Uni-Encoder: A Fast and Accurate Response Selection Paradigm for Generation-Based Dialogue Systems

Chiyu Song[*] [1,2], **Hongliang He**[*] [1,2], **Haofei Yu**[3], **Pengfei Fang**[4,2], **Leyang Cui**[5], **Zhenzhong Lan**[†] [2]
[1]Zhejiang University, [2]School of Engineering, Westlake University
[3]Language Technologies Institute, Carnegie Mellon University, [4]Southeast University, [5]Tencent AI Lab
{songchiyu, hehongliang, lanzhenzhong}@westlake.edu.cn
haofeiy@cs.cmu.edu, fangpengfei@seu.edu.cn, leyangcui@tencent.com

Abstract

Sample-and-rank is a key decoding strategy for modern generation-based dialogue systems. It helps achieve diverse and high-quality responses by selecting an answer from a small pool of generated candidates. The current state-of-the-art ranking methods mainly use an encoding paradigm called *Cross-Encoder*, which separately encodes each context-candidate pair and ranks the candidates according to their fitness scores. However, *Cross-Encoder* repeatedly encodes the same lengthy context for each candidate, resulting in high computational costs. *Poly-Encoder* addresses the above problems by reducing the interaction between context and candidates, but with a price of performance drop. In this work, we develop a new paradigm called *Uni-Encoder*[1], that keeps the full attention over each pair as in *Cross-Encoder* while only encoding the context once, as in *Poly-Encoder*. *Uni-Encoder* encodes all the candidates with the context in one forward pass. We use the same positional embedding for all candidates to ensure they are treated equally and design a new attention mechanism to avoid confusion. Our *Uni-Encoder* can simulate other ranking paradigms using different attention and response concatenation methods. Extensive experiments show that our proposed paradigm achieves new state-of-the-art results on four benchmark datasets with high computational efficiency. For instance, it improves $R_{10}@1$ by 2.9% with an approximately $4\times$ faster inference speed on the Ubuntu V2 dataset.

Paradigm	Context-Response Full Attention	Avoidance of Context Recomputation	Performance
Bi-Encoder	✗	✓	80.6%
Cross-Encoder	✓	✗	82.8%
Poly-Encoder	✗	✓	80.9%
Uni-Encoder (Ours)	✓	✓	**85.9%**

Table 1: *Uni-Encoder* maintains the full attention between context and candidates while only encoding the lengthy context once. It is both fast and accurate compared with existing paradigms. Performance is the $R@1$ values evaluated on the Ubuntu Dialogue Corpus V2, and we refer to Humeau et al. (2019) for the results of *Bi-*, *Cross-*, and *Poly-Encoder*. The pre-trained BERT weights are all from Devlin et al. (2019).

1 Introduction

One of the major milestones of artificial intelligence is the ability to converse freely in natural language. Researchers in this field are working on building open-domain dialogue systems capable of handling a variety of topics. Depending on the implementation, these works can be categorized as **retrieval-based** (Lowe et al., 2015; Tao et al., 2019; Yuan et al., 2019) or **generation-based** (Vinyals and Le, 2015; Serban et al., 2016). Retrieval-based systems carry out conversations by selecting an optimal response from a **large** candidate pool, which shows advantages in producing fluency and relevant response. However, retrieval-based systems may be limited by the capacity of the pre-defined candidate pool. Generation-based systems generate reasonable responses by a sequence-to-sequence model. Previous work shows that generation-based systems tend to give repetition or contradictory responses (Nie et al., 2021; Cui et al., 2022).

To combine the advantage of both methods, Adiwardana et al. (2020) proposed a “sample-and-rank” method, which first samples a **small** pool of candidate responses from the generator and then reranks the candidates to get the best response by a ranker. Because a ranking model can view the whole responses while a pure generation method can only generate answers based on partial information, sample-and-rank method often performs better than the pure sample method. Under the sample-and-rank framework, researchers have greater free-

[*] Equal contribution.
[†] Corresponding author.
[1] Uni means one or unified here.

dom to explore different ranking methods (Zhang et al., 2020; Roller et al., 2021; Bao et al., 2021; Thoppilan et al., 2022). They can encode candidates on-the-fly and encode them with the context. *Cross-Encoder* (Urbanek et al., 2019) is one such paradigm. It jointly encodes the historical context with every candidate using full attention and ranks them according to the context-candidate matching scores. Despite its superior performance, *Cross-Encoder* repeatedly encodes the context for each candidate. Since contexts are often much longer than responses, the computation is slow for practical use. *Poly-Encoder* (Humeau et al., 2019; Roller et al., 2021) mitigates the above problem by reducing the full attention at every layer of Transformer (Vaswani et al., 2017) to global attention at the last layer. However, later work (Gu et al., 2020, 2021; Han et al., 2021) confirms the importance of full attention and still uses *Cross-Encoder* as the base building block for response selection.

One interesting research question is whether there is a way to realize full attention between each context-response pair without repeatedly encoding the same long context. To answer the above question, we proposed a new paradigm called *Uni-Encoder*, as presented in Table 1. In this new paradigm, all the candidates are concatenated with the context and jointly input to the same encoder in one forward pass. In the end, a softmax classifier is used to decide which candidate needs to be selected. If we concatenate candidates and context, we will get two problems. First, it is challenging to learn a good set of representations for candidates as they have different positional embeddings. Second, the averaging effect of the attention mechanism makes it difficult to distinguish various candidates. To address the above two problems, we propose two modifications to the traditional encoder networks.

