Query Expansion and Entity Weighting for Query Reformulation Retrieval in Voice Assistant Systems

Zhongkai Sun
zhongkas@amazon.com
Amazon Alexa AI
Seattle, WA, USA

Sixing Lu
cynthilu@amazon.com
Amazon Alexa AI
Seattle, WA, USA

Chengyuan Ma
mchengyu@amazon.com
Amazon Alexa AI
Seattle, WA, USA

Xiaohu Liu
derecliu@amazon.com
Amazon Alexa AI
Seattle, WA, USA

Chenlei Guo
guochenl@amazon.com
Amazon Alexa AI
Seattle, WA, USA

ABSTRACT

Voice assistants such as Alexa, Siri, and Google Assistant have become increasingly popular worldwide. However, linguistic variations, variability of speech patterns, ambient acoustic conditions, and other such factors are often correlated with the assistants misinterpreting the user's query. In order to provide better customer experience, retrieval based query reformulation (QR) systems are widely used to reformulate those misinterpreted user queries. Current QR systems typically focus on neural retrieval model training or direct entities retrieval for the reformulating. However, these methods rarely focus on query expansion and entity weighting simultaneously, which may limit the scope and accuracy of the query reformulation retrieval. In this work, we propose a novel Query Expansion and Entity Weighting method (QEEW), which leverages the relationships between entities in the entity catalog (consisting of users' queries, assistant's responses, and corresponding entities), to enhance the query reformulation performance. Experiments on Alexa annotated data demonstrate that QEEW improves all top precision metrics, particularly 6% improvement in top10 precision, compared with baselines not using query expansion and weighting; and more than 5% improvement in top10 precision compared with other baselines using query expansion and weighting.

CCS CONCEPTS

• **Information systems → Novelty in information retrieval**.

KEYWORDS

query expansion, entity weighting, query reformulation, retrieval system

ACM Reference Format:

Zhongkai Sun, Sixing Lu, Chengyuan Ma, Xiaohu Liu, and Chenlei Guo. 2022. Query Expansion and Entity Weighting for Query Reformulation Retrieval in Voice Assistant Systems. In *Phoenix '22: the 15th ACM International WSDM Conference, Feb 21–25, 2022, Phoenix, AZ*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/1122445.1122456

Phoenix '22, Feb 21–25, 2022, Phoenix, AZ

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Virtual voice assistants such as Alexa, Siri, and Google Assistant have become daily necessities for worldwide users. There are two main components in such systems, namely Automatic Speech Recognition (ASR) and Natural Language Understanding (NLU) to extract the semantic information from an input voice query. However, misinterpretations may occur owing to factors such as semantic/speech pattern variants and ambient acoustic conditions, resulting in incorrect responses and unsatisfactory user experience. Therefore, it has become a vital yet challenging task to enable voice assistants to correctly understand those misinterpreted user queries.

Many works have been devoted to query reformulating (QR) to correct misinterpreted user queries [2, 3, 7, 11, 22, 23]. Among QR systems, the retrieval and ranking based approaches are efficient and widely-used in the industry [4, 7, 22]. These works focus more on retrieval models without additional entity information. However, in voice assistant systems where task-oriented queries are usually short and ambiguous, the query intention may not be fully interpreted and this limitation negatively impact the performance of QR system whose relevance strongly relies on the query. For instance, a user's input query ("play long distance love by Sheena Easton") contains a correct singer name ("Sheena Easton") but also a lyric ("long distance love") instead of the correct song name ("telefone"). To reformulate this query so that virtual voice assistants can find the target song to play, the QR system needs to utilize the existing entities to obtain the correct song name "telefone"; besides, the model also needs to make a decision between two candidates ("play long distance love by little feat" and "play telefone by sheena easton") as they both contain key words ("long distance love" or "sheena easton") in the original query.

Query expansion is an effective approach to enrich the original oral query. Some researches consider the correlations between a query and its expansion candidates to select the most relevant candidate from the candidate pool: [5] proposes a query expansion method based on probabilistic correlations between query logs; [12] uses a DNN model to directly predict the scores for query expansion terms. Both approaches make query more complete but do not consider the weighting between original query and expanded ones. Other works have studied how to directly generate the expansions:

[8] proposes a concept-based translation method to generate the expanded query; [17] uses BERT [6] and BART [14] with a next query prediction task to do query suggestion; docT5query [18] implements a powerful T5 model [19] to predict the relevant query for each document. The generative approaches are able to provide large number of variants, but it may also bring noise and lead to incorrect reformulations.

