collect an empirical dataset from the anonymous search logs of a large e-commerce site during April 2019. When a customer comes to this site and starts a new search session, search queries will be collected if the customer's behavior within one search session matches the pattern as below:

(1) Start a product search session by searching with a query.
(2) No action is taken on the current search result.
(3) Start a second product search with a rewritten query.
(4) Click/purchase on the search result of the rewritten query.

The original and rewritten query are marked as the source and the target query. To reduce noise in the scratch data, we remove following queries: 1) empty queries; 2) non-English queries; 3) queries which contain an ISBN number or an URL; 4) irrelevant query pairs which has no overlap tokens between source and target query; 5) wordy queries with the number of token greater than 10. In total, we collect 38.5 million pairs of queries, where more than 90% of the source queries have less than or equal to 6 tokens. We split them into training set (37.9 million pairs), validation set (315 thousand pairs), and test set (315 thousand pairs).

4.2 Compared Methods and Parameter Settings

Theoretically, QUEEN is compatible with lots of models with encoder-decoder structures. Without loss of generality, we adopt two widely used models, long short-term memory (LSTM) [11], and transformer[12], to create two variants in our experiments. Here are the details and the parameter settings of them:

QUEEN-L: QUEEN-L is generated by integrating QUEEN with a 6-layer LSTM model with 512 hidden units on both the encoder and decoder.

QUEEN-T: QUEEN-T is generated by integrating QUEEN with a 6-layer encoder/decoder transformer model. Each layer has a multi-head attention layer which contains eight heads. We used Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.998$.

In both QUEEN-T and QUEEN-L, the annotation embedding is concatenated with the addition of word embedding and positional embedding, and we set the dimension of annotation embedding to be 16.

Our implementation is based on OpenNMT[6], an open-source Neural Machine Translation framework. Both source and target queries are preprocessed using byte-pair encoding [9] with shared vocabulary that contains about 32,000 tokens. We set the dropout rate to 0.1 and the embedding dimension to 512 for all models. We train each model on an NVIDIA Tesla V100 GPU. All models are trained for a total of 800,000 iterations. We compare the performance of QUEEN-T and QUEEN-L to their baseline models:

LSTM [11]: a 6-layer LSTM model with 512 hidden units
Transformer [12]: a 6-layer transformer model with 512 units.

4.3 Evaluation Metrics

We use following evaluation metrics:

Recall@8: The sentence-level Recall@8 is the proportion of test query pairs that the target query matches one of the top-8 predicted candidates.

Jaccard@8: The word-level Jaccard similarity quantifies *order-insensitive* query similarity. For each test query pair, Jaccard@8 is the highest Jaccard similarity between the top-8 predicted candidates and the target query.

BLEU@8: BLEU score[8] is widely adopted for evaluating n-grams overlap. For each test query pair, BLEU@8 is the highest BLEU score between the top-8 predicted candidates and the target query.

4.4 Evaluation Results

Table 1: The performance of QUEEN-T and QUEEN-L compared to their baseline models

Algorithm		Relative Gain		
Our method	Baseline	Recall@8	Jaccard@8	BLEU@8
QUEEN-L	LSTM	+6.60%	+6.94%	+23.33%
QUEEN-T	Transformer	+4.55%	+1.97%	+2.94%

Table 1 shows the prediction performance of all models on the testing data. From Table 1, we can observe that both QUEEN-L and QUEEN-T outperforms their baseline versions, which indicates that involving query annotation further improves deep-learning models' performance. Compared to plain LSTM and Transformer model, QUEEN-L and QUEEN-T have higher Recall@8, Jaccard@8, and BLEU@8.

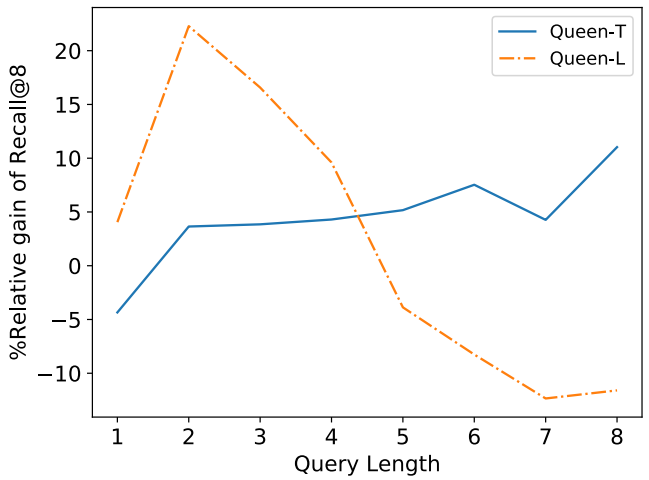

Figure 2: Analysis of different models by query length. For each metric, the x-axis is the length of source query. The y-axis is the relative gain of Recall@8.