First, we use the same set of positional embeddings for all candidates so that they are all treated equally because each is a possible continuation of the given context.

Second, we also design a novel attention mechanism for our new paradigm that only allows context-candidate attention and forbids the candidates to attend to each other directly.

Through changing these two designs, *Uni-Encoder* can simulate the effects of any other paradigm (*Cross-*, *Bi-* or *Poly-Encoder*) by changing how context and candidate attend to each other and how many candidates are processed in a single forward pass.

We evaluate our new paradigm on four benchmark datasets: PersonaChat (Zhang et al., 2018), Ubuntu Dialogue Corpus V1 (Lowe et al., 2015), Ubuntu Dialogue Corpus V2 (Lowe et al., 2017), and Douban Conversation Corpus (Wu et al., 2017). Empirical results show that our method achieves state-of-the-art performance, jointly with high computational efficiency. For instance, our *Uni-Encoder* has an absolute 2.9% $R@1$ improvement over the state-of-the-art *Cross-Encoder* on the widely used Ubuntu Dialogue Corpus V2 dataset. It also has a lower computational cost than *Cross-Encoder* and is approximately four times faster at inference time.

Our source code and model checkpoints will be released for reproducibility and future research[2].

2 Related Work

Neural approaches for open-domain dialogue have seen significant recent progress. Due to this progress, generation-based dialogue systems have started outperforming retrieval-based methods (Roller et al., 2021) as they can handle a wider variety of topics. Adiwardana et al. (2020) show that sample-and-rank provides much more diverse and content-rich responses than beam-search. An additional ranking step allows responses to have full attention/view over themselves and the context, while pure generation methods only have left attention/view. This different view is why an additional ranking process is needed. In this study, we particularly focus on improving this ranking process.

Because scoring candidates given a context is a classical problem in machine learning, numerous methods (Urbanek et al., 2019; Reimers and Gurevych, 2019; Adiwardana et al., 2020) have been developed over the years. We will only discuss a few closely related works. Please refer to Humeau et al. (2019) for a more detailed discussion.

Bi-Encoder (Reimers and Gurevych, 2019) encodes the context and the candidate separately, then scores the relatedness between their representations. Due to its simplicity and efficiency, *Bi-Encoder* often serves as a baseline method when a new dataset introduces (Lowe et al., 2015; Dinan et al., 2019). One significant advantage of the *Bi-Encoder* is that its response representations can

[2] https://github.com/dll-wu/Uni-Encoder

be pre-computed as they are context-independent. However, in modern generation-based dialogue systems, this advantage becomes a weakness. It is not necessary to pre-encode responses that are generated on-the-fly. And without context-response interaction, the ranking performance is severely weakened. *Poly-Encoder* (Humeau et al., 2019) improves the accuracy of the *Bi-Encoder* by adding a learned self-attention layer on top of the context and candidate features extracted from both encoders. Nevertheless, *Cross-Encoder* is preferable to generation-based dialogues systems in practice due to its high effectiveness (Urbanek et al., 2019; Humeau et al., 2019). Instead of encoding each context and response pair separately, they encode them jointly using a full attention mechanism.

Recent improvements in response selection are mostly on *Cross-Encoder*. For example, Li et al. (2021) adapt contrastive learning to *Cross-Encoder* with a specially designed strategy and obtain a significant performance gain. Lu et al. (2020) and Gu et al. (2020) add speaker change information to the inputs showing a large improvement in the response selection task. Whang et al. (2020) and Han et al. (2021) further post-train the encoder on domain-specific data and see additional improvements. To further utilize target data, Xu et al. (2021) and Whang et al. (2021) investigate some additional self-supervised learning tasks. These tasks served as additional objectives jointly trained with the response selection task. Unlike all the above improvements, our improvement is on the encoder itself and can incorporate these additional tricks.

3 Methods

This section elaborates on the problem formulation of dialogue response selection, compares different paradigms to model this task, and describes our implementation of *Uni-Encoder*.

3.1 Problem Formulation

Re-ranking methods formulate the multi-turn response selection as a set of binary classification tasks.

In practice, given a dialogue context $C = \{u_1, u_2, ..., u_N\}$, where $u_k, k = 1, \dots, N$ denotes a single utterance from either speaker, the response selection task is required to choose an optimal response from a candidate pool, denoted by $P = \{r_1, r_2, ..., r_M\}$. Every candidate r_i is respectively paired with the context C, denoted as $f(C, r_i)$. The encoding function f yields a representation that later undergoes non-linear transformations to predict a value of 1 for a proper match and 0 otherwise.

However, this binary classification view is not an efficient way of training the encoder because we need to encode the context C once for each pair of context-response comparisons. Instead, Humeau et al. (2019) leveraged in-batch negative training and viewed this task as a multi-choice selection problem. This formulation optimizes, e.g., $softmax(f(C) \cdot f(r_1), ..., f(C) \cdot f(r_M))$ by a ground truth label that is one-hot on the index of the sole positive candidate.