Entity (term/phrase) weighting is also a common approach to emphasize the key words in the query and improve the retrieval performance. To predict the weight of each entity, conventional methods use the corpus statistics based methods like TF-IDF, BM25, and SRNM [13, 24] to compute the scores of entities. Recently, neural network based methods have been widely studied: [20] uses a supervised query performance prediction method to learn query term weighting; [25] utilizes a random-walk model to learn latent representations for candidates to capture the relative importance between entities; [1] uses pre-trained language models to learn sparse entity representations in order to better predict the importance of each entity. However, these works haven't utilized the expansion and term weighting simultaneously. DeepImapct [16] first expands the documents using docT5query [18], and then uses a contrastive loss to predict the weight of each term in the query and document. However, DeepImapct focuses on generating the expansions for long documents (e.g. a paragraph), but it has not been expanded to short query scenarios.

In this paper, we integrate query expansion and entity weighting together to propose a novel method called **Q**uery **E**xpansion and **E**ntity **W**eights prediction (QEEW) for query reformulation. Specifically, QEEW first establishes an entity Expansion Knowledge Base (EEKB) by categorizing and linking entities in an entity catalog (which consists of user queries, assistant's responses and corresponding entities) to expand queries. An entity weighting prediction model is then trained using annotated query reformulation pairs with expansions obtained from the EEKB. By integrating query entity weighting with query expansion, QEEW can retrieve reformulations more relevant to the user's intention. Our evaluation on annotated Alexa data shows that the proposed QEEW improves 2.6%, 5.3%, 1.1% on precision@50, precision@10, and precision@1 compared with best scores obtained from using other methods.

2 METHODOLOGY

This section describes the details of the proposed QEEW method. The training data are pairs of annotated misinterpreted queries and their corresponding reformulations, and the EEKB is built from a pre-established user-satisfied entity catalog. To determine if a user's query is successfully processed by the voice assistant, an automatic friction estimation system [10] is applied, and only the non-frictional queries and their corresponding assistant's responses in the catalog will be used to build the EEKB. As shown in Figure 1, our proposed QEEW model mainly consists of two components:

- **Entity Expansion Knowledge Base** An EEKB which contains connections between different entities is built by categorizing entities based on their occurrences in queries and responses, both of which are obtained form the user satisfied entity catalog. Once this knowledge base is built, it will be used to provide expansion candidates in both training and test steps. Details of the building EEKB and how to expand query with it will be introduced in section 2.1.
- **Entity Weight Prediction Model** After expanding the query from the EEKB, an entity weight prediction model will be trained to predict the weighting scores of both entities in the original query and those expansions obtained from the EEKB. Section 2.2 introduces the details of this entity weight prediction model.

The expanded query with the predicted entity weights can then be used in the downstream retrieval/ranking QR system to obtain the final reformulation.

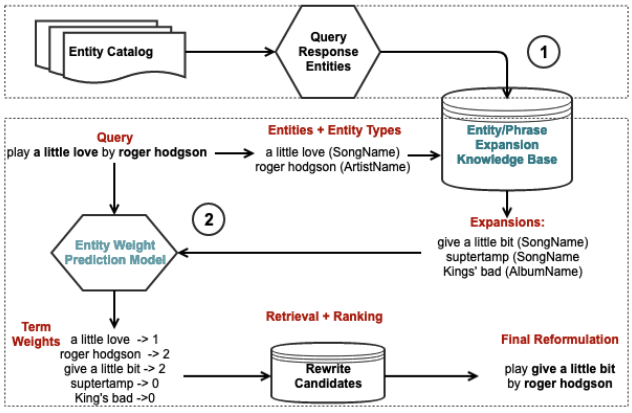

Figure 1: The overview of the workflow: 1) entity catalog data is first used to build the Entity Expansion Knowledge Base; 2) an entity weight prediction model is trained with annotated Alexa <query, reformulation> pairs.

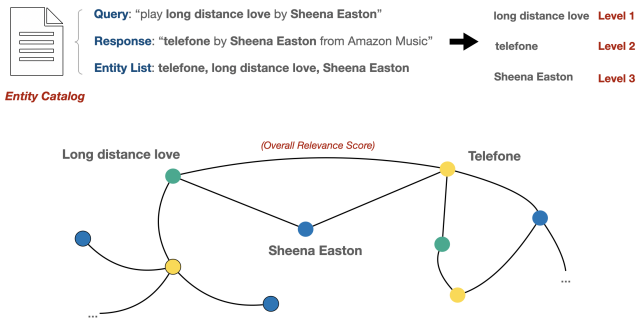

Figure 2: Illustration of the building process of the entity connection knowledge base. First, each entity in one entity catalog entry will be categorized based on its occurrence in this entry's query and response. Second, every two entities in the same entry will be connected with each other. Third, for each edge, the "overall relevance score" is calculated based on these two entities' respective scores in all times they appear together.

