Distilling Knowledge for Fast Retrieval-based Chat-bots

Amir Vakili Tahami
a_vakili@ut.ac.ir
University of Tehran

Kamyar Ghajar
k.ghajar@ut.ac.ir
University of Tehran

Azadeh Shakery
shakery@ut.ac.ir
University of Tehran

ABSTRACT

Response retrieval is a subset of neural ranking in which a model selects a suitable response from a set of candidates given a conversation history. Retrieval-based chat-bots are typically employed in information seeking conversational systems such as customer support agents. In order to make pairwise comparisons between a conversation history and a candidate response, two approaches are common: cross-encoders performing full self-attention over the pair and bi-encoders encoding the pair separately. The former gives better prediction quality but is too slow for practical use. In this paper, we propose a new cross-encoder architecture and transfer knowledge from this model to a bi-encoder model using distillation. This effectively boosts bi-encoder performance at no cost during inference time. We perform a detailed analysis of this approach on three response retrieval datasets.

CCS CONCEPTS

• **Information systems** → *Retrieval models and ranking*; • **Computing methodologies** → *Natural language processing*.

KEYWORDS

Retrieval-based chat-bot, Response ranking, Neural information retrieval

1 INTRODUCTION

Response retrieval is a subset of neural ranking in which a model selects a suitable response from a set of candidates given a conversation history. Retrieval-based chat-bots are typically employed in information seeking conversational systems such as customer support agents. They have been used in real-world products such as Microsoft XiaoIce [15] and Alibaba GroupâĂŹs AliMe Assist [9].

To find the best response to a particular conversation's chat history traditional text retrieval methods such as term frequency have proven to be insufficient [10], therefore the majority of modern research focuses on neural ranking approaches [4, 6, 10]. These methods rely on training artificial neural networks on large datasets for

the task of selecting a suitable response among a set of candidates according to a conversation history.

By pre-training large scale language models on vast corpora and subsequently fine-tuning these models on downstream tasks, researchers have achieved state-of-the-art results in a wide variety of natural language tasks [3]. This process has also been successfully applied to the task of response retrieval [4, 6, 13]. Current state-of-the-art response retrieval focuses on using these pre-trained transformer language models such as BERT [3]. When using a deep pre-trained transformer for the task of comparing two text inputs, two approaches are common: either encoding representations separately (bi-encoding) or encoding the concatenation of the two (cross-encoding). The BERT bi-encoder encodes two separate representations using pre-trained deep multi-layer transformers and compares them using a dot product operation. The BERT cross-encoder concatenates the conversation history and candidate response and encodes them into a single representation, which is fed into a fully connected network that gives a matching score. The latter method achieves better prediction quality but is far too slow for practical use [6].

While bi-encoding does give worse results, previous work has shown that one can significantly reduce its inference time by pre-encoding candidate responses offline so that during inference, only the conversation history needs to be encoded. This, in turn, means that at inference time, bi-encoders can potentially perform pairwise comparisons between a conversation history and millions of candidate responses. Such a feat is impossible to do with cross-encoders as they must recalculate encodings for each conversation history and candidate response pair. Naturally, this makes bi-encoders a desirable solution in conversational systems where real-time response selection is required [6]. Because of this improving the performance of bi-encoders is a popular avenue of research when it comes to response retrieval.

In this paper, we demonstrate one possible improvement to bi-encoders, which will boost their prediction quality without affecting their prediction speed. We propose transferring knowledge from the better performing BERT cross-encoder to the much faster BERT bi-encoder. This method will raise BERT bi-encoder prediction quality without increasing inference time. We employ knowledge distillation, which is an approach where a model teaches another model to mimic it as a student [5]. Essentially, the student model learns to reproduce the outputs of the more complex teacher model. Unlike gold labels, the output of a neural network is not constrained to a binary variable and as such it can provide a much richer signal when training the student model. Knowledge distillation has been successfully applied in natural language understanding, machine translation, and language modeling tasks [7, 16, 20].

We also introduce a new cross-encoder architecture we call the enhanced BERT cross-encoder. This architecture is specifically designed for the task of response retrieval and gives better results

Table 1: Statistics for the datasets.