3.2 Task Modeling Paradigms

In the following, we reuse the same set of notations in Section 3.1. Accordingly, *Bi-*, *Poly-*, *Cross-*, and *Uni-Encoder* model the response selection task as follows.

For *Bi-Encoder*, selecting the proper response r is picking the candidate that has the highest dot product with the context:

$$f(C) \cdot f(r_1), ..., f(C) \cdot f(r_M) \tag{1}$$

where the response encoding is independent of the context encoding. Humeau et al. (2019) show that, under the multi-choice view, the larger the M is, the better the results are.

Poly-Encoder is a variant of *Bi-Encoder*. The only difference is that it adds an additional lightweight attention layer:

$$g(f(C), f(r_1)), ..., g(f(C), f(r_M)) \tag{2}$$

where g is the light-weight attention component over the context and response representations generated by encoder f.

Cross-Encoder has full attention between the context and responses. However, it has difficulty in taking the multi-choice view because it needs to recompute the context for each candidate, which can result in a memory explosion. That is, for *Cross-Encoder*, each context and response pair needs to go through the network f together:

$$f(C, r_1), ..., f(C, r_M) \tag{3}$$

In this way, for a batch containing K context-response pairs, the heavy encoder f needs to encode K^2 times, both computationally and memory intensive.

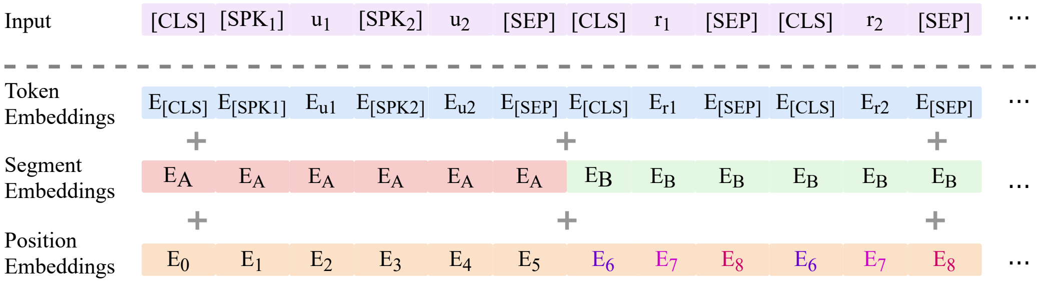

Input
[CLS] [SPK1] u1 [SPK2] u2 [SEP] [CLS] r1 [SEP] [CLS] r2 [SEP] ...
Token Embeddings
E[CLS] E[SPK1] Eu1 E[SPK2] Eu2 E[SEP] E[CLS] Er1 E[SEP] E[CLS] Er2 E[SEP] ...
Segment Embeddings
EA EA EA EA EA EA EB EB EB EB EB EB ...
Position Embeddings
E0 E1 E2 E3 E4 E5 E6 E7 E8 E6 E7 E8 ...

Figure 1: Input embeddings of the *Uni-Encoder*. The positional embeddings of responses are repeated because each candidate is a possible continuation of the given context and should be treated equally. However, this new design will cause confusion among candidates. We address this problem by designing a new attention mechanism.

Uni-Encoder also has full attention between the context and responses. Since all the candidate responses are concatenated and jointly encoded with the context in one forward pass, it naturally integrates the multi-choice view. Then the representation of each response is aggregated, and the most confident candidate is selected after feeding them into a softmax function:

$$softmax(f(C, r_1, ..., r_M)) \tag{4}$$

Comparing formulas 1 to 4, we can see that *Bi-Encoder* has no interaction between context and responses in the encoding process; *Poly-Encoder* allows partial interaction through a light-weight attention component; both *Cross-* and *Uni-Encoder* allow full interaction. Meanwhile, *Uni-Encoder* avoids the drawback of *Cross-Encoder* that repeatedly encodes the same lengthy context. Additionally, it establishes an exchange of information between candidates during the encoding process.

3.3 Inputs to the Ranking Models: Same Positional Embedding for All Responses

We take the pre-trained BERT (Devlin et al., 2019) as our encoder. As illustrated in Fig. 1, the inputs to the BERT encoder consist of three components: the token embeddings, the segment embeddings help to distinguish between context and candidates, and the positional embeddings. In our setting, the positional embeddings for all the responses (E_6 to E_8 in Fig.1) are repeated, treating each candidate as a coequal because they are all possible continuations of the context. We also have a separate speaker token for each utterance in the context to tell the model who is speaking. A `[CLS]` and a `[SEP]` token are placed before and after each candidate separately.

