Foresight: Generating Masked Response as Query for Knowledge Selection

Kavin R V
Indian Institute of Science Education and Research
Bhopal, India
kavin19@iiserb.ac.in

Suvodip Dey
Indian Institute of Technology
Hyderabad, India
cs19resch01003@iith.ac.in

Maunendra Sankar Desarkar
Indian Institute of Technology
Hyderabad, India
maunendra@cse.iith.ac.in

Abstract

Recent works in knowledge intensive NLP tasks have demonstrated that the integration of re-ranker components as a pivotal technique for enhancing performance, especially for conversation based tasks. Building upon this trajectory of research, we present "ForeSight," a specialized re-ranker tailored for dialogue systems. ForeSight employs a novel approach by utilizing a masked response as the query for re-ranking. The re-ranker model is optimized through a process of self-distillation, utilizing keyword estimation likelihood objective as teacher for the primary ranking objective. To evaluate the efficacy of ForeSight, we conduct experiments on a knowledge-grounded conversation dataset known as Wizard of Wikipedia. Our findings exhibit a notable improvement in re-ranking performance when employing a masked response as opposed to using only the last utterance.

CCS Concepts: • **Natural Language Processing** → **Dialogue Systems**; • **Information Systems** → *Re-Ranking*.

Keywords: NLP, transformers, dialogue systems, re-ranking, keywords

1 Introduction and Background

With recent development in Large Language models have shown that they store huge amount of world knowledge within their parameters, but they are known to suffer from producing statement with factual errors or hallucinations. So there is recent trend of relying on the external knowledge like wikipedia to facilitate knowledge intensive conversations. Retrieval Augmented model like RAG, REALM, FiD etc relies heavily on the output of retrieval. Re ranking ranking helps to substantially reduce reliance for accurate generation.

1.1 Passage Ranking

Typically ranking model use cross attention based model where input is a sequence of concatenated query and candidate passage and fed into the model. This model stucture is mostly only explored for BERT-like encoder only models. RankT5 [9] introduces a different approach that out performs previous approaches. Recent State-of-Art Conversation model like GripRank[1], QKConv [4] have shown adding a Re-Rank component to the model significantly boosts the performace of these model, improves the quality of dialogue generation. Also most of the previous works have only used the previous conversation as query for the re-ranking/retrieval, but using them as a query would bring some complications like for example the conversation with a several topics could confuse such models, also using only the last conversation won't be helpfull when the question is based on some statement already made deep in conversation.

1.2 Knowledge Distillation

Recent models for Knowledge intensive task like FiD[7], HindgeSight, GripRank[1], RocketQA, has shown that distilling knowledge from Reader model to revtriever model significantly helps to boost their performance. Different from these model we introduce a way to utilize self distillation specific for conversation models.

2 Methodology

2.1 Problem Formulation

In our methodology, we address the task of re-ranking a list of passages to improve their ordering based on their relevance to a given query. Specifically, we are provided with a set of retrieved passages denoted as $P = \{p_1, p_2, \ldots, p_k\}$, where each p_i represents a passage, and k signifies the number of retrieved passages. These passages are associated with a given query denoted as q. The objective of the re-ranker is to generate a re-ordered list of passages denoted as $P' = \{p'_1, p'_2, \ldots, p'_k\}$, such that the new order better reflects the relevance of passages to the query. To accomplish this task, we introduce a re-ranker denoted as $\mathcal{R}(P'|q, P; \theta_{\mathcal{R}})$, which is parameterized by $\theta_{\mathcal{R}}$. The re-ranker takes as input the query q, the original list of retrieved passages P, and the parameterization $\theta_{\mathcal{R}}$.

2.2 Masked Response as Query

In this section, we introduce our novel approach of employing a masked response as the query for the re-ranking process. The rationale behind this technique is to enhance the re-ranking process's efficacy by focusing on key information within the conversation. To implement this approach, we follow a multi-step procedure.

First, we initiate the creation of the masked response by utilizing an unsupervised keyword extractor called YAKE

[5]. The output of this step is a set of extracted keywords denoted as $K = \{k_1, \ldots, k_n\}$, extracted from the original response. For the masking process we consider the history of previous utterances denoted as $U = \{u_1, \ldots, u_t\}$, along with the relevant passage p_r. The masking process involves determining whether a particular keyword k_x should be masked in the response $r = u_{t+1}$. This decision is based on the following criteria: if the keyword k_x is absent in all previous utterances u_y within U, and if the keyword k_x is present in the relevant passage p_r. If both conditions are met, the keyword k_x is masked in the response r.

