# Comparing Neighbors Together Makes it Easy: Jointly Comparing Multiple Candidates for Efficient and Effective Retrieval

**Jonghyun Song**[†], **Cheyon Jin**[†], **Wenlong Zhao**[◇], **Andrew McCallum**[◇], **Jay-Yoon Lee**[*†]
[†] Seoul National University [◇]University of Massachusetts Amherst
{hyeongoon11, cheyonjin, lee.jayyoon}@snu.ac.kr
{wenlongzhao, mccallum}@umass.edu

## Abstract

A common retrieve-and-rerank paradigm involves retrieving relevant candidates from a broad set using a fast bi-encoder (BE), followed by applying expensive but accurate cross-encoders (CE) to a limited candidate set. However, relying on this small subset is often susceptible to error propagation from the bi-encoders, which limits the overall performance. To address these issues, we propose the Comparing Multiple Candidates (CMC) framework. CMC compares a query and multiple embeddings of similar candidates (i.e., neighbors) through shallow self-attention layers, delivering rich representations contextualized to each other. Furthermore, CMC is scalable enough to handle multiple comparisons simultaneously. For example, comparing 10K candidates with CMC takes a similar amount of time as comparing 16 candidates with CE. Experimental results on the ZeSHEL dataset demonstrate that CMC, when plugged in between bi-encoders and cross-encoders as a seamless intermediate reranker (BE-CMC-CE), can effectively improve recall@k (+6.7%-p, +3.5%-p for R@16, R@64) compared to using only bi-encoders (BE-CE), with negligible slowdown (<7%). Additionally, to verify CMC's effectiveness as the final-stage reranker in improving top-1 accuracy, we conduct experiments on downstream tasks such as entity, passage, and dialogue ranking. The results indicate that CMC is not only faster (11x) but also often more effective than cross-encoders with improved prediction accuracy in Wikipedia entity linking (+0.7%-p) and DSTC7 dialogue ranking (+3.3%-p).

## 1 Introduction

The two-stage approach of retrieval and reranking has become a predominant method in tasks such as entity linking (EL) (Wu et al., 2020; Zhang and Stratos, 2021; Xu et al., 2023), open-domain question answering (ODQA) (Nogueira and Cho, 2019; Agarwal et al., 2022b; Shen et al., 2022; Qu et al., 2020), and dialogue systems (Mele et al., 2020). Typically, bi-encoders (BE) are used to efficiently retrieve relevant candidates from a large set of documents (e.g., knowledge base), and then cross-encoders (CE) effectively rerank only a limited subset of candidates already retrieved by BE (Nogueira and Cho (2019); Figure 1.a-b).

The current BE-CE approach, although widely adopted, has an efficiency-effectiveness trade-off and is susceptible to error propagation. When less accurate BE retrieves candidates, the whole framework risks the error propagation of missing the gold candidates due to inaccuracies from the retriever. Simply increasing the number of candidates is not a viable solution considering the slow serving time of CE[1][2]. Consequently, users are faced with the dilemma of deciding which is worse: error propagation from BE versus the slow runtime of CE.

To resolve this issue, various strategies have been proposed to find an optimal balance in the efficiency-effectiveness trade-off. Prior works (Khattab and Zaharia (2020); Zhang and Stratos (2021); Cao et al. (2020); Humeau et al. (2019)) have enhanced bi-encoder architectures with a late interaction component. However, these models only focus on single query-candidate pair interaction. Also, they sometimes require storing entire token embeddings per candidate sentence which results in tremendous memory use (Figure 1.c).

Our proposed Comparing Multiple Candidates (CMC) makes reranking easy by comparing similar candidates (i.e., neighbors) together. By jointly contextualizing the single vector embeddings from each candidate through shallow bi-directional self-attention layers, CMC achieves high prediction accuracy and runtime efficiency that are comparable

[*] Corresponding author

[1] For the serving time of cross-encoders, see §D.1.

[2] Furthermore, increasing the number of candidates for CE does not necessarily improve end-to-end accuracy (Wu et al., 2020). We confirm this in the experiments. See appendix D.6.

*Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22255–22269
November 12-16, 2024 ©2024 Association for Computational Linguistics

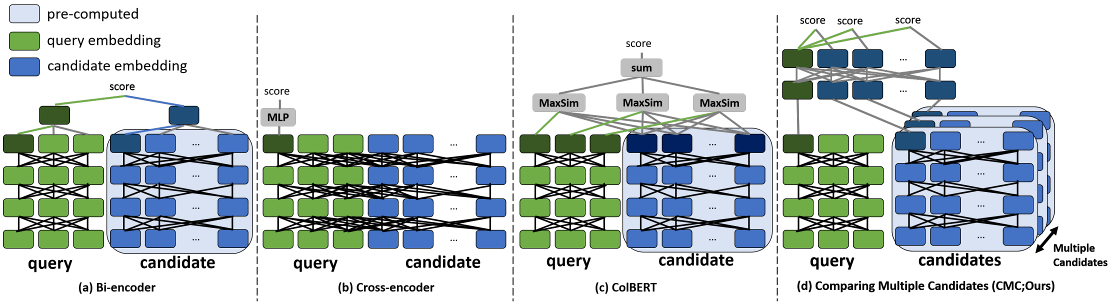

Figure 1: Model architectures for retrieval tasks. (a), (b), and (c) are existing architectures. (d) is our proposed 'Comparing Multiple Candidates (CMC)' architecture, which computes compatibility score by comparing the embeddings of a query and K multiple candidates via self-attention layers. Contrary to (a)-(c), CMC can process multiple candidates at once rather than conducting several forward passes for each (query, candidate) pair.

to, or better, than existing methods which require single or multiple vector embeddings.

In other words, CMC only takes a single forward pass for input $(\text{query}, \text{candidate}_1, ..., \text{candidate}_k)$ with a pre-computed single vector embedding. In contrast, models such as CE and other late interaction models take $k$ separate forward passes for input pairs $(\text{query}, \text{candidate}_1), ..., (\text{query}, \text{candidate}_k)$, sometimes requiring multiple vector embeddings per each candidate. CMC maintains both the *efficiency* of BE with pre-computed single-vector candidate embeddings, and the *effectiveness* of CE with interactions between query and multiple candidates (Figure 1.d).

Practitioners can plug in CMC as the *seamless intermediate reranker* (BE-CMC-CE) which can enhance retrieval performance with negligible extra latency. This improvement is crucial for preventing error propagation from the retrieval process, resulting in more reliable candidates for the final stage (Figure 2-3). On the other hand, CMC also can serve as a fast and effective *final-stage reranker* improving top-1 accuracy (BE-CMC). If there's a time constraint, using CMC as the final reranker can be a good option, as running a cross-encoder requires significantly more time (Table 3; Figure 4).