3.4 Attention Mechanisms: An Unified Ranking Framework

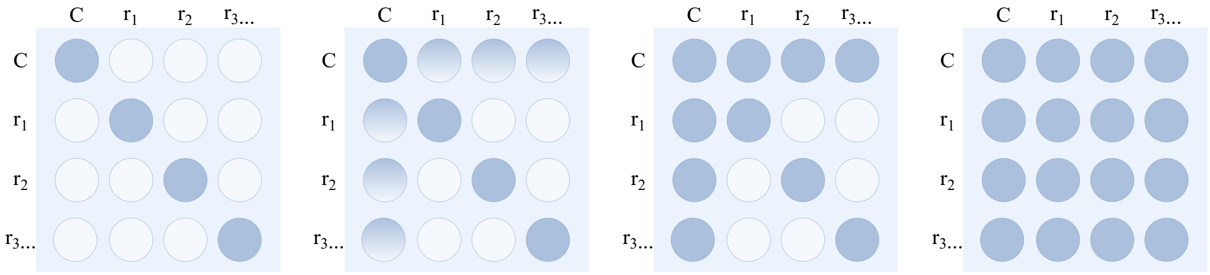
As Shown in Fig. 2, we design a new attention mechanism called Arrow Attention for *Uni-Encoder*. Arrow Attention allows full attention between context and candidates while forbidding candidates from directly attending to each other. It realizes parallel processing of multiple candidates while only needing to process the context once.

Fig. 2 also shows that *Uni-Encoder* can simulate other popular ranking frameworks by using different attention mechanisms. Specifically, (a) our work is equivalent to *Bi-Encoder* if the Diagonal Attention is used instead, where the context and the candidates do not attend to each other. (b) The Light-Arrow Attention corresponds to *Poly-Encoder*, where the context and candidates interact only at the last encoder layer through some additional light-weight attention. And the response representations are only available at the global feature level, e.g., the `[CLS]` head or average token embedding. (c) The Arrow attention is tailored for *Uni-Encoder*, where the context and the candidates have full attention, but the candidates do not attend to each other. (d) To test the extreme, we also have Square Attention, where all the context and responses attend to each other. However, it brings confusion among candidates as they share the same set of positional embeddings. The position confusion problem is addressed if it only processes one candidate at a time, which is equivalent to *Cross-Encoder* by doing so.

4 Experiments

4.1 Experimental Setup

We initialize our implementation with the BERT (Devlin et al., 2019) checkpoint provided by the



(a) Diagonal (for Bi-Encoder)

(b) Light-Arrow (for Poly-Encoder)

(c) Arrow (for Uni-Encoder)

(d) Square (*for Cross-Encoder)

Figure 2: The context-response attention maps corresponding to four paradigms, where attention is only allowed in filled areas ●. The Arrow attention (c) is tailored for *Uni-Encoder*, which realizes full attention between context and candidates and prevents candidates from directly attending to each other. The Light-Arrow attention (b) was introduced in *Poly-Encoder* (Humeau et al., 2019), where context and candidates only have attention in the last transformer layer. Changing the attention type and candidate number in parallel computation easily converts our work to other paradigms. For example, using the Diagonal attention (a) instead would make it a *Bi-Encoder*, and *using the Square attention (d) while processing only one candidate at a time would make it a *Cross-Encoder*.

Huggingface package[3]. We also test post-training (Whang et al., 2021; Han et al., 2021) on top of pre-trained BERT when the checkpoints are available. The post-trained checkpoints are provided by Han et al. (2021). As introduced in Section 2, the post-training strategy is a common technique to adapt the general pre-trained knowledge to the target domain. In practice, it continues the models' pre-training on domain-specific texts before fine-tuning them on downstream tasks to attain better performances. All the experiments are run on six NVIDIA A100-SXM4-40GB GPUs with CUDA 11.1. We use the Noam scheduler and the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.98$, and weight decay $= 0.01$. For experiments on the Ubuntu Corpus V2, we use a peak lr of 2e-4. As we want each dataset to reach the maximum batch size in training, their learning rates are also adjusted accordingly in Section 4.4. As for the loss function, we add masked language modeling (MLM) loss on top of the classification loss with the same weight coefficients. We use the average token embedding from each candidate as the input to the softmax function. Models are all run until they converge, measured by a validation set.

4.2 Dataset and Evaluation Metrics

In this section, we evaluate the proposed *Uni-Encoder* across four standard datasets, i.e., PersonaChat (Zhang et al., 2018), Ubuntu Dialogue Corpus V1 (Lowe et al., 2015), Ubuntu Dialogue Corpus V2 (Lowe et al., 2017), and Douban Conversation Corpus (Wu et al., 2017).

PersonaChat (Zhang et al., 2018) is a crowd-sourced dataset with two-speaker talks conditioned on their given persona, containing short descriptions of characters they will imitate in the dialogue.

Ubuntu Dialogue Corpus V1 (Lowe et al., 2015) contains 1 million conversations about technical support for the Ubuntu system. We use the clean version proposed by Xu et al. (2017), which has numbers, URLs, and system paths replaced by special placeholders.

Ubuntu Dialogue Corpus V2 (Lowe et al., 2017) has several updates and bug fixes compared to V1. The major one is that the training, validation, and test sets are split into different periods. We choose this dataset to conduct a detailed study of *Uni-Encoder* as it is the only dataset that *Poly-Encoder* (Humeau et al., 2019) uses and has complete train/dev/test sets published.

Douban Conversation Corpus (Wu et al., 2017) consists of web-crawled dyadic dialogs from a Chinese social networking website called Douban. Topics in this dataset are open-domain, and all the conversations are longer than two turns. Unlike other datasets where each context only has one proper response, the test set of Douban provides multiple proper responses.