The result of this masking procedure is the creation of a masked response, denoted as r_{masked}. This masked response retains key contextual information while focusing on elements that have not been extensively covered in prior dialogue. For the re-ranking task, this masked response r_{masked} is combined with the last utterance u_t to form an effective query denoted as $q = \{u_t, r_{\text{masked}}\}$.

2.3 Passage Ranker

In our ranking methodology, we employ an architecture inspired by RankT5 [9] to effectively rank passages based on their relevance to given queries. This architecture, compared to the traditional Cross-Encoder model used for re-ranking, has demonstrated significant performance improvements, along with the added capability of generation, which is a key aspect discussed in Section 2.4

For a given i-th query q_i and it's j-th retrieved passage p_{ij}, the input representation s_{ij} is structured as follows:

$$s_{ij} = \text{question: } q_i \text{ context: } p_{ij} \tag{1}$$

This architecture can be viewed as a simple adaptation of the T5 model [8], focusing on the first output token from the decoder. By passing the input through the T5 model, we obtain unnormalized logits z across the entire vocabulary:

$$z = \text{RankHead}(\text{Dec}(\text{Enc}(s_{ij}))) \tag{2}$$

Here, RankHead is a dense layer that projects the output hidden state to the vocabulary space $\mathcal{V}$. Notably, the absence of a softmax layer over the vocabulary results in arbitrary real numbers in the logits z. Additionally, since our objective is ranking and not token generation, the need for previous tokens $t_{1:k-1}$ is eliminated.

To determine the ranking score for a given pair of query q_i and passage p_{ij}, we specify an unused token in the T5 vocabulary, denoted as "<extra_id_80>". The corresponding unnormalized logits for this token are taken as the ranking score:

$$\hat{y}_{ij} = z_{\text{<extra\_id\_80>}} \tag{3}$$

Here, $z_{\{w\}}$ represents the logits corresponding to the token $w \in \mathcal{V}$. For this objective we use a listwise softmax cross entropy loss[2, 3] based on the predicted scores $\hat{y}_i$ and relevance labels y_i.

$$l_{\text{ranking}}(y_i, \hat{y}_i) = -\sum_{j=1}^{m} y_{ij} \log\left(\frac{e^{\hat{y}_{ij}}}{\sum_{j'} e^{\hat{y}_{ij'}}}\right) \tag{4}$$

2.4 Keyword Estimation and Self Distillation

For knowledge distillation we take logit score of all the keywords that are masked in response from the decoder output. So for t-th keyword k_t we calculate keyword estimation score $\hat{z}^t$ of t-th token

$$\hat{z}^t = \text{LMHead}(\text{Dec}(\text{Enc}(s_{ij}), k_{1:t-1})) \tag{5}$$

Notice, we again only obtain unnormalized logits $\hat{z}^t$ across the entire vocabulary. To get the estimation score of keyword k_t

$$g_{ij}^{k_t} = \hat{z}^t_{\{k_t\}} \tag{6}$$

And we to obtain the overall keyword estimation score of a passage we sum g_{ij}^{kt} over all the keywords.

$$g_{ij} = \sum_t g_{ij}^{k_t} \tag{7}$$

Based on this we experiment with two different losses, listwise softmax cross entropy loss and KL divergence loss.

$$l_{\text{KE}}(y_i, g_i) = -\sum_{j=1}^{m} y_{ij} \log\left(\frac{e^{g_{ij}}}{\sum_{j'} e^{g_{ij'}}}\right) \tag{8}$$

and

$$l_{\text{KLKE}}(y_i, \hat{y}_i, g_i) = \text{KLDiv}(y_i || g_i) \tag{9}$$

we train the model two losses, $l_1 = l_{\text{ranking}}(y_i, \hat{y}_i) + l_{\text{KE}}(y_i, g_i)$ and $l_2 = l_{\text{ranking}}(y_i, \hat{y}_i) + l_{\text{KLKE}}(y_i, \hat{y}_i, g_i)$

3 Experimental Setup

3.1 Dataset

We use Wizard of Wikipedia [6] dataset for all our experiments. In training split of this data contains around 18000 conversations between wizard and apprentice. Each turn contains a knowledge pool containing 7 passage that was retrieved based on the previous conversation and it also contains the label for the correct passage only for wizards turns/responses. So after cleaning the data and restructuring it to contain all the response from wizards as output we had dataset with around 74000 training examples.