To demonstrate the impact of QUEEN on different models, we analyze the incremental improvement of QUEEN-T and QUEEN-L by the length of source query. In Figure 2, we plot the relative gain of recall@8 of QUEEN-T and QUEEN-L compared to Transformer and LSTM model under different query lengths. From this figure, we observe that QUEEN-T performs well for long queries than Transformer, which demonstrates the ability of capturing the long-term dependency. For short queries (length < 4), QUEEN-L shows obvious improvement comparing to the baseline LSTM model.

4.5 Model Complexity

In this subsection, we would like to study the annotation embedding's impact to the model complexity We vary the dimension of annotation embedding between 256, 128, 64, 32, and 16, while we keep the dimension of final input embedding as 512. The dimensions of the addition of word embedding are 256, 384, 448, 480, and 496, respectively. We plot the results in Figure 3, and it shows that the model complexity is reducing as the dimension of annotation embedding is increasing. The reason is: annotation has a much smaller vocabulary than normal language, so keep the same dimension of the final embedding while assign more dimension to the annotation embedding reduces complexity of the model.

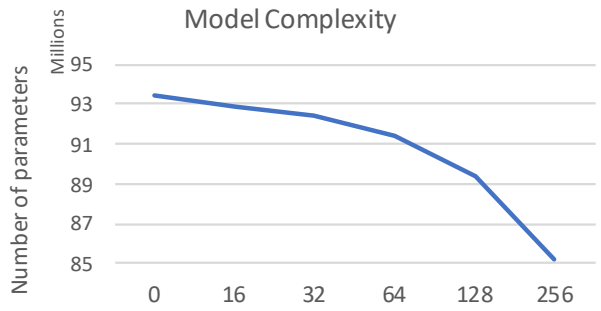

Figure 3: Number of parameters under different dimensions of the annotation embedding.

5 CONCLUSION

In this paper, we formulate the query rewriting task as a query-to-query generation problem. Specifically, we present a practical deep learning solution for the query rewriting problem. By integrating the query annotations, which is the by-product of the current production system, our model improve the prediction accuracy and enhanced the overall search experience. Empirical results built on 38.5 million anonymous customer search logs demonstrates that our proposed model not only achieves better prediction performance but also reduces the complexity of baseline deep learning models.

REFERENCES

[1] Ioannis Antonellis, Hector Garcia Molina, and Chi Chao Chang. 2008. Simrank++: query rewriting through link analysis of the click graph. *Proceedings of the VLDB Endowment* 1, 1 (2008), 408–421.

[2] Hang Cui, Ji-Rong Wen, Jian-Yun Nie, and Wei-Ying Ma. 2002. Probabilistic query expansion using query logs. In *Proceedings of the 11th international conference on World Wide Web*. ACM, 325–332.

[3] Bruno M Fonseca, Paulo Golgher, Bruno Pôssas, Berthier Ribeiro-Neto, and Nivio Ziviani. 2005. Concept-based interactive query expansion. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM, 696–703.

[4] Mihajlo Grbovic, Nemanja Djuric, Vladan Radosavljevic, Fabrizio Silvestri, and Narayan Bhamidipati. 2015. Context-and content-aware embeddings for query rewriting in sponsored search. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. ACM, 383–392.

[5] Yunlong He, Jiliang Tang, Hua Ouyang, Changsung Kang, Dawei Yin, and Yi Chang. 2016. Learning to rewrite queries. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 1443–1452.

[6] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. 2017. OpenNMT: Open-Source Toolkit for Neural Machine Translation. In *Proc. ACL*. https://doi.org/10.18653/v1/P17-4012

[7] Xiaoyu Liu, Shunda Pan, Qi Zhang, Yu-Gang Jiang, and Xuanjing Huang. 2018. Generating Keyword Queries for Natural Language Queries to Alleviate Lexical Chasm Problem. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 1163–1172.

[8] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 311–318.

[9] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2016). https://doi.org/10.18653/v1/p16-1162

[10] Alessandro Sordoni, Yoshua Bengio, and Jian-Yun Nie. 2014. Learning concept embeddings for query expansion by quantum entropy minimization. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*.

[11] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. arXiv:cs.CL/1409.3215

[12] 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.

[13] Jingang Wang, Junfeng Tian, Long Qiu, Sheng Li, Jun Lang, Luo Si, and Man Lan. 2018. A Multi-task Learning Approach for Improving Product Title Compression with User Search Log Data. arXiv:cs.CL/1801.01725

[14] Jinxi Xu and W Bruce Croft. 2017. Quary expansion using local and global document analysis. In *Acm sigir forum*, Vol. 51. ACM, 168–175.