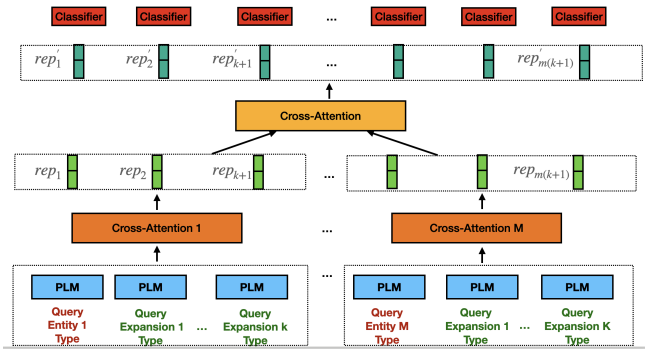

Figure 3: The illustration of the entity weight prediction model. A hierarchical structure is applied to predict the relative weights for all entities

2.1 Entity Expansion Knowledge Base

The EEKB is built using a comprehensive high-quality entity catalog consisting of user's queries, user-satisfied assistant's responses, and every query-response pair's unique entity list. This entity catalog is able to provide corrections for those inaccurate entities; and the voice assistant's responses can also help to indicate entities' relative importance. By connecting each entity with others and categorizing each entity according to its existence in query and response, the EEKB can be established with the relationships between different types of entities. Therefore, a query can be expanded by finding the most relevant expansions in this EEKB for every entity in the query.

Algorithm 1 Building the entity expansion knowledge base

Input: **ENTITY CATALOG** $\mathbb{C} = \{(C_i\}, 1 \le i \le K,$
where $C_i = \{Query_i, Response_i, E_i\}$,
where $E_i \leftarrow$unique entities in $\{\text{Query}_i, Response_i\}$
KG $\leftarrow\{\}$
for $i \in \{1, \ldots, K\}$ **do**
  for e_j in E_i **do**
    if e_j in $Query_i, Response_i$ **then**
      $Level_j \leftarrow 3$
    else if e_j in $Response_i$ **then**
      $Level_j \leftarrow 2$
    else
      $Level_j \leftarrow 1$
    end if
  end for
  for e_m, e_n in $E_i, m \neq n$ **do**
    $KG[e_m][e_n]$ += $level_m \times level_n$
    $KG[e_n][e_m]$ += $level_n \times level_m$
  end for
end for

Figure 2 demonstrates the building process of the EEKB. Nodes in the EEKB are entities from the entity catalog, and each node is connected with others when both nodes appear in the same entry in the entity catalog. The score on the edge indicates the overall relevance level of two nodes. For an entity in one entry, its current relevance level is based on its occurrence in query and response. Level 1 means that this entity only exists in the query; level 2 represents that it only exists in the response; and level 3 indicates that it exists in both query and response. The motivation is that the higher the relevance level, the more relevant the corresponding entity with the query is. In the example in Figure 2, the entity "Sheena Easton" has a relevance level 3 because it exists in both query and response. Every two entities' relevance levels are also recorded. For instance, in Figure 2, "$telefone$" and "$SheenaEaston$" will have a relevance level pair $[level2, level3]$. The overall relevance level score between two connected nodes is defined as the sum of the product of the their respective scores each time they appear together. The detailed algorithm is presented in Algorithm 1.

Once EEKB is established, it can be used to expand a specific query. For each entity in the original target query, its K most relevant nodes (ranked according to the overall relevance score) is obtained from the knowledge base. However, since typically only limited expansions are useful for reformulation retrieval, the weighting model trained in 2.2 can help to polish the expansion. For example, some of the expanded phases may be assigned weight=0 and will be removed from the expansion.

2.2 Entity Weights Prediction Model

The weight prediction model is trained for entities in the original query as well as expansions obtained from the EEKB in section 2.1. The training data is the labelled Alexa query reformulation pairs, and the weight to be predicted for each entity is a specific weighting score level based on the reformulation.