3.2 Implementation Details

We Initialize all our model with T5-base Checkpoint. For all the models we set the token size as 512. And they are all trained with batch size of 4 and gradient accumulation steps of 8, so an effective batch size of 32 on a single V100 32GB GPU for 2 epochs.

Table 1. Re-ranking performance using query as last utterance and masked response compared to only using last utterance

	MRR	Recall@			NDCG@		
		1	2	3	1	2	3
RankT5 Last utterance only	88.57	81.12	91.59	95.29	81.12	87.72	89.58
RankT5 Without KE	94.79	90.94	97.09	98.39	90.94	94.82	95.47
RankT5 With KE	94.79	91.00	96.93	98.39	91.00	94.74	95.47
RankT5 With KLKE	94.96	91.29	96.98	98.56	91.29	94.88	95.67

Table 2. Re-ranking performance using query as last utterance and generated masked response compared to only using last utterance

	MRR	Recall@			NDCG@		
		1	2	3	1	2	3
RankT5 Last utterance only	88.57	81.12	91.59	95.29	81.12	87.72	89.58
RankT5 Without KE	85.29	76.14	88.08	93.33	76.14	83.68	86.30
RankT5 With KE	85.65	76.41	89.04	93.99	76.41	84.38	86.85
RankT5 With KLKE	86.56	77.97	89.80	93.88	77.97	85.43	87.47

4 Results

To compare the proposed model we train RankT5 using only the last utterance as query. We also generate masked response given the previous conversation, using a simple conditional generation using T5. The result from this generated response is shown in Table 2. From Table 1 we can see using masked response significantly improves the performance. Also while not helpful in Table1 Keyword Estimation seems to produce better result when using the generated masked response when compared to one without any KE loss. All the metrics are calculated using Torchmetrics.

5 Conclusion

We can see based on the results produced using the actual masked response as query gives us significant performance compared to when using only the previous utterance. We can also see quite a bit of performance improvement when we use Keyword Estimation Loss, More so when using it in KL divergence loss. But there is significant decrease in performance when using the generated masked response from a naive T5 generation. The future work would focus on bring the re-ranking performance of using generated masked response as query close to when using the actual masked response as query. We also would like to focus bringing this keyword estimation based distillation to retriever as well.

References

[1] Jiaqi Bai, Hongcheng Guo, Jiaheng Liu, Jian Yang, Xinnian Liang, Zhao Yan, and Zhoujun Li. 2023. GripRank: Bridging the Gap between Retrieval and Generation via the Generative Knowledge Improved Passage Ranking. *arXiv preprint arXiv:2305.18144* (2023).

[2] Sebastian Bruch, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2019. An Analysis of the Softmax Cross Entropy Loss for Learning-to-Rank with Binary Relevance. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval* (Santa Clara, CA, USA) *(ICTIR '19)*. Association for Computing Machinery, New York, NY, USA, 75–78. https://doi.org/10.1145/3341981.3344221

[3] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*. 89–96.

[4] Mingzhu Cai, Siqi Bao, Xin Tian, Huang He, Fan Wang, and Hua Wu. 2022. Query Enhanced Knowledge-Intensive Conversation via Unsupervised Joint Modeling. *arXiv preprint arXiv:2212.09588* (2022).

[5] Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. YAKE! Keyword Extraction from Single Documents Using Multiple Local Features. *Inf. Sci.* 509, C (jan 2020), 257–289. https://doi.org/10.1016/j.ins.2019.09.013

[6] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241* (2018).

[7] Gautier Izacard and Edouard Grave. 2020. Distilling knowledge from reader to retriever for question answering. *arXiv preprint arXiv:2012.04584* (2020).

[8] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research* 21, 1 (2020), 5485–5551.

[9] Honglei Zhuang, Zhen Qin, Rolf Jagerman, Kai Hui, Ji Ma, Jing Lu, Jianmo Ni, Xuanhui Wang, and Michael Bendersky. 2023. Rankt5: Fine-tuning T5 for text ranking with ranking losses. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2308–2313.