	UDC			DSTC7			MANtIS		
№ of candidates	10			100			11		
	Trn	Vld	Tst	Trn	Vld	Tst	Trn	Vld	Tst
№ of samples	500k	50k	50k	100k	5k	1k	82k	18k	18k

than the regular BERT cross-encoder. It also has the advantage of being faster to train. This model serves as our teacher, and we use the BERT bi-encoder [6] as our student model. We evaluate our approach on three response retrieval data-sets. Our experiments show that our knowledge distillation approach enhances the prediction quality of BERT the bi-encoder. This increase comes to a no-cost during inference time.

2 METHOD

First, we explain the task in further detail. Next, we describe the teacher and student models used for the knowledge distillation approach. Then we describe the knowledge distillation procedure.

2.1 Task Definition

Task of response retrieval can be formalized as follows: Suppose we have a dataset $\mathcal{D} = \{c_i, r_i, y_i\}_{i=1}^{N}$ where $c_i = \{t_1, \cdots, t_m\}$ represents the conversation and $r_i = \{t_1, \cdots, t_n\}$ represents a candidate response and $y_i \in \{0, 1\}$ is a label. $y_i = 1$ means that r_i is a suitable choice for c_i. t_i are tokens extracted from text . The goal of a model should be to learn a function $g(c, r)$ that predicts the matching degree between any new conversation history c and a candidate response r. Once a given model ranks a set of candidates, its prediction quality is then measured using recall@1 (1 if the model's first choice is correct otherwise 0) and mean reciprocal rank (MRR).

2.2 Model Architecture

For the student network, we use the previously proposed BERT bi-encoder [6]. The conversation history and response candidate tokens are encoded separately using BERT. To aggregate the final layer encodings into a single vector, the first token's encoding, which corresponds to an individual [CLS] token, is selected. BERT requires all inputs to be prepended with this special token. The two aggregated vectors are compared using a dot-product operation.

Similarly, our teacher model uses a BERT transformer to encode the conversation history and candidate response. However, for comparing the last layer encodings we use a combination of scaled dot-product attention [18] and the *SubMult* function [19] for calculating the matching score. Below we give a brief explanation of these components before describing how they are used.

In an attention mechanism, each entry of a key vector $k \in \mathbb{R}^{n_k \times d}$ is weighted by an importance score defined by its similarity to each entry of query $q \in \mathbb{R}^{n_q \times d}$. For each entry of q the entries of k are then linearly combined with the weights to form a new representation. Scaled dot-product attention is a particular version of attention defined as:

$$Att(q,k) = softmax\left(\frac{q \cdot k^T}{\sqrt{d}}\right) \cdot k \qquad (1)$$

The *SubMult* function [19] is a function designed for comparing two vectors $a \in \mathbb{R}^d$ and $b \in \mathbb{R}^d$ which has been used to great effect in various text matching tasks including response retrieval [17]. It is defined as follows:

$$SubMult(a,b) = a \oplus b \oplus (a - b) \oplus (a \odot b) \qquad (2)$$

where $\oplus$ and $\odot$ are concatenation and hadamard product operators respectively.

Utilizing these components we build our enhanced cross-encoder architecture. First, like the bi-encoder, we encode the conversation history $c \in \mathbb{R}^{m \times d}$ and candidate response $r \in \mathbb{R}^{n \times d}$ as follows:

$$c' = T(c), \ r' = T(r)$$

where T is the BERT transformer and $c' \in \mathbb{R}^{m \times d}, r' \in \mathbb{R}^{n \times d}$ are the encoded tokens.