In experiments, we evaluate CMC on Zero-SHot Entity-Linking dataset (ZeSHEL; Logeswaran et al. (2019)) to investigate how much CMC seamlessly enhances a retriever's performance when plugged in to BE (BE-CMC). The results show CMC provides higher recall than baseline retrievers at a marginal increase in latency (+0.07x; Table 1). Compared to standard BE-CE, plugging in CMC as the seamless intermediate reranker (BE-CMC-CE) can provide fewer, higher-quality candidates to CE, ultimately improving the accuracy of CE reranking. (Table 2). To examine the effectiveness of CMC which acts as the final stage reranker, we evaluate CMC on entity, passage, and dialogue ranking tasks. We observe that CMC outperforms CE on Wikipedia entity linking datasets (+0.7p accuracy) and DSTC7 dialogue ranking datasets (+3.3p MRR), requiring only a small amount (0.09x) of CE's latency (Table 3).

The main contributions of the paper are as follows:

- We present a novel reranker, CMC, which improves both accuracy and scalability. CMC contextualizes candidate representations with similar candidates (i.e., neighbors), instead of solely focusing on a single query-candidate pair (§3).
- CMC can serve as the seamless intermediate reranker which can significantly improve retrieval performance with only a negligible increase in latency. This results in a more confident set of candidates for the final-stage reranker that improves end-to-end accuracy compared to conventional bi-encoders (§4.3)
- Experimental results show that the final stage reranking of CMC is highly effective on passage, entity, and dialogue ranking tasks compared to various baselines among the low-latency models (§4.4).
- Additionally, we show that CMC can benefit from domain transfer from sentence encoders while BE and many others cannot (§4.5).

## 2 Background and Related Works

### 2.1 Retrieve and Rerank

Two-stage retrieval systems commonly consist of a fast retriever and a slow but accurate reranker. Although the retriever is fast, its top-1 accuracy

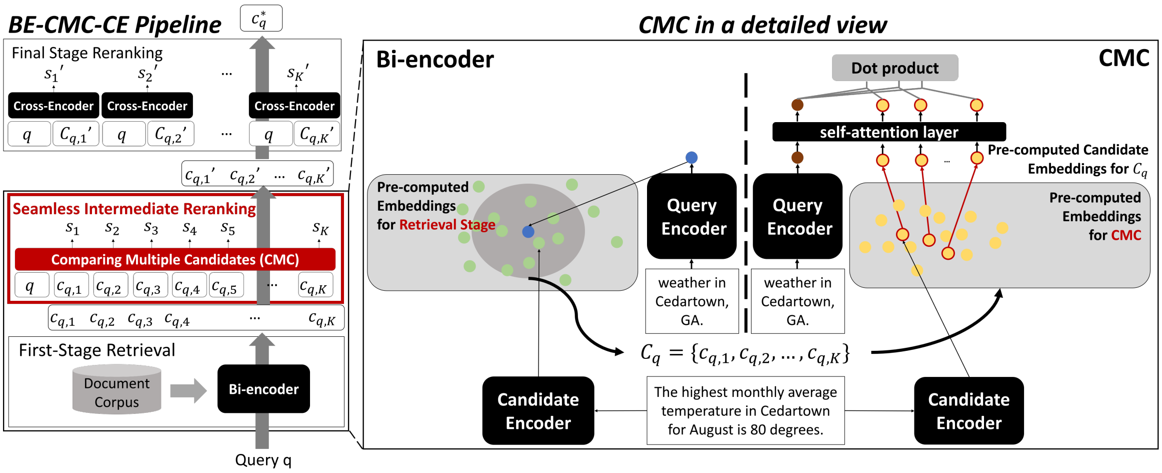
BE-CMC-CE Pipeline
$c_q^*$
Final Stage Reranking
$s_1'$
$s_2'$
…
$s_K'$
Cross-Encoder
Cross-Encoder
Cross-Encoder
$q$
$C_{q,1}'$
$q$
$C_{q,2}'$
…
$q$
$C_{q,K}'$
$c_{q,1}'$ $c_{q,2}'$ … $c_{q,K}'$
Seamless Intermediate Reranking
$s_1$ $s_2$ $s_3$ $s_4$ $s_5$ $s_K$
Comparing Multiple Candidates (CMC)
$q$ $c_{q,1}$ $c_{q,2}$ $c_{q,3}$ $c_{q,4}$ $c_{q,5}$ … $c_{q,K}$
$c_{q,1}$ $c_{q,2}$ $c_{q,3}$ $c_{q,4}$ … $c_{q,K}$
First-Stage Retrieval
Document Corpus
Bi-encoder
Query q
CMC in a detailed view
Bi-encoder
CMC
Dot product
self-attention layer
Pre-computed Candidate Embeddings for $C_q$
Pre-computed Embeddings for Retrieval Stage
Query Encoder
Query Encoder
weather in Cedartown, GA.
weather in Cedartown, GA.
Pre-computed Embeddings for CMC
$C_q = \{c_{q,1}, c_{q,2}, \dots, c_{q,K}\}$
Candidate Encoder
The highest monthly average temperature in Cedartown for August is 80 degrees.
Candidate Encoder

Figure 2: Overview of the proposed CMC framework that compares multiple candidates at once. CMC can *seamlessly enhance retriever*, finding top-K' candidates, or function as a direct reranker which outputs top-1 candidate. Candidate embeddings for bi-encoders and CMC are both precomputed while query embeddings for bi-encoders and CMC are computed in parallel on the fly. After bi-encoders retrieve top-$K$ candidates, CMC indexes the corresponding candidate embeddings and passes through a two-layer transformer encoder. Here, the additional latency is limited to the execution of self-attention layers.

tends to be suboptimal. Therefore, a candidate set $C_q = \{c_{q,1}, c_{q,2}, \dots, c_{q,K}\} \subseteq \mathcal{C}$ whose elements are $K$ most relevant candidates in the corpus $\mathcal{C}$ is retrieved for further reranking.

A reranker $s_\theta(q, c_{q,j})(1 \leq j \leq K)$ is a model trained to assign a fine-grained score between the query $q$ and each candidate $c_{q,j}$ from the relatively small set of candidates $C_q$. It is an expressive model that is slower but more accurate than the retriever. The candidate with the highest score $c_q^* = \arg\max_{c_{q,j} \in C_q} s_\theta(q, c_{q,j})$ is the final output of the retrieve-and-rerank pipeline where query $q$ should be linked.

## 2.2 Related Work

**Bi-encoders and Cross-encoders** In two-stage retrieval, the compatibility score between the query and candidate can be computed by diverse functions. Nogueira et al. (2019a) retrieve candidates using the bag-of-words BM25 retriever and then apply a cross-encoder reranker, transformer encoders that take the concatenated query and candidate tokens as input (Logeswaran et al., 2019; Wu et al., 2020). Instead of BM25 retriever, other works (Lee et al., 2019; Gillick et al., 2019; Karpukhin et al., 2020) employ a pre-trained language model for a bi-encoders retriever to encode a query and a candidate separately and get the compatibility score. The scalability of bi-encoders as a retriever arises from the indexing of candidates and maximum inner-product search (MIPS); however, they tend to be less effective than cross-encoders as candidate representations do not reflect the query's information (Figure 1.a-b). To enhance the performance of bi-encoders, follow-up works propose a task-specific fine-tuned model (Gao and Callan, 2022), injecting graph information (Wu et al., 2023; Agarwal et al., 2022a), and multi-view text representations (Ma et al., 2021; Liu et al., 2023).