As shown in the workflow in Figure 1, the task is defined as predicting the weighting score level of all entities (both original and expanded) for a given query. Entities exist in the reformulation will have higher weighting score levels while those not in the reformulation will have lower score levels. The weighting score level of each entity is assigned according to the existence in the labelled Alexa query and reformulation pairs: 1) if this entity exist neither in the original defective query nor in the reformulation, the score level is labelled as 0; 2) the score level of entities in original query will be labeled as 1; 3) the score level of entities in reformulation (either in request or not) will be labeled as 2, which indicates the most important weight.

The weight training task is defined as: give a query Q with m entities $E_1, E_2, ..., E_m$ and each entity E_i has k expansions $E_i^1, E_i^2, ..., E_i^k$, the target is to predict the importance score l_p for every entity E_p (both original and expanded) where $1 \le p \le m(k+1)$. In this work, we train a hierarchical model to predict each entity's weight. Figure 3 demonstrates the structure of the hierarchical model, which can better learning the relationships between each entity and its expansions instead of predicting all of the weights in one step.

As shown in Figure 3, a pretrained language model (PLM) is used to encode each entity E_p. Specifically, each E_p is concatenated with the original query and its entity type (e.g., "SongName", "AuthorName") to form the input text "Query [SEP] Entity [SEP] Type". This input text will be encoded to an embedding by the PLM. After that, M first-layer cross-attentions modules are applied to M raw entity and their expansions. So that each first-layer cross-attention

Method	P@50	P@10	P@1
DeepImapct	63.8%	32.4%	16.3%
Elasticsearch	85.4%	67.6%	20.6%
docT5query on Elasticsearch	85.6%	68.3%	20.5%
QEEW on Elasticsearch	**88.2%**	**73.6%**	25.1%
SimCSE	72.7%	64.0%	33.7%
QEEW on SimCSE	75.9%	64.1%	**34.8%**

Table 1: Main result table. QEEW on Elasticsearch is able to achieve significantly better performance on P@50 and P@10; QEEW on SimCSE achieves the best performane on P@1.

Method	P@50	P@10	P@1
Elasticsearch	85.4%	67.6%	20.6%
Elasticsearch with expansion	86.3%	68.5%	21.1%
Elasticsearch with weight	87.4%	73.1%	25.0%
QEEW on Elasticsearch	**88.2%**	**73.6%**	**25.1%**

Table 2: The ablation study on Elasticsearch demonstrates the usefulness of both expansion and weight prediction.

module can learn the relationships between each raw entity and its expansions. At last, each entity's representation rep_p (where $1 \le p \le m(k+1)$) from all first cross-attention modules are jointly input to the second cross attention module. Therefore, the interactions between all entities can be learned and the outputs rep'_p (where$1 \le p \le m(k+1)$) will then be input to the final classification layer to predict the importance weight logits g_p. Cross-entropy is used to calculate the loss between each entity's predicted logits g_p and its true label l_p, the final loss is defined as the average of these losses:

$$\mathcal{L}_{final} = \frac{1}{m(k+1)} \sum_{p=1}^{m(k+1)} \text{CrossEntropy}(l_p, g_p) \tag{1}$$

3 EXPERIMENTS AND EVALUATION

We extract 2000 hours of successfully processed Alexa data from an aggregated and anonymized datalog (which has no user specific information), and use it to build the EEKB and the retrieval index. An annotated query-reformulation dataset is split into 60K/15k/27k, to be used for training, evaluation, and testing, respectively.

The Robert-Base[15] model is utilized as the encoder and a multi-head attention module is used as the cross-attention layer in Figure 3. Hyper-parameters of the model can be found in Appendix A. P@K (Precision of top-K retrieved candidates) is used as the evaluation metrics, and four baselines are used for the comparison.
1) The ElasticSearch[1], which is a popular string match retrieval method based on the inverted indexing and BM25 [13];
2) The SimCSE [9], which is a neural network based model that leverages the contrastive learning to build a representation encoder;
3) DocT5query [18], which learns to expand the document by predicting its relevant query using the T5 model. The expanded document can be used in a specific retrieval system, e.g. Elasticsearch.
4) DeepImpact [16], which performs both document expansion and semantic importance estimation using DocT5Query and a contrastive learning based term prediction.