To compare the encoded conversation history c' and encoded candidate response r', first we perform a cross attention operation using the previously described components:

$$\begin{aligned} \hat{c} &= W_1 \cdot SubMult(c', Att(c', r')) \\ \hat{r} &= W_1 \cdot SubMult(r', Att(r', c')) \end{aligned} \qquad (3)$$

where $W_1 \in \mathbb{R}^{4d \times d}$ is a a learned parameter. We aggregate $\hat{c} \in \mathbb{R}^{m \times d}$ and $\hat{r} \in \mathbb{R}^{n \times d}$ by concatenating the first token (corresponding to [CLS]), the max pool and average pool over the tokens:

$$\begin{aligned} \bar{c} &= \hat{c}_1 \oplus \max_{1 \le i \le m} \hat{c}_i \oplus \underset{1 \le i \le m}{\text{mean}} \hat{c}_i \\ \bar{r} &= \hat{r}_1 \oplus \max_{1 \le i \le n} \hat{r}_i \oplus \underset{1 \le i \le n}{\text{mean}} \hat{r}_i \end{aligned} \qquad (4)$$

We compare the aggregated $\bar{c}, \bar{r} \in \mathbb{R}^d$ vectors using a final *SubMult* function and a two layer fully connected network:

$$g(c, r) = W_2(ReLU(W_3 \cdot SubMult(\bar{c}, \bar{r})))$$

where $W_2 \in \mathbb{R}^{12d \times d}, W_3 \in \mathbb{R}^{d \times 1}$ are learned parameters. Our enhanced BERT architecture essentially encodes the conversation history and candidate response tokens separately using BERT, then applies as single layer of cross-attention on those encodings.

We believe our enhanced cross-encoder architecture will perform better than regular cross-encoders for two reasons. Firstly, we do not concatenate conversation history and candidate responses. This means we can use the encoded candidate response tokens of other samples in a training batch as negative samples [11]. Scaled dot-product attention is simple enough that recalculating it for other candidates in the batch does not add significant overhead, especially when compared to rerunning BERT for every possible conversation history and candidate response pair. Thus we can process more negative samples than would be feasible in a regular cross-encoder. Previous research has already shown that increasing the number of negative samples is effective for response retrieval [6]. Secondly, the addition of the *SubMult* function means we can achieve much more refined text matching between the conversation history and candidate response.

2.3 Distillation Objective

Distillation achieves knowledge transfer at the output level. The student learns from both dataset gold labels and teacher predicted probabilities, which are also a useful source of information [1]. For example, in sentiment classification, certain sentences might have very strong or weak polarities and binary labels are not enough to convey this information.

Similar to previous work [16], we add a distillation objective to our loss function which penalizes the mean squared error loss between the student and teacher model outputs:

$$\mathcal{L}_{\text{distill}} = ||z^{(T)} - z^{(S)}||^2$$

where $z^{(T)}, z^{(S)}$ are the teacher and student model outputs. At training time the distillation objective is used in conjunction with regular cross entropy loss as follows:

$$\mathcal{L} = \alpha \cdot \mathcal{L}_{CE} + (1 - \alpha) \cdot \mathcal{L}_{\text{distill}}$$

where α is a hyper-parameter. This procedure is model agnostic and can transfer information between entirely different architectures.

3 EXPERIMENTS

In this section we give a brief overview of experiments settings.

3.1 Datasets

We consider three information-seeking conversation datasets widely used in the training of neural ranking models for response retrieval. The Ubuntu Dialogue Corpus (UDC) [10] and DSTC7 sentence selection track dataset [2] are collected form a chatroom dedicated to the support of the Ubuntu operating system. We also include a version of UDC where the training set has been reduced to 20% so as to study the effects of limited training data. MANtIS [13] was built from conversations of 14 different sites of the Stack Exchange Network. The statistics for these datasets are provided in Table 1. Data augmentation, where each conversation is split into multiple samples, is a popular method in dialog research for boosting the performance of response retrieval models. In this paper, we refrain from using this approach as our focus is not beating state-of-the-art results but empirically demonstrating the effectiveness of knowledge distillation even in limited-resource settings.

3.2 Baselines

We divide our experiments into three parts. 1. Comparing the regular BERT cross-encoder and our enhanced BERT cross-encoder. Here we aim to demonstrate the superiority of our proposed cross-encoder architecture 2. Comparing the BERT bi-encoder with and without distillation. Here we wish to demonstrate the effectiveness of the knowledge distillation approach. 3. Finally, we also train a BiLSTM bi-encoder with and without distillation in order to confirm the distillation process works with shallow student models. The BiLSTM bi-encoder uses the same tokens as BERT models, but their embeddings are not pre-trained and initialized randomly. We use the same aggregation strategy (eq. 4) to aggregate the BiLSTM hidden states. Our code will be released as open-source.