**Late Interaction** Late interaction models, which typically function as either a retriever or a reranker, enhance bi-encoder architectures with a late interaction component between the query and the candidate.

Poly-encoder (Humeau et al., 2019) and Mix-Encoder (Yang et al., 2023) represent query information through cross-attention with a candidate to compute the matching score. However, these models have overlooked the opportunity to explore the interaction among candidates.

Sum-of-Max (Khattab and Zaharia, 2020; Zhang and Stratos, 2021) and DeFormer (Cao et al., 2020) rely on maximum similarity operations or extra cross-encoder layers on top of bi-encoders. However, they lack scalability due to the need to precompute and save every token embedding per each candidate.[3] As a collection of documents continuously changes and grows, this storage requirement

[3] For example, 3.2TB is required for storing ~5M entity descriptions from Wikipedia, each with 128 tokens. In contrast, storing a single vector embedding per entity description for bi-encoders only requires 23GB.

poses practical limitations on managing and updating the document indices.

CMC differs from these models in its enhanced scalability by comparing a single embedding for each candidate. This approach provides a deeper exploration of relational dynamics from interactions across multiple candidates while improving time and memory efficiency.

**Listwise Ranking** CMC is not the first approach to compare a list of documents to enhance ranking performance (Han et al., 2020; Zhang et al., 2022; Xu et al., 2023). These listwise ranking methods process cross-encoder logits for the list $(\text{query}, \text{candidate}_1, \ldots, \text{candidate}_\text{K})$ to rerank $K$ candidates from cross-encoders. However, these approaches lack scalability and efficiency due to reliance on cross-encoder representations.

Unlike previous listwise ranking models, we propose a method that employs representations from independent sentence encoders rather than cross-encoders. Boosting scalability with independent representations, CMC can seamlessly enhance retrievers by maintaining prediction accuracy.

## 3 Proposed Method

### 3.1 Model Architecture

Comparing Multiple Candidates, CMC, employs shallow self-attention layers to capture both query-candidate and candidate-candidate interactions. Unlike other late interaction models which compute the compatibility scores by only considering a single query-candidate pair (Khattab and Zaharia, 2020; Humeau et al., 2019; Yang et al., 2023), CMC compares each candidate to the query and other candidates at the same time (Figure 1.(d)). The self-attention layer in CMC processes the concatenated representations of the query and multiple candidates, derived from the independent query and candidate encoders. In this way, CMC obtains enhanced representations of the query and every candidate by contextualizing them with each other. Also, this architecture is scalable to a large set of corpus by pre-computing and indexing candidate embeddings. For example, processing 2K candidates only takes twice as long as processing 100 (Figure 4).

**Query and Candidate Encoders** Prior to CMC, the first-stage retriever (e.g., bi-encoders) retrieves the candidate set with K elements $C_q = \{c_{q,1}, \ldots c_{q,K}\}$ for query $q$. CMC then obtains the aggregated encoder output (e.g., [CLS] token embedding) of query sentence tokens $\mathbf{h}_q^{sent}$ and candidate sentence tokens $\mathbf{h}_{c_{q,j}}^{sent}$ from the query encoder $\texttt{Enc}_{qry}$ and the candidate encoder $\texttt{Enc}_{can}$. These encoders play the same role as conventional bi-encoders by condensing each query and candidate information into a single vector embedding but are trained separately from the first-stage stage retriever.

$$\mathbf{h}_q^{sent} = \text{agg}(\texttt{Enc}_{qry}(\texttt{[CLS]}\mathrm{x}_q^0 \ldots \mathrm{x}_q^k)) \tag{1}$$

$$\mathbf{h}_{c_{q,j}}^{sent} = \text{agg}(\texttt{Enc}_{can}(\texttt{[CLS]}\mathrm{x}_{c_{q,j}}^0 \ldots \mathrm{x}_{c_{q,j}}^k)) \tag{2}$$

$\mathrm{x}_q$ and $\mathrm{x}_{c_{q,j}}$ are tokens of each query and candidate. The aggregator function agg extracts [CLS] embedding from the last layer of encoder[4].

**Self-attention Layer** The shallow self-attention layers process concatenated embeddings of a query and all candidates. This lightweight module enables parallel computation (*efficient*) and outputs contextualized embeddings via interactions between query and candidates (*effective*). In the reranking perspective, Representing candidates together with self-attention layers (Attn) enables fine-grained comparison among candidates. The self-attention layers consist of two layers of vanilla transformer encoder (Vaswani et al., 2017) in Pytorch without positional encoding.

$$\left[\mathbf{h}_q^{\texttt{CMC}}; \mathbf{h}_{c_{q,1}}^{\texttt{CMC}}; \ldots; \mathbf{h}_{c_{q,K}}^{\texttt{CMC}}\right] = \texttt{Attn}\left(\left[\mathbf{h}_q^{sent}; \mathbf{h}_{c_{q,1}}^{sent}; \ldots; \mathbf{h}_{c_{q,K}}^{sent}\right]\right) \tag{3}$$

Subsequently, the reranker computes the final prediction $c_q^*$ via dot products of query and candidate embeddings from the self-attention layer:

$$c_q^* = \underset{c_{q,j} \in C_q}{\arg\max}\, \mathbf{h}_q^{\texttt{CMC}} \cdot \left(\mathbf{h}_{c_{q,j}}^{\texttt{CMC}}\right)^\top \tag{4}$$

### 3.2 Training

**Optimization** The training objective is minimizing the cross-entropy loss regularized by the Kullback-Leibler (KL) divergence between the score distribution of the trained model and the bi-encoder. The loss function is formulated as:

$$\mathcal{L}(q, \tilde{C}_q) = \textstyle\sum_{i=1}^{K}(-\lambda_1 y_i \log(p_i) + \lambda_2 p_i \log\left(\frac{p_i}{r_i}\right)) \tag{5}$$

[4] For entity linking tasks, both the query (mention) and candidate (entity) sentences include custom special tokens that denote the locations of mention and entity words. These include [SEP], [query_start], [query_end], and [DOC] tokens following Wu et al. (2020).

$y_i$ and $p_i$ are the ground truth and predicted probability for i-th candidate. The retriever's probability for the candidate is represented as $r_i$. $\lambda_1$ and $\lambda_2$ are weights combining the two losses.

**Negative Sampling** We sample hard negatives based on the first-stage retriever's score for each query-candidate pair $(q, c_{q,j})$: $\forall j \in \{1, \ldots, K\} \setminus \{\text{gold index}\}$,

$$c_{q,j} \sim \frac{\exp(s_{\text{retriever}}(q, c_{q,j}))}{\sum_{\substack{k=1\& \\ k\neq\text{gold index}}}^{K} \exp(s_{\text{retriever}}(q, c_{q,k}))} \quad (6)$$

In experiments, CMC and other baselines follow the same optimization and negative sampling strategy.[5]

### 3.3 Inference

**Offline Indexing** CMC can pre-compute and index the embeddings of candidates in the collection (e.g., knowledge base), unlike cross-encoders (Figure 1). This offline indexing scheme significantly reduces inference time compared to cross-encoders, making the runtime of CMC comparable to that of bi-encoders (§4.4). While reducing time complexity, CMC is highly memory-efficient requiring less than $1\%$ of index size needed by Sum-of-Max and Deformer, which store every token embedding per candidate. This is because CMC only stores a single vector embedding per candidate.