The statistics of four benchmark datasets are shown in Table 2. They vary greatly in volume, language, and topic. During training, we recycle the other labels in the same batch as negative samples instead of using the pre-defined negative candidates in each dataset. Several metrics are used to evalu-

[3] https://huggingface.co/models

Dataset		Train	Valid	Test
PersonaChat	Turns	65,719	7,801	7,512
	Positive:Negative	1:19	1:19	1:19
Ubuntu V1	Pairs	1M	0.5M	0.5M
	Positive:Negative	1:1	1:9	1:9
Ubuntu V2	Pairs	1M	195.6k	189.2k
	Positive:Negative	1:1	1:9	1:9
Douban	Pairs	1M	50k	6,670
	Positive:Negative	1:1	1:1	1.2:8.8

Table 2: Statistics of four benchmark datasets.

ate our model following previous works. We use $R_c@k$ to evaluate the model performance across four datasets. The mean reciprocal rank (MRR) metric is additionally calculated for PersonChat and Douban Conversation Corpus datasets. In the Douban Conversation Corpus, we also report the $P@1$ and mean average precision (MAP) values because it contains multiple positive candidates for a given context. It is also noted that the proportion of the positive and negative samples of the validation set is significantly different from that of the test set in the Douban Conversation Corpus. To alleviate this discrepancy, we also utilize the in-batch negative labels in the validation stage to determine an appropriate checkpoint for inference.

4.3 Validating Our Design Choices

In this section, we will validate our two design choices through a set of controlled experiments. As described in Section 3.3 and 3.4, we are able to simulate different paradigms by replacing the attention mechanism in *Uni-Encoder* with some minor modifications. We thus conduct experiments in this unified framework to control all other variables and make the fairest comparisons. Note that the *Cross-Encoder* (iii) has to repeatedly encode the same lengthy context with every candidate, resulting in high memory usage and smaller batch size (5 in our experiments). The experimental results are shown in Table 3.

Why Repeating Position ID for Responses? Let us first compare the results in Row (i) vs. Row (ii), where the only difference is that Row (i) use the same set of position IDs for all responses while Row (ii) has unique position IDs. Uni-Encoder with repeated position ID has significantly better results. This observation confirms our hypothesis that our responses should be treated equally.

Why using Full Attention Between Context and Responses? If we compare the results of Row (i) with Row (v) and Row (vi), where the main differences lie in how much attention we have between context and responses, we can see that full attention can significantly boost performance. In fact, the more interaction (attention) they have, the better results they can get. Specifically, Poly-Encoder in Row(vi) has more interaction than Bi-Encoder in Row (v), and Uni-Encoder in Row (i) has more interaction than Poly-Encoder. These comparisons validate our design choices for full attention between context and responses.

Why Avoiding Attention Among Responses? Comparing results in Row (i) and Row (iii), we can see that if we allow attention among responses, the performance drops significantly. This is easy to understand because if we allow attention among responses, it will be difficult for the ranker to distinguish them.

Why Avoiding Recomputing the Context? It is easy to understand that if we recompute the lengthy context, the computational time increases dramatically, which we will measure quantitatively in Section 4.5. Here we show another dimension of the consequence of recomputing the context. As shown in Row (iv), the repetitive computation of the context stops the *Cross-Encoder* from having a large batch size because of the memory constraint. However, a good enough batch size, hence negative samples, is important for a multi-choice setting, as examined in Humeau et al. (2019). As a result, the performance of *Cross-Encoder* (iv) is only on par with *Poly-Encoder* (vi).

4.4 Comparison with State-of-the-Art Methods

We compare *Uni-Encoder* with the existing state-of-the-art methods in Table 4. Noted that, different from the comparison in Table 3, the methods in Table 4 are not entirely comparable as they have different additional training tricks. And these tricks often have a high impact on the performance of these methods. The only message we want to deliver here is that *Uni-Encoder* can achieve state-of-the-art performance even without some of these complex training tricks.

For Ubuntu Corpus V1 and Douban Conversation Corpus, we also employ the advanced post-training model from Han et al. (2021) and list the

Paradigm	Setup	Bs per GPU	Ubuntu Corpus V2			
			R_{10}@1	R_{10}@2	R_{10}@5	MRR
(i) Uni-Encoder	Arrow Attn w/ Res Concat	8	**0.859**	**0.938**	0.990	**0.915**
(ii) Uni-Encoder w/o Repeated Position ID	Arrow Attn w/ Res Concat	8	0.837	0.933	**0.992**	0.903
(iii) Concat-Cross-Encoder	Square Attn w/ Res Concat	8	0.826	0.916	0.980	0.892
(iv) Cross-Encoder	Square Attn w/o Res Concat	5	0.844	0.930	0.987	0.905
(v) Bi-Encoder	Diagonal Attn w/ Res Concat	8	0.835	0.925	0.987	0.899
(vi) Poly-Encoder	Light-Arrow Attn (360) w/ Res Concat	8	0.844	0.929	0.989	0.906

Table 3: Comparisons between different paradigms implemented according to the setups described in Section 3.4. By replacing the attention mechanism in *Uni-Encoder*, a unified framework can simulate different paradigms, which optimally controls all other training variables for fair comparisons. Please note the *Cross-Encoder* (iii) cannot reach the same large batch size as the others as it is more memory-intensive. For *Poly-Encoder*, we choose the best setting with 360 context codes.

results separately with ♣ as it significantly affects the results and not all the methods use it.