To evaluate the effectiveness of QEEW, we applied it to 1) string-match based Elasticsearch and 2) the neural embedding based SimCSE. On Elasticsearch, we first build the reformulation index from reformulation candidates set, and append each query with its expanded entities to form a new query. The predicted weights are applied to the retrieval score function so that the higher weighted entities contribute to higher scores for ranking; On SimCSE, the retrieval of ranking reformulations is based on the similarity score between query and reformulation strings, and the relevance score is the Euclidean Distance between the embeddings of query and reformulations. To apply the QEEW, we also adjust the similarity scores of retrieval reformulations based on the weights so that the reformulations that contain higher weighted entities will be with higher ranking scores.

Table 1 shows that the proposed QEEW method can achieve the highest P@50 and P@1 when applied to Elasticsearch, and achieves the best P@1 when applied to the SimCSE. Specifically, QEEW can improve P@50, P@10, and P@1 by 2.8%, 6%, and 4.5% respectively in Elasticsearch and by 3.2%, 0.1%, and 1.1% respectively in SimCSE. Notice that the SimCSE achieves higher P@1 but lower P@10/P@50 compared with Elasticsearch, this is because that the SimCSE is trained with a semantic matching objective so that it performs better on precise matching, while Elasticsearch enables a wider search and thus achieves higher P@50 and P@10. When comparing with (3) DocT5Query and (4) DeepImapct, our proposed method still achieves better performance. Specifically, Elasticsearch on QEEW can improve 2.6%, 5.3%, and 4.6% on P@50, P@10, and P@1 respectively compared to docT5query on Elasticsearch; QEEW on SimCSE can improve 12.2%, 31.7%, and 18.5% on P@50, P@10, and P@1 respectively compared to DeepImapct. The reason why (3) and (4) cannot achieve as good performance as our proposal is that these models are more suitable for longer paragraphs but fail to perfectly expand to the short and ambiguous spoken queries.

Table 2 demonstrates the ablation study result of applying QEEW to Elasticsearch. Elasticsearch with predicted expansions singly is able to gain 0.9%, 0.9%, 0.5% on P@50, P@10, and P@1; while with predicted weights singly can improve 2%, 5.5%, 4.4% on P@50, P@10, and P@1. This result demonstrates weighting can significantly improve the performance compared to query expansion alone.

4 CONCLUSIONS AND FUTURE WORK

In this work, we propose a novel method named as QEEW for query expansion and entity weighting predicion. Experiments on annotated Alexa data demonstrates that 1) QEEW can increase retrieval precision when applying to both conventional string-matching retrieval system (e.g., Elasticsearch) and novel neural network retrieval system (e.g., SimCSE); 2) QEEW achieves better performance compared with SOTA retrieval based QR systems (e.g., DocT5Query, DeepImpact).

For the future work, we would like to investigate more effective ways to learn entity extension/ weighting and how to leverage the EEKB and predicted entity weights to improve other tasks such as recommendation and personalization.

[1]Version 7.10 is used in this work

REFERENCES

[1] Yang Bai, Xiaoguang Li, Gang Wang, Chaoliang Zhang, Lifeng Shang, Jun Xu, Zhaowei Wang, Fangshan Wang, and Qun Liu. 2020. SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval. *arXiv preprint arXiv:2010.00768* (2020).

[2] Christian Buck, Jannis Bulian, Massimiliano Ciaramita, Wojciech Gajewski, Andrea Gesmundo, Neil Houlsby, and Wei Wang. 2017. Ask the right questions: Active question reformulation with reinforcement learning. *arXiv preprint arXiv:1705.07830* (2017).

[3] Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. 2008. Context-aware query suggestion by mining click-through and session data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 875–883.

[4] Eunah Cho, Ziyan Jiang, Jie Hao, Zheng Chen, Saurabh Gupta, Xing Fan, and Chenlei Guo. 2021. Personalized Search-based Query Rewrite System for Conversational AI. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*. 179–188.

[5] 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*. 325–332.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[7] Xing Fan, Eunah Cho, Xiaojiang Huang, and Chenlei Guo. 2021. Search based Self-Learning Query Rewrite System in Conversational AI. In *2nd International Workshop on Data-Efficient Machine Learning (De-MaL)*.

[8] Jianfeng Gao and Jian-Yun Nie. 2012. Towards concept-based translation models using search logs for query expansion. In *Proceedings of the 21st ACM international conference on Information and knowledge management*. 1–10.

[9] Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple Contrastive Learning of Sentence Embeddings. *arXiv preprint arXiv:2104.08821* (2021).