**Parallel Computation of Query Representations** The end-to-end runtime for retrieving and reranking with CMC can be comparable to that of bi-encoder retrieval. The runtime can be further improved by parallelizing query encoders in both bi-encoder and CMC (Figure 2). Ideally, the additional latency for running CMC is limited to the execution of a few self-attention layers.

**CMC as the Seamless Intermediate Reranker** CMC can serve as a seamless intermediate reranker that maintains the latency-wise user experience while providing improved retrieval performance when combined with a bi-encoder. Thanks to the parallel computation discussed above, plugging in CMC after bi-encoders should minimally impact retrieval latency compared to just using the bi-encoder. The process starts with the first-stage retrievers, such as bi-encoders, retrieving a broad set of candidates. CMC then narrows this set down to fewer, higher-quality candidates with a more manageable number (e.g., 64 or fewer) for the reranker. Since CMC, the seamless intermediate reranker, filters candidates from the first-stage retriever with negligible additional latency, its runtime is comparable to that of bi-encoders. As a result, the improved candidate quality boosts the prediction accuracy of the final-stage reranker (e.g., cross-encoders) with only a marginal increase in computational cost (Figure 3; §4.3).

**CMC as the Final Stage Reranker** CMC can obviously serve as the final-stage reranker to increase top-1 accuracy. Enriching contextualized representations of the query and candidates helps improve top-1 accuracy in reranking while maintaining efficiency with a single vector embedding. Notably, CMC remains effective even when the number of candidates varies during inference, despite being trained with a fixed number of candidates. For example, when trained with 64 candidates on the MS MARCO passage ranking dataset, CMC still performs effectively with up to 1K candidates. This demonstrates not only the scalability of CMC but also its robustness in processing a diverse range of candidate sets (§4.4).

## 4 Experiments

### 4.1 Dataset

To evaluate the robustness of CMC, we conduct experiments on various ranking tasks where the retrieve-and-rerank approach is commonly used. For entity linking, we utilize datasets linked to the Wikipedia knowledge base (AIDA-CoNLL (Hoffart et al., 2011), WNED-CWEB (Guo and Barbosa, 2018), and MSNBC (Cucerzan, 2007)), as well as a ZEro-SHot Entity Linking dataset (ZeSHEL; Logeswaran et al. (2019)) based on the Wikia[7] knowledge base. The candidates are retrieved from bi-encoders fine-tuned for each knowledge base (Wu et al., 2020; Yadav et al., 2022). For passage ranking, we conduct an experiment on MS MARCO with 1K candidates from BM25 as the first-stage retriever following Bajaj et al. (2016). For dialogue ranking tasks, we test our model on DSTC7 challenge (Track 1) (Yoshino et al., 2019), where candidates are officially provided. The primary metric used is recall@k, as datasets typically have only

[5] The code and link to datasets are available at https://github.com/yc-song/cmc

[6] recall@64 of Poly-encoder and Sum-of-max from Zhang and Stratos (2021) is reported as 84.34 and 89.62, respectively.

[7] now Fandom: https://www.fandom.com

|  | Method | Test |  |  |  |  |  | Speed | Index Size |
|---|---|---|---|---|---|---|---|---|---|
|  |  | R@1 | R@4 | R@8 | R@16 | R@32 | R@64 | (ms) | (GB) |
| Single-View | BM25 | 25.9 | 44.9 | 52.1 | 58.2 | 63.8 | 69.1 |  |  |
|  | Bi-encoder (BE$^\spadesuit$) | 52.9 | 64.5 | 71.9 | 81.5 | 85.0 | 88.0 | 568.9 | 0.2 |
|  | Arbo-EL | 50.3 | 68.3 | 74.3 | 78.4 | 82.0 | 85.1 | - | - |
|  | GER | 42.9 | 66.5 | 73.0 | 78.1 | 81.1 | 85.7 | - | - |
|  | Poly-encoder (Poly) $^\heartsuit$ | 40.0±0.7 | 60.2±0.9 | 67.2±0.7 | 72.2±0.8 | 76.5±0.8 | 80.2±0.8 | 581.0 | 0.2 |
|  | BE + Poly$^\heartsuit$ | 56.9±0.8 | 74.8±0.6 | 80.1±0.7 | 84.2±0.5 | 87.5±0.4 | 90.2±0.3 | 574.6 | 0.4 |
|  | Sum-of-max (SOM)$^\heartsuit$ | 27.1±1.8 | 64.1±1.4 | 73.2±0.9 | 79.6±0.7 | 84.1±0.4 | 88.0±0.4 | 6393.0 | 25.7 |
|  | BE + SOM$^\heartsuit$ | 58.5±1.0 | 76.2±1.1 | 81.6±1.0 | 85.8±0.9 | 88.9±0.7 | 91.4±0.6 | 2958.3 | 0.2 |
|  | - w/ offline indexing |  |  |  |  |  |  | 597.3 | 25.9 |
|  | BE$^\spadesuit$ + CMC(Ours) | **59.1**±0.3 | **77.6**±0.3 | **82.9**±0.1 | **86.3**±0.2 | **89.3**±0.2 | **91.5**±0.1 | 607.2 | 0.4 |
| Multi-View | MuVER | 43.5 | 68.8 | 75.9 | 77.7 | 85.9 | 89.5 | - | - |
|  | MVD | 52.5 | 73.4 | 79.7 | 84.4 | 88.2 | 91.6 | - | - |
|  | MVD + CMC(Ours) | **59.0** | **77.8** | **83.1** | **86.7** | **89.9** | **92.4** | - | - |

Table 1: Retrieval performance over ZeSHEL dataset. The best and second-best results are denoted in bold and underlined. BE$^\spadesuit$ is bi-encoder from Yadav et al. (2022) which is used for CMC. $^\heartsuit$ indicates our implementation as recall@k for all k are not provided in previous work[6]. results on BE + Reranker (e.g., BE+CMC) are conducted over the top 512 candidates from the first-stage retriever and averaged over experiments with 5 random seeds.

one answer or rarely a few answers per query. Further details are presented in §B.

## 4.2 Training Details

CMC and other baselines are trained under the same training strategies. All models use the same loss function and negative sampling (§3.2) with the AdamW optimizer and a 10% linear warmup scheduler. Also, we examine diverse sentence encoder initialization for CMC and late interaction models, including vanilla BERT and BERT-based models fine-tuned on in- and out-of-domain datasets. After training, we select the best results for each model.[8] For ZeSHEL, training CMC and other low-latency baselines for one epoch on an NVIDIA A100 GPU takes about 4 hours. The training details for each dataset are in §C, and the ablation studies for diverse training strategies are presented in §4.5 and §D.5.