3.3 Implementation Details

Our models are implemented in the PyTorch framework [12]. For our BERT component, we used Distilbert [14] since it provides results somewhat close to the original implementation despite having only 6 layers of transformers instead of 12. We tune α from a set of $\{0.25, 0.5, 0.75\}$. We train models using Adam optimizer [8]. We use a learning rate of 5×10^{-5} for BERT models and 10^{-3} for the BiLSTM bi-encoder. For consistency, we set the batch size to 8 for all models. For each dataset, we set the maximum number of tokens in the conversation history and candidate responses so that no more than 20% of inputs are truncated.

Unfortunately, due to limited computing resources, we are unable to beat state-of-the-art results reported by [6]. Our models are trained on a single GPU; thus, we had to make compromises on the number of input tokens, number of negative samples, and model depth.

4 RESULTS AND DISCUSSION

In this section, we go over the results of our experiments. We analyze both prediction quality and efficiency.

4.1 Prediction Quality

The first two rows of table 2 demonstrate the effectiveness our the enhanced BERT cross-encoder relative to the regular BERT cross-encoder. These results indicate that employing a task-specific single layer cross-attention mechanism on top of separately encoded inputs is highly effective for the task of response retrieval. Of particular note is the increased gap between the performance of the two methods when using smaller training sets (UDC$_{20\%}$, MANtIS, DSTC7). This shows that the regular bert-cross model struggles when fine-tuned with smaller response-retrieval sets and data augmentation or a some other method must be used to achieve acceptable results. In contrast, our enhanced BERT cross-encoder's R@1 only dropped by 3.3 points when its training set was reduced to a fifth.

To further demonstrate the effectiveness of our modifications to the BERT cross-encoder architecture, we perform an ablation study on the reduced UDC dataset. We replace the *SubMult* function with a concatenation operation. We also try removing cross-attention (3). In both cases, their removal significantly degrades model quality.

Across the datasets, bi-encoders show significant gains when trained with knowledge distillation. The increase in performance is relatively substantial. Such gains usually require an increase in model complexity, however with knowledge distillation, we are effectively gaining a free boost in performance as there is no extra cost at inference time. The best results were obtained with an α of 0.5. This indicates that in response retrieval, unlike other tasks such as sentiment classification and natural language inference [16], the gold labels cannot be replaced entirely with teacher outputs.

4.2 Prediction Efficiency

We demonstrate the trade-off in speed and performance between the BERT bi-encoder and our enhanced BERT cross-encoder. We measure the time it takes to process test samples in the DSTC7

Table 2: Prediction quality metrics across all datasets. Metrics for models trained with knowledge distillation, which are significant relative to models trained without it, are marked in bold. We use paired two-tailed t-tests with a p-value<0.05 to perform significance tests. For easier reading metrics have been multiplied by 100. No data augmentation has been used and training samples are used as is. +KD indicates a model trained with knowledge distillation.

	UDC$_{20\%}$		UDC		MANtIS		DSTC7	
	R@1	MRR	R@1	MRR	R@1	MRR	R@1	MRR
BERT cross	66.1	76.8	76.5	84.8	59.8	72.0	36.9	47.9
BERT cross enhanced	**76.2**	**84.5**	**79.5**	**86.9**	**66.7**	**77.3**	**53.3**	**63.3**
- SubMult	73.4	82.6	—	—	—	—	—	—
- Attention	67.2	78.6	—	—	—	—	—	—
BiLSTM bi-encoder	59.2	72.4	69.4	80.2	35.6	55.1	34.3	46.1
BiLSTM bi-encoder + KD	**63.0**	**75.2**	**70.4**	80.8	**45.5**	**61.4**	**39.4**	**50.1**
BERT bi-encoder	64.9	76.9	72.9	82.7	47.9	58.4	39.9	51.8
BERT bi-encoder + KD	**66.1**	**77.6**	**75.8**	**84.6**	**53.4**	**67.3**	**53.8**	**54.7**

Table 3: Average milliseconds to process a single test sample.