[10] Saurabh Gupta, Xing Fan, Derek Liu, Benjamin Yao, Yuan Ling, Kun Zhou, Tuan-Hung Pham, and Chenlei Guo. 2021. RoBERTaIQ: An Efficient Framework for Automatic Interaction Quality Estimation of Dialogue Systems. In *2nd International Workshop on Data-Efficient Machine Learning (DeMaL)*.

[11] 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*. 1443–1452.

[12] Ayyoob Imani, Amir Vakili, Ali Montazer, and Azadeh Shakery. 2019. Deep neural networks for query expansion using word embeddings. In *European Conference on Information Retrieval*. Springer, 203–210.

[13] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* (1972).

[14] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461* (2019).

[15] 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).

[16] Antonio Mallia, Omar Khattab, Torsten Suel, and Nicola Tonellotto. 2021. Learning passage impacts for inverted indexes. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1723–1727.

[17] Agnès Mustar, Sylvain Lamprier, and Benjamin Piwowarski. 2020. Using BERT and BART for Query Suggestion.. In *CIRCLE*.

[18] Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to docTTTTTquery. *Online preprint* (2019).

[19] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683* (2019).

[20] Haggai Roitman. 2019. Query Term Weighting based on Query Performance Prediction. *arXiv preprint arXiv:1902.10371* (2019).

[21] 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.

[22] Zhuoyi Wang, Saurabh Gupta, Jie Hao, Xing Fan, Dingcheng Li, Alexander Hanbo Li, and Chenlei Guo. 2021. Contextual Rephrase Detection for Reducing Friction in Dialogue Systems. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 1899–1905.

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

[24] Hamed Zamani, Mostafa Dehghani, W Bruce Croft, Erik Learned-Miller, and Jaap Kamps. 2018. From neural re-ranking to neural ranking: Learning a sparse representation for inverted indexing. In *Proceedings of the 27th ACM international conference on information and knowledge management*. 497–506.

[25] Yuxiang Zhang, Yaocheng Chang, Xiaoqing Liu, Sujatha Das Gollapalli, Xiaoli Li, and Chunjing Xiao. 2017. Mike: keyphrase extraction by integrating multidimensional information. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1349–1358.

A MODEL HYPERPARAMETER

For the weight predcition model in Figure 3, Robert-Base[2] is used as the pre-trained language model. Multi-head attention [21] is used as the cross-Attention module, and we set $head = 6$ and $dropout = 0.3$ in this work. A single linear layer is used as the classifier, with a $dropout = 0.5$. Cross-Entropy is used to calculate the loss between each predicted logits and true labels. AdamW is used as the optimizer, with $learningrate = 3e-5$ and $eps = 1e-8$. Training epoch is set as 20, and an early stopping mechanism is applied on the validation step to select the best model.

B EVALUATION METHODS DETAILS

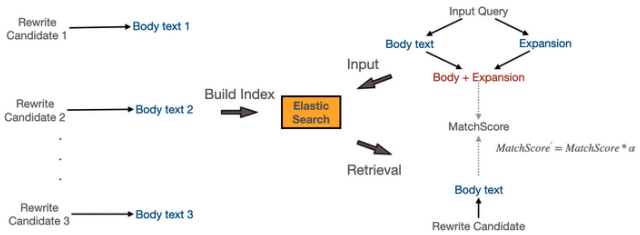

Figure 4: Illustration of using ElasticSearch to measure the model's performance.

Figure 4 demonstrates how we use the ElasticSearch to evaluate the retrieval performance. Each reformulation candidate's body text is input to the ElasticSearch to build the index; given an input query, it will be first concatenated with the expansion to form a new "query + expansion", and this "query + expansion" will be used to retrieval the relevant reformulation candidates. The matching score is calculated based on BM25. After the retrieval, each retrieved reformulation candidate's matching score will be adjusted according to the predicted weight from model 3: if the candidate contains the entity that is predicted as "2", the candidate's retrieval score will be multiplied by a constant α. We set $\alpha = 1.5$ in this work.

The SimCSE method is a contrastive learning based method. A Roberta-base model is trained using Alexa labelled data with the objective to minimize the distance between positive query-reformulation pairs and maximize the distance between negative pairs. After the training, the model can be used to encode both the query and reformulation candidates to embeddings. We then used the FAISS [3] to retrieve the most similar reformulation embeddings for each query embedding. Similarly, we adjust each retrieval's score by dividing the original matching distance by $\alpha = 1.2$ if this reformulation contains an important entity.

[2]https://huggingface.co/roberta-base
[3]https://github.com/facebookresearch/faiss