## 4.3 CMC as the Seamless Intermediate Reranker

We conduct two experiments on the ZeSHEL dataset to verify the impact of CMC as the seamless intermediate retriever (BE+CMC+CE). We examine whether the introduction of CMC can improve retrieval performance with negligible overhead as promised. In the first experiment, we compare the performance and speed of CMC plugged in with bi-encoders (BE+CMC) with other retrieval pipelines. Remarkably, even when other rerankers are plugged in with the same bi-encoder, CMC still achieves the highest Recall@k (Table 1) at a marginal latency increase. In the second experiment, we assess how a more confident set of candidates retrieved by BE+CMC contributes to improving end-to-end (BE+CMC+CE) accuracy compared to solely using bi-encoders (Figure 3).

**Baselines** To assess CMC's effectiveness in enhancing retrieval, we evaluate BE+CMC on 512 bi-encoder retrieved candidates and compare it to baselines categorized into two types: single- and multi-view retrievers.[9] We use bi-encoders (Yadav et al., 2022) and MVD (Liu et al., 2023) as the first-stage retrievers for the single-view and multi-view settings, respectively. For the baselines, we select the state-of-the-art retrievers for the ZeSHEL dataset. For single-view retrievers, we select the poly-encoder (Humeau et al., 2019), Sum-of-max (Zhang and Stratos, 2021), Arbo-EL (Agarwal et al., 2022b), and GER (Wu et al., 2023). Among these, Arbo-EL and GER utilize graph information, unlike CMC and other baselines. For multi-view retrievers, we include MuVER (Ma et al., 2021) and MVD (Liu et al., 2023).

**Experimental Results** In Table 1, plugging in CMC with a single-view retriever outperforms baselines across all $k$, demonstrating its effectiveness in the end-to-end retrieval process. With a marginal increase in latency (+0.07x), CMC boosts recall@64 to 91.51% on the candidates from the first-stage retriever, which has a recall@64 of 87.95%. Especially, the recall of Poly-encoder and Sum-of-max lags behind CMC even when they are plugged in

[8] If more favorable results are found in prior works over the same candidates, we use those results.

[9] Single-view retrievers consider only a single global view derived from the entire sentence, whereas multi-view retrievers divide candidate information into multiple local views.

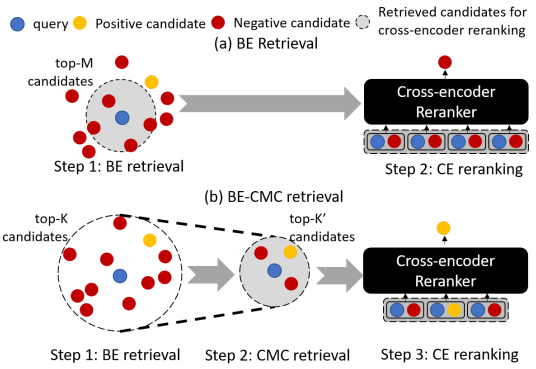

Figure 3: Illustration of candidate retrieval for cross-encoders (CE). Suppose cross-encoders can process up to M candidates due to limited scalability. (a) In bi-encoder (BE) retrieval, the BE-CE framework takes M candidates and risks missing the gold candidates due to inaccurate bi-encoders, causing the entire system to suffer from error propagation from the retriever and fail to get the correct candidate. (b) When CMC is introduced as the seamless intermediate reranker (BE-CMC-CE), CMC can consider a significantly larger pool (K) of BE candidates. This allows CMC to provide much fewer K' (K>M>K') and higher-quality candidates to the CE while increasing the chance to include the positive candidate.

with the same bi-encoders (BE+Poly & BE+SOM). Sum-of-max, which closely follows CMC, requires a tremendous index (60x of CMC) to achieve comparable latency to CMC. To show that CMC seamlessly enhances any retriever type, we examine the increase in recall of CMC upon a multi-view retriever (MVD+CMC). The results show that CMC consistently improves recall performance, moving from 91.55% to 92.36% at recall@64. This demonstrates CMC's general capability to enhance recall performance, regardless of the first-stage retriever. For the effect of the number of candidates from the first-stage retriever, see §D.2.

We question whether BE+CMC can reduce the latency of the overall retrieval and reranking process while maintaining the overall accuracy (Figure 3). In essence, if we can have fewer but higher quality candidates, end-to-end accuracy can be improved while CE forward passes are called fewer times with a reduced set of candidates. To examine the quality of candidates from the seamless intermediate reranker CMC, we report the final reranking accuracy of cross-encoders when candidates are retrieved by BE+CMC and compare it to conventional BE retrieval (Table 2).

Table 2 shows that cross-encoders outperform conventional bi-encoders, even with fewer candidates retrieved by CMC. Cross-encoders with 16 candidates from CMC are 1.75x faster and achieve slightly better accuracy compared to using 64 bi-encoder candidates (line 3 vs. 8-9). Furthermore, cross-encoders reach the best accuracy with 64 candidates from CMC, surpassing the accuracy obtained with the same number of bi-encoder candidates, with only a marginal increase in latency (line 3 vs. 11).

| | Retrieved (k) | | Recall@k | Unnormalized Accuracy | | | | | Comparative |
|---|---|---|---|---|---|---|---|---|---|
| | Bi-encoder | CMC | | Forgotten Realms | Lego | Start Trek | Yugioh | Macro Avg. | Latency (%) |
| 1 | 8 | - | 77.72 | 78.92 | 65.14 | 62.76 | 48.64 | 63.87 | 38.90% |
| 2 | 16 | - | 81.52 | 80.17 | 66.14 | 63.69 | 49.64 | 64.91 | 48.85% |
| 3 | 64 | - | 87.95 | 80.83 | 67.81 | 64.23 | 50.62 | 65.87 | 100% |
| 4 | 64 | 8 | 82.45 | 80.67 | 66.56 | 64.54 | 50.71 | 65.62 | 43.04% |
| 5 | 256 | 8 | 82.86 | 80.92 | 66.89 | 64.42 | 50.86 | 65.77 | 43.36% |
| 6 | 512 | 8 | 82.91 | 80.75 | 67.14 | 64.35 | 51.01 | 65.81 | 43.55% |
| 7 | 64 | 16 | 85.46 | 80.5 | 66.97 | 64.47 | 50.68 | 65.66 | 56.76% |
| 8 | 256 | 16 | 86.22 | 80.75 | 67.31 | **64.63** | 51.1 | 65.95 | 57.08% |
| 9 | 512 | 16 | 86.22 | 80.83 | 67.64 | 64.49 | 50.95 | 65.98 | 57.27% |
| 10 | 256 | 64 | 90.91 | **81.17** | 67.64 | 64.37 | 50.92 | 66.03 | 104.46% |
| 11 | 512 | 64 | **91.51** | 81.00 | **67.89** | 64.42 | 50.86 | **66.04** | 104.65% |

Table 2: Unnormalized accuracy[10] of cross-encoders across various candidate configurations on the ZeSHEL dataset. We underlined when the cross-encoders show superior accuracy with candidates filtered by CMC compared to those from bi-encoders. The top-performing scenarios in each category are highlighted in **bold**. We measure the comparative latency required for running cross-encoders over 64 bi-encoder candidates (260.84ms). For your reference, the CMC runtime 2x when increasing the number of candidates by 16x (from 128 to 1048), while able to compare up to 16k candidates at once. (§D.1)

## 4.4 CMC as the Final Stage Reranker

**Baselines** Baselines are categorized into high-, medium-, and low-latency models. We adopt cross-encoders as our primary baseline for the high-latency model. For the medium-latency models, we include Deformer and Sum-of-max, which utilize all token embeddings to represent candidate information. For the low-latency models, we include the Bi-encoder, Poly-encoder, and Mixencoder, all of which require a single vector embedding for representation and have a serving time similar to that of the Bi-encoder. In this context, CMC is classified as a low-latency method because it requires a single embedding for the candidate and takes 1.17x serving time of the Bi-encoder.