№ of candidates	10	100
BERT bi-encoder	5.6	6.2
BERT cross-encoder enhanced	81.1	981.2

dataset and show the average time for each example in table 3. Time taken by the cross-encoder to process a set of candidate responses grows exponentially large as the set increases in size. In the case of BERT bi-encoders, since candidate vectors can be computed offline, increasing candidates has a negligible impact on inference time.

5 CONCLUSION AND FUTURE WORK

In this paper, we introduced an enhanced BERT cross-encoder architecture modified for the task of response retrieval. Alongside that, we utilized knowledge distillation to compress the complex BERT cross-encoder network as a teacher model into the student BERT bi-encoder model. This increases the BERT bi-encoders prediction quality without affecting its inference speed. We evaluate our approach on three domain-popular datasets. The proposed methods were shown to achieve statistically significant gains.

One possible avenue for research is the exploration of other knowledge transfer methods. Substituting the relatively simple BERT bi-encoder architecture with a more complex architecture [4] or developing further improvements to the BERT cross-encoder are also viable alternatives.

REFERENCES

[1] Jimmy Ba and Rich Caruana. 2014. Do deep nets really need to be deep?. In *Advances in neural information processing systems*.

[2] Lazaros Polymenakos Chulaka Gunasekara, Jonathan K. Kummerfeld and Walter S. Lasecki. 2019. DSTC7 Task 1: Noetic End-to-End Response Selection. In *7th Edition of the Dialog System Technology Challenges at AAAI 2019*.

[3] 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*.

[4] Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Ivan Vulić, et al. 2019. ConveRT: Efficient and Accurate Conversational Representations from Transformers. *arXiv preprint arXiv:1911.03688* (2019).

[5] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

[6] Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2020. Poly-encoders: architectures and pre-training strategies for fast and accurate multi-sentence scoring. In *8th International Conference on Learning Representations, ICLR 2020*.

[7] Yoon Kim and Alexander M Rush. 2016. Sequence-Level Knowledge Distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

[8] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[9] Feng-Lin Li, Minghui Qiu, Haiqing Chen, Xiongwei Wang, Xing Gao, Jun Huang, Juwei Ren, Zhongzhou Zhao, Weipeng Zhao, Lei Wang, Guwei Jin, and Wei Chu. 2017. *AliMe Assist* : An Intelligent Assistant for Creating an Innovative E-commerce Experience. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017*.

[10] Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*.

[11] Pierre-Emmanuel Mazaré, Samuel Humeau, Martin Raison, and Antoine Bordes. 2018. Training Millions of Personalized Dialogue Agents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*.

[12] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic Differentiation in PyTorch. In *NIPS Autodiff Workshop*.

[13] Gustavo Penha and Claudia Hauff. 2020. Curriculum Learning Strategies for IR: An Empirical Study on Conversation Response Ranking. In *European Conference on Information Retrieval*. Springer.

[14] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019).

[15] Heung-Yeung Shum, Xiao-dong He, and Di Li. 2018. From Eliza to XiaoIce: challenges and opportunities with social chatbots. *Frontiers of Information Technology & Electronic Engineering* (2018).

[16] Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from BERT into simple neural networks. *arXiv preprint arXiv:1903.12136* (2019).

[17] Chongyang Tao, Wei Wu, Can Xu, Wenpeng Hu, Dongyan Zhao, and Rui Yan. 2019. Multi-Representation Fusion Network for Multi-Turn Response Selection in Retrieval-Based Chatbots. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*.

[18] 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*.

[19] Shuohang Wang and Jing Jiang. 2017. A Compare-Aggregate Model for Matching Text Sequences. In *5th International Conference on Learning Representations,*

ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings.

[20] Seunghak Yu, Nilesh Kulkarni, Haejun Lee, and Jihie Kim. 2018. On-device neural language model based word prediction. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations.*