As shown in Table 4, *Uni-Encoder* achieves the best overall performance across all four benchmarks. For example, it improves the R@1 value on PersonaChat, Ubuntu V1, and Ubuntu V2 datasets by 2.6%, 0.5%, and 2.9%, respectively.

However, *Uni-Encoder* only achieves the best results on the Douban Corpus on four of the six metrics. We conjecture that the positive example size discrepancy between the training set and test set is the reason for its poorer performance. In *Uni-Encoder*, we have chosen the multi-choice setting, assuming there is only one positive response. This setting allows us to leverage response concatenation and in-batch negative training to separate the positive sample from negative examples. However, multiple positive candidates in Douban Corpus at inference time (but not in training) break this assumption and may confuse the network. Our future study will quantify the impact of this assumption.

Uni-Encoder also outperforms some of the more complex methods that rely on expensive training tricks, such as Liu et al. (2021) adapted BiGRU to capture conversation-level representations, and Su et al. (2021) leveraged hierarchical curriculum learning in their work. These approaches typically yield better outcomes, but at the expense of increased training budgets. In contrast, *Uni-Encoder* only retains the MLM loss from pre-training and adds two extra tokens to distinguish between different speakers.

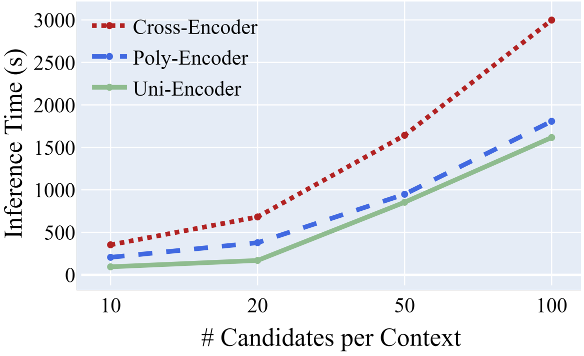


Figure 3: The inference time comparison for *Uni-Encoder* and other paradigms on the Ubuntu V2 test set. Please note that *Poly-Encoder* cannot pre-compute candidate embeddings in a generation-based dialogue system, so the results differ from those reported in Humeau et al. (2019) on retrieval tasks.

4.5 Lower Computational Cost

In addition to the accuracy gain, we also see that *Uni-Encoder* is computational efficiency compared to other paradigms. We test it on the Ubuntu V2 test set (189,200 contexts). The implementation of *Cross-* and *Poly-Encoder* follows the method proposed in Humeau et al. (2019).

Despite the fact that candidate pools in generation-based dialogue systems are typically small, we are interested in understanding the performance of *Uni-Encoder* with enlarged pools. To this end, we vary the pool size from 10 and 20

Models	Ubuntu Corpus V2					PersonaChat			
	R_{10}@1	R_{10}@2	R_{10}@5			R_{20}@1	MRR		
BERT (Devlin et al., 2019)	0.781	0.890	0.980			0.707	0.808		
Poly-Encoder 360 (Humeau et al., 2019)	0.809	-	0.981			-	-		
SA-BERT (Gu et al., 2020)	0.830	0.919	0.985			-	-		
BERT-CRA (Gu et al., 2021)	-	-	-			0.843	0.903		
Uni-Encoder (Ours)	**0.859***	**0.938***	**0.990***			**0.869***	**0.922***		
	Ubuntu Corpus V1			Douban Conversation Corpus					
	R_{10}@1	R_{10}@2	R_{10}@5	MAP	MRR	P@1	R_{10}@1	R_{10}@2	R_{10}@5
BERT (Devlin et al., 2019)	0.808	0.897	0.975	0.591	0.633	0.454	0.280	0.470	0.828
SA-BERT (Gu et al., 2020)	0.855	0.928	0.983	0.619	0.659	0.496	0.313	0.481	0.847
BERT-SL (Xu et al., 2021)	0.884	0.946	0.990	-	-	-	-	-	-
BERT+FGC (Li et al., 2021)	0.829	0.910	0.980	0.614	0.653	0.495	0.312	0.495	0.850
UMS$_{BERT}$ (Whang et al., 2021)	0.843	0.920	0.982	0.597	0.639	0.466	0.285	0.471	0.829
MDFN (Liu et al., 2021)	0.866	0.932	0.984	0.624	0.663	0.498	0.325	0.511	0.855
SA-BERT+HCL (Su et al., 2021)	0.867	0.940	0.992	0.639	0.681	0.514	**0.330**	0.531	0.858
♣UMS$_{BERT+}$ (Whang et al., 2021)	0.875	0.942	0.988	0.625	0.664	0.499	0.318	0.482	0.858
♣BERT-UMS+FGC (Li et al., 2021)	0.886	0.948	0.990	0.627	0.670	0.500	0.326	0.512	0.869
♣BERT-FP (Han et al., 2021)	0.911	0.962	**0.994**	0.644	0.680	0.512	0.324	0.542	**0.870**
Uni-Encoder (Ours)	0.886	0.946	0.989	0.622	0.662	0.481	0.303	0.514	0.852
♣Uni-Enc+BERT-FP (Ours)	**0.916***	**0.965***	**0.994**	**0.648***	**0.688***	**0.518**	0.327	**0.557**	0.865