[10] The unnormalized accuracy of the reranker in ZeSHEL is defined as the accuracy computed on the entire test set. In contrast, the normalized accuracy is evaluated on the subset of test instances for which the gold entity is among the top-k candidates retrieved by the initial retriever. For example, if the retriever correctly identifies candidates for three out of five instances and the reranker identifies one correct candidate, unnormalized accuracy is 1/5 = 20%, and normalized accuracy is 1/3 = 33%.

|  | Tasks<br>Datasets | Entity Linking<br>Wikipedia<br>Accuracy | <br>ZeSHEL<br>Accuracy | Passage Ranking<br>MS MARCO Dev<br>R@1 | <br><br>MRR@10 | Dialogue Ranking<br>DSTC7 Challenge<br>R@1 | <br><br>MRR@10 | Compuational Efficiency<br>Total Speed | <br>Extra Memory |
|---|---|---|---|---|---|---|---|---|---|
| High-latency | Cross-encoder | 80.2±0.2 | **65.9**$^{\dagger}$ | **25.4** | **36.8** | 64.7 | 73.2 | 12.9x | - |
| Medium-latency | Deformer | 79.6±0.8 | <u>63.6</u>±0.3 | 23.0$^{\dagger}$ | 35.7$^{\dagger}$ | **68.6** | **76.4** | 4.39x | 125x |
|  | Sum-of-max | <u>80.7</u>±0.2 | 58.8±1.0 | 22.8$^{\dagger}$ | 35.4$^{\dagger}$ | 66.9 | 75.5 | 5.20x | - |
|  | - w/ offline indexing |  |  |  |  |  |  | 1.05x | 125x |
| Low-latency | Bi-encoder | 77.1$^{\dagger}$ | 52.9$^{\dagger}$ | 22.9 | 35.3 | 67.8 | 75.1 | 1x | 1x |
|  | Poly-encoder | 80.2±0.1 | 57.6±0.6 | 23.5 | 35.8 | <u>68.6</u> | <u>76.3</u> | 1.01x | 1.0x |
|  | MixEncoder | 75.4±1.4 | 57.9±0.3 | 20.7$^{\dagger}$ | 32.5$^{\dagger}$ | 68.2$^{\dagger}$ | 75.8$^{\dagger}$ | 1.12x | 1.0x |
|  | CMC (Ours) | **80.9**±0.1 | 59.2±0.3 | <u>23.9</u> | <u>35.9</u> | 68.0 | 75.7 | 1.17x | 1.0x |

Table 3: Reranking Performance on four datasets with three downstream tasks: Entity Linking (Wikipedia-KB based datasets (Hoffart et al., 2011; Guo and Barbosa, 2018; Cucerzan, 2007), ZeSHEL (Logeswaran et al., 2019), Passage Ranking (MS MARCO Passage Ranking (Bajaj et al., 2016), and Dialogue Ranking (Gunasekara et al., 2019). The best result is denoted in **bold** and the second-best result is <u>underlined</u>. MRR stands for mean reciprocal rank. In the entity linking datasets, the results are averaged across five random seeds. To show the computing resources required for the reranking process, we define reranking latency in terms of relative latency and additional memory usage compared to bi-encoders. $^{\dagger}$ indicates that more favorable results are sourced from Wu et al. (2020); Yang et al. (2023); Yadav et al. (2022), respectively.

**Comparison with Low-latency Models** CMC is highly effective across diverse datasets, outperforming or being comparable to other low-latency baselines. Notably, CMC surpasses bi-encoders on every dataset with only a marginal increase in latency. This indicates that replacing simple dot products with self-attention layers across multiple candidates can enhance reranking performance, likely by taking advantage of the relational dynamics among the candidates. Evaluated against the Poly-encoder and MixEncoder, CMC demonstrates superior prediction capability in tasks like passage ranking and entity linking, which require advanced reading comprehension capability.

**Comparison with Medium-latency Models** When compared with Medium-latency models such as Deformer and Sum-of-max, CMC demonstrates its capability not only in memory efficiency but also in maintaining strong performance. CMC mostly surpasses these models in entity linking and passage ranking tasks. Also, CMC offers significant improvements in speed over Deformer (1.17x vs. 4.39x) and Sum-of-max without caching (1.17x vs. 5.20x). For Sum-of-max with caching, it requires a huge memory index size (125x) to accomplish a similar latency to CMC. If a 125x index size is not available in practice, the speed becomes impractical introducing scalability limitations. This analysis implies that CMC's single-vector approach is significantly faster and more memory efficient, while still demonstrating a comparable ability to represent candidate information with fewer tokens, often surpassing more complex methods.

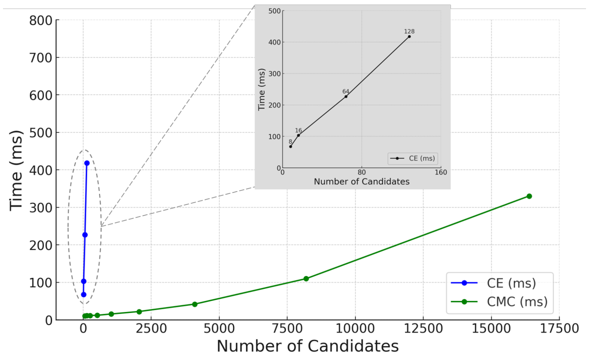

Figure 4: The relationship between the number of candidates and the corresponding time measurements in milliseconds for two different models: Cross-encoder (CE) and Comparing Multiple Candidates (CMC).

**Comparison with High-latency Models** Given the importance of computational resources and serving time in applications, CMC is a practical alternative to cross-encoders, with 11.02x speedup and comparable reranking accuracy. CMC outperforms the cross-encoder in the Wikipedia entity linking (+0.7p accuracy) and DSTC7 dialogue ranking (+3.3p MRR). Also, CMC presents a competitive result in MS MARCO and ZeSHEL dataset, achieving the second- or third-best in prediction. This comparison suggests that the self-attention layer in CMC effectively substitutes for the token-by-token interaction in cross-encoders while enhancing the computational efficiency of the reranking process.

In summary, to achieve the best accuracy, we recommend the 3-stage retrieval pipeline of bi-encoders + CMC + cross-encoders (BE-CMC-CE) that is both more accurate and substantially faster than the widely adopted bi-encoder + cross-encoder

(BE-CE), as shown in Table 2 and §4.3. If there's a time constraint, using CMC as the final reranker can be a good option since inferring with 16 candidates using a cross-encoder takes approximately the same amount of time as comparing around 10K candidates with CMC (Figure 4)

### 4.5 Ablation Study

Through the experiments, we notice an improved reranking performance on CMC when transferring the sentence encoder from another domain. To examine whether this is CMC-specific characteristic, we conduct an experiment that investigate how different sentence encoder initializations affect the performance of late-interaction models. For each model, we consider sentence encoder initializations with BERT-based bi-encoders fine-tuned for an in-domain (ZeSHEL; (Yadav et al., 2022)) and out-domain (MS-MARCO; (Guo and Barbosa, 2018)), as well as vanilla BERT (Devlin et al., 2018); then for each combination of model and sentence-encoder initialization, we fine-tune the model on ZeSHEL dataset and report its test set results.

In Table 4, different initialization strategies show different effects for each model. CMC and Poly-encoder show significant performance increases with out-of-domain sentence encoder initialization. This can be attributed to both models utilizing single candidate embeddings. Other models, such as Sum-of-max and MixEncoder, show negligible impact from sentence encoder initialization, whereas Deformer and Bi-encoder perform best with vanilla BERT. These findings suggest that CMC and the poly-encoder, which compress candidate information into single embeddings, can benefit from initialization from out-of-domain sentence encoders. As a practical recommendation, we advise practitioners to try out-of-domain initialization when using CMC for potentially improved performance.

## 5 Conclusion

In this paper, we present a novel and intuitive retrieval and reranking framework, Comparing Multiple Candidates (CMC). By contextualizing the representations of candidates through the self-attention layer, CMC achieves improvements in prediction performance with a marginal increase in speed and memory efficiency. Experimental results show that CMC acts as a seamless intermediate reranker between bi-encoders and cross-encoders. The retrieval pipeline of BE-CMC-CE is not only more accurate but also substantially faster than the widely adopted bi-encoder + cross-encoder (BE-CE). Meanwhile, experiments on four different datasets demonstrate that CMC can serve as the efficient final stage reranker. These empirical results emphasize CMC's effectiveness, marking it as a promising advancement in the field of neural retrieval and reranking.

| (Valid/Test) | | Sentence Encoder Initialization | | |
|---|---|---|---|---|
| | | Vanilla | Fine-tuned with | |
| | Model | BERT | In-domain<br>(ZeSHEL) | Out-of-domain<br>(MS MARCO) |
| Medium-<br>Latency | Deformer | **65.40/63.58** | 64.42/62.43 | 57.01/57.46 |
| | Sum-of-max | **59.57**/58.37 | 58.77/57.65 | 59.15/**58.79** |
| Low-<br>Latency | Bi-encoder | **55.54/52.94** | 55.54/52.94 | 49.32/44.01 |
| | Poly-encoder | 53.37/52.49 | 55.75/54.22 | **57.41/58.22** |
| | MixEncoder | **58.63/57.92** | 58.32/57.68 | 58.52/57.70 |
| | CMC (Ours) | 56.15/55.34 | 58.04/56.20 | **60.05/59.23** |

Table 4: Comparison of unnormalized accuracy on valid/test set of ZeSHEL over different sentence encoder initialization (Vanilla BERT (Devlin et al., 2018), Bi-encoder fine-tuned for in- (Yadav et al., 2022) and out-of-domain (Guo et al., 2020)) dataset. We denote the best case for each method as bold.

## Limitations

In the future, we plan to test the CMC's performance with over 1000 candidates with batch processing. It has not yet been extensively researched whether CMC can effectively retrieve from a large collection, e.g., a collection comprising more than 1 million candidates. Furthermore, we plan to tackle the issue that arises from the concurrent operation of both a bi-encoder and CMC index, which currently requires double the index size. This is a consequence of running two separate encoder models in parallel. To address this, we will investigate an end-to-end training scheme, thereby enhancing the practicality and efficiency of running both the Bi-encoder and CMC simultaneously.

## Acknowledgement

We thank Nishant Yadav for his helpful discussions and feedback. This work was supported in part by the National Research Foundation of Korea (NRF) grant (RS-2023-00280883, RS-2023-00222663) and New Faculty Startup Fund from Seoul National University, and with the aid of computing resources from Artificial Intelligence Industry Center Agency, Google cloud platform research credits, and the National Super computing Center with super computing resources including technical support (KSC-2023-CRE-0176).

## References

Dhruv Agarwal, Rico Angell, Nicholas Monath, and Andrew McCallum. 2022a. Entity linking via explicit mention-mention coreference modeling. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 4644–4658.

Sumit Agarwal, Suraj Tripathi, Teruko Mitamura, and Carolyn Rose. 2022b. Zero-shot cross-lingual open domain question answering. In Proceedings of the Workshop on Multilingual Information Access (MIA), pages 91–99.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. arXiv preprint arXiv:1611.09268.

Qingqing Cao, Harsh Trivedi, Aruna Balasubramanian, and Niranjan Balasubramanian. 2020. DeFormer: Decomposing pre-trained transformers for faster question answering. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 4487–4497, Online. Association for Computational Linguistics.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL), pages 708–716.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2020. Autoregressive entity retrieval. arXiv preprint arXiv:2010.00904.

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.

Luyu Gao and Jamie Callan. 2022. Unsupervised corpus aware language model pre-training for dense passage retrieval. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2843–2853.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. arXiv preprint arXiv:1909.10506.

Chulaka Gunasekara, Jonathan K Kummerfeld, Lazaros Polymenakos, and Walter Lasecki. 2019. Dstc7 task 1: Noetic end-to-end response selection. In Proceedings of the First Workshop on NLP for Conversational AI, pages 60–67.

Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. 2020. Accelerating large-scale inference with anisotropic vector quantization. In International Conference on Machine Learning, pages 3887–3896. PMLR.

Zhaochen Guo and Denilson Barbosa. 2018. Robust named entity disambiguation with random walks. Semantic Web, 9(4):459–479.

Shuguang Han, Xuanhui Wang, Mike Bendersky, and Marc Najork. 2020. Learning-to-rank with bert in tf-ranking. arXiv preprint arXiv:2004.08476.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In Proceedings of the 2011 conference on empirical methods in natural language processing, pages 782–792.

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 arXiv:1905.01969.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. arXiv preprint arXiv:2004.04906.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, pages 39–48.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. arXiv preprint arXiv:1906.00300.

Yi Liu, Yuan Tian, Jianxun Lian, Xinlong Wang, Yanan Cao, Fang Fang, Wen Zhang, Haizhen Huang, Denvy Deng, and Qi Zhang. 2023. Towards better entity linking with multi-view enhanced distillation. arXiv preprint arXiv:2305.17371.

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-shot entity linking by reading entity descriptions. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 3449–3460.

Xinyin Ma, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Weiming Lu. 2021. Muver: improving first-stage entity retrieval with multi-view entity representations. arXiv preprint arXiv:2109.05716.

Ida Mele, Cristina Ioana Muntean, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, and Ophir Frieder. 2020. Topic propagation in conversational search. In Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, pages 2057–2060.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. arXiv preprint arXiv:1901.04085.

Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. 2019a. Multi-stage document ranking with bert. arXiv preprint arXiv:1910.14424.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019b. Document expansion by query prediction. arXiv preprint arXiv:1904.08375.

Chen Qu, Liu Yang, Cen Chen, Minghui Qiu, W Bruce Croft, and Mohit Iyyer. 2020. Open-retrieval conversational question answering. In Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, pages 539–548.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2021. Colbertv2: Effective and efficient retrieval via lightweight late interaction. arXiv preprint arXiv:2112.01488.

Xiaoyu Shen, Svitlana Vakulenko, Marco Del Tredici, Gianni Barlacchi, Bill Byrne, and Adrià de Gispert. 2022. Low-resource dense retrieval for open-domain question answering: A comprehensive survey. arXiv preprint arXiv:2208.03197.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. Advances in neural information processing systems, 30.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 6397–6407.

Taiqiang Wu, Xingyu Bai, Weigang Guo, Weijie Liu, Siheng Li, and Yujiu Yang. 2023. Modeling fine-grained information via knowledge-aware hierarchical graph for zero-shot entity retrieval. In Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, pages 1021–1029.

Zhenran Xu, Yulin Chen, Baotian Hu, and Min Zhang. 2023. A read-and-select framework for zero-shot entity linking. arXiv preprint arXiv:2310.12450.

Nishant Yadav, Nicholas Monath, Rico Angell, Manzil Zaheer, and Andrew McCallum. 2022. Efficient nearest neighbor search for cross-encoder models using matrix factorization. arXiv preprint arXiv:2210.12579.

Yuanhang Yang, Shiyi Qi, Chuanyi Liu, Qifan Wang, Cuiyun Gao, and Zenglin Xu. 2023. Once is enough: A light-weight cross-attention for fast sentence pair modeling. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 2800–2806, Singapore. Association for Computational Linguistics.

Koichiro Yoshino, Chiori Hori, Julien Perez, Luis Fernando D'Haro, Lazaros Polymenakos, Chulaka Gunasekara, Walter S Lasecki, Jonathan K Kummerfeld, Michel Galley, Chris Brockett, et al. 2019. Dialog system technology challenge 7. arXiv preprint arXiv:1901.03461.

Wenzheng Zhang and Karl Stratos. 2021. Understanding hard negatives in noise contrastive estimation. arXiv preprint arXiv:2104.06245.

Yanzhao Zhang, Dingkun Long, Guangwei Xu, and Pengjun Xie. 2022. Hlatr: enhance multi-stage text retrieval with hybrid list aware transformer reranking. arXiv preprint arXiv:2205.10569.

# A Potential Risks

This research examines methods to accelerate the retrieval and reranking process using efficient and effective CMC. While the proposed CMC might exhibit specific biases and error patterns, we do not address these biases in this study. It remains uncertain how these biases might affect our predictions, an issue we plan to explore in future research.

# B Detailed Information of Datasets

**Wikipedia Entity Linking** For standard entity linking, we use AIDA-CoNLL dataset (Hoffart et al., 2011) for in-domain evaluation, and WNED-CWEB (Guo and Barbosa, 2018) and MSNBC (Cucerzan, 2007) datasets for out-of-domain evaluation. These datasets share the same Wikipedia knowledge base. For comparison with the baseline results from Wu et al. (2020), we employ the 2019 English Wikipedia dump, containing 5.9M entities. We employed a bi-encoder as an initial retriever that yields an average unnormalized accuracy of 77.09 and a recall@10 of 89.21. Unnormalized accuracy is measured for each dataset and macro-averaged for test sets.

AIDA-CoNLL dataset is licensed under a Creative Commons Attribution-ShareAlike 3.0 Unported License. We are not able to find any license information about WNED-CWEB and MSNBC datasets.

**Zero-shot Entity Linking (ZeSHEL)** ZeSHEL (Logeswaran et al., 2019) contains mutually exclusive entity sets between training and test data. The dataset comprises context sentences (queries) each containing a mention linked to a corresponding gold entity description within Wikia knowledge base. Unlike Wikipedia entity linking datasets where the entity set of train and test set overlaps, the entity set for ZeSHEL is mutually exclusive and this setup tests the model's ability to generalize to new entities. We employed a bi-encoder from (Yadav et al., 2022) whose recall@64 is 87.95. On top of these candidate sets, we report macro-averaged unnormalized accuracy, which is calculated for those mention sets that are successfully retrieved by the retriever and macro-averaged across a set of entity domains. For statistics of entity linking datasets, see Table 5. ZeSHEL is licensed under the Creative Commons Attribution-Share Alike License (CC-BY-SA).

The predominant approach for reranking in ZeSHEL dataset is based on top-64 candidate sets from official BM25 (Logeswaran et al., 2019) or bi-encoder (Wu et al., 2020; Yadav et al., 2022). On top of these candidate sets, we report macro-averaged normalized accuracy, which is calculated for those mention sets that are successfully retrieved by the retriever and macro-averaged across a set of entity domains.

| Dataset | | # of Mentions | # of Entities |
|---|---|---|---|
| AIDA | Train | 18848 | 5903530 |
| | Valid (A) | 4791 | |
| | Valid (B) | 4485 | |
| MSNBC | | 656 | |
| WNED-WIKI | | 6821 | |
| ZeSHEL | Train | 49275 | 332632 |
| | Valid | 10000 | 89549 |
| | Test | 10000 | 70140 |

Table 5: Staistics of Entity Linking datasets.

**MS MARCO** We use a popular passage ranking dataset MS MARCO which consists of 8.8 million web page passages. MS MARCO originates from Bing's question-answering dataset with pairs of queries and passages, the latter marked as relevant if it includes the answer. Each query is associated with one or more relevant documents, but the dataset does not explicitly denote irrelevant ones, leading to the potential risk of false negatives. For evaluation, models are fine-tuned with approximately 500K training queries, and MRR@10, Recall@1 are used as a metric. To compare our model with other baselines, we employed Anserini's BM25 as a retriever (Nogueira et al., 2019b). The dataset is licensed under Creative Commons Attribution 4.0 International.

**DSTC 7 Challenge (Track 1)** For conversation ranking datasets, we involve The DSTC7 challenge (Track 1) (Yoshino et al., 2019) . DSTC 7 involves dialogues taken from Ubuntu chat records, in which one participant seeks technical assistance for diverse Ubuntu-related issues. For these datasets, an official candidate set which includes gold is provided. For statistics for MS MARCO and DSTC 7 Challenge, see Table 6

# C Training Details

**Negative Sampling** Most of previous studies that train reranker (Wu et al., 2020; Xu et al., 2023) employ a fixed