Table 4: Evaluations on four benchmark datasets. The models marked with ♣ have been post-trained, and the others are fine-tuned based on the naive BERT (Devlin et al., 2019). $\star$ denotes statistical significance with p-value < 0.05.

to 50 and 100 for each context by randomly selecting additional candidates from the corpus. We then conducted all speed tests on a single NVIDIA A100-SXM4-40GB with CUDA 11.1. The batch size for each paradigm was maximized as much as possible. The results are presented in Figure 2. *Uni-Encoder* demonstrates 4× faster inference speed compared to *Cross-Encoder* when the pool size is appropriate. As the pool size increases, the advantages of *Uni-Encoder* become more pronounced. Compared with *Poly-Encoder*, *Uni-Encoder* exhibits a similar trend, with slightly better overall efficiency. Furthermore, we have also deployed *Uni-Encoder* in a commercial psychotherapy chatbot to rank the responses generated by large language models (LLMs). It has shown to be even more advantageous in this real-world dialogue application, as it returns results with only one forward pass, thus reducing the latency caused by other factors such as data transfer.

4.6 Qualitative Analysis

To further understand the performance gap between different paradigms, we take the model checkpoints from Section 4.3 to go through examples that these methods predict differently. Some of the studied cases are shown in Table 5 in Appendix. *Uni-Encoder* is found to have the most specific and diverse selections. In contrast, even though some results of the other paradigms are not logically problematic, they sometimes prefer more generic responses. We conjecture this difference results from the fact that *Uni-Encoder* compares and scores all the responses simultaneously. Candidates can still interact adequately with each other through their common attention to the context. With such an advantage, it would be easier to distinguish hard negatives from true positives.

5 Discussion

This paper presents a new paradigm for the generation-based dialogue response selection task. Our proposed *Uni-Encoder* avoids re-computing the lengthy context in the current state-of-the-art *Cross-Encoder* method while maintaining the full context to candidate attention. Experimental results on four benchmark datasets show that our approach is both fast and accurate. As *Uni-Encoder* holds the potential to build a more effective and efficient ranking paradigm, our future research will explore its usage in broader applications, such as improving the reward model in the reinforcement learning from human feedback (RLHF) framework (Stiennon et al., 2020; Nakano et al., 2021; Ouyang et al., 2022).

6 Limitations

One major limitation of *Uni-Encoder* is its suitability only for generation-based dialogue systems in which the number of responses is small. A two-stage approach is necessary for retrieval-based systems: Context-independent encoding methods like *Poly-Encoder* first filter out a small set of candidates from the large pool, then *Uni-Encoder* can pick out the best response from the pre-filtered collection. Moreover, as discussed in Section 5, *Uni-Encoder* could be a good component of the RLHF approach. However, the increasing research of pure generation methods with alignments baked-in (Arora et al., 2022; Liu et al., 2023) may gradually replace the SFT+RL method. Consequently, *Uni-Encoder* will have a smaller and smaller impact in terms of application. Nevertheless, because *Uni-Encoder* unified all other ranking paradigms, we believe it remains helpful even as a theoretical framework.

References

Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. 2020. Towards a human-like open-domain chatbot. *ArXiv preprint*, abs/2001.09977.

Kushal Arora, Kurt Shuster, Sainbayar Sukhbaatar, and Jason Weston. 2022. Director: Generator-classifiers for supervised language modeling. *ArXiv preprint*, abs/2206.07694.

Siqi Bao, Huang He, Fan Wang, Hua Wu, Haifeng Wang, Wenquan Wu, Zhihua Wu, Zhen Guo, Hua Lu, Xinxian Huang, et al. 2021. Plato-xl: Exploring the large-scale pre-training of dialogue generation. *ArXiv preprint*, abs/2109.09519.

Leyang Cui, Fandong Meng, Yijin Liu, Jie Zhou, and Yue Zhang. 2022. Towards robust online dialogue response generation.

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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Jia-Chen Gu, Tianda Li, Quan Liu, Zhen-Hua Ling, Zhiming Su, Si Wei, and Xiaodan Zhu. 2020. Speaker-aware BERT for multi-turn response selection in retrieval-based chatbots. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 2041–2044. ACM.

Jia-Chen Gu, Hui Liu, Zhen-Hua Ling, Quan Liu, Zhigang Chen, and Xiaodan Zhu. 2021. Partner matters! an empirical study on fusing personas for personalized response selection in retrieval-based chatbots. *ArXiv preprint*, abs/2105.09050.

Janghoon Han, Taesuk Hong, Byoungjae Kim, Youngjoong Ko, and Jungyun Seo. 2021. Fine-grained post-training for improving retrieval-based dialogue systems. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1549–1558, Online. Association for Computational Linguistics.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2019. Poly-encoders: Transformer architectures and pre-training strategies for fast and accurate multi-sentence scoring. *ArXiv preprint*, abs/1905.01969.

Yuntao Li, Can Xu, Huang Hu, Lei Sha, Yan Zhang, and Daxin Jiang. 2021. Small changes make big differences: Improving multi-turn response selection\\in dialogue systems via fine-grained contrastive learning. *ArXiv preprint*, abs/2111.10154.

H Liu, C Sferrazza, and P Abbeel. 2023. Chain of hindsight aligns language models with feedback. *ArXiv preprint*, abs/2302.02676.

Longxiang Liu, Zhuosheng Zhang, Hai Zhao, Xi Zhou, and Xiang Zhou. 2021. Filling the gap of utterance-aware and speaker-aware representation for multi-turn dialogue. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 13406–13414. AAAI Press.

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*, pages 285–294, Prague, Czech Republic. Association for Computational Linguistics.

Ryan Lowe, Nissan Pow, Iulian Vlad Serban, Laurent Charlin, Chia-Wei Liu, and Joelle Pineau. 2017. Training end-to-end dialogue systems with the ubuntu dialogue corpus. *Dialogue & Discourse*, 8(1):31–65.

Junyu Lu, Xiancong Ren, Yazhou Ren, Ao Liu, and Zenglin Xu. 2020. Improving contextual language models for response retrieval in multi-turn conversation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 1805–1808. ACM.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *ArXiv preprint*, abs/2112.09332.

Yixin Nie, Mary Williamson, Mohit Bansal, Douwe Kiela, and Jason Weston. 2021. I like fish, especially dolphins: Addressing contradictions in dialogue modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Online. Association for Computational Linguistics.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Eric Michael Smith, Y-Lan Boureau, and Jason Weston. 2021. Recipes for building an open-domain chatbot. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 300–325, Online. Association for Computational Linguistics.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 3776–3784. AAAI Press.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. 2020. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Yixuan Su, Deng Cai, Qingyu Zhou, Zibo Lin, Simon Baker, Yunbo Cao, Shuming Shi, Nigel Collier, and Yan Wang. 2021. Dialogue response selection with hierarchical curriculum learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1740–1751, Online. Association for Computational Linguistics.

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, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, pages 267–275. ACM.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. Lamda: Language models for dialog applications. *ArXiv preprint*, abs/2201.08239.

Jack Urbanek, Angela Fan, Siddharth Karamcheti, Saachi Jain, Samuel Humeau, Emily Dinan, Tim Rocktäschel, Douwe Kiela, Arthur Szlam, and Jason Weston. 2019. Learning to speak and act in a fantasy text adventure game. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 673–683, Hong Kong, China. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *ArXiv preprint*, abs/1506.05869.

Taesun Whang, Dongyub Lee, Chanhee Lee, Kisu Yang, Dongsuk Oh, and Heuiseok Lim. 2020. An effective domain adaptive post-training method for BERT in response selection. In *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020*, pages 1585–1589. ISCA.

Taesun Whang, Dongyub Lee, Dongsuk Oh, Chanhee Lee, Kijong Han, Dong-hun Lee, and Saebyeok Lee. 2021. Do response selection models really know what's next? utterance manipulation strategies for multi-turn response selection. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 14041–14049. AAAI Press.

Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2017. Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 496–505, Vancouver, Canada. Association for Computational Linguistics.

Ruijian Xu, Chongyang Tao, Daxin Jiang, Xueliang Zhao, Dongyan Zhao, and Rui Yan. 2021. Learning an effective context-response matching model with self-supervised tasks for retrieval-based dialogues. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 14158–14166. AAAI Press.

Zhen Xu, Bingquan Liu, Baoxun Wang, Chengjie Sun, and Xiaolong Wang. 2017. Incorporating loose-structured knowledge into conversation modeling via recall-gate lstm. In *2017 international joint conference on neural networks (IJCNN)*, pages 3506–3513. IEEE.

Chunyuan Yuan, Wei Zhou, Mingming Li, Shangwen Lv, Fuqing Zhu, Jizhong Han, and Songlin Hu. 2019. Multi-hop selector network for multi-turn response selection in retrieval-based chatbots. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 111–120, Hong Kong, China. Association for Computational Linguistics.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

A Qualitative Analysis

#	Examples	
1	A: have you looked in system settings >brightness and lock ? not power options B: yes, of course. I'm here because the standard ways are failing on two my precise installations	
	⋆Uni: care to post a screenshot?	**Cross: I was just wondering**
	Bi: sry	**Poly: Ah, ok.**
2	A: Is there a way to force apt-get to install a package even if apt is locked by another running apt? B: you don't want to do that wait till the updates are done then A: It will take to long. Its a do-release-upgrade	
	⋆Uni/Cross: that will break things if you interupt it	
	Bi: Yes. I've done it several times	**Poly: ok**
3	A: Does anyone know if there is a crossfeed plugin for Rhythmbox in the repositories? B: why do want to feed rhythmbox? A: crossfeed is a type of signal processing that removes the separation inherent in stereo recordings it's for headphone listening	
	⋆Uni/Cross/Poly: it's called crossfade ;)	
	Bi: could you explain more about what you want?	

Table 5: Cases studied from Ubuntu V2 for comparing selections of different paradigms where ⋆ denotes the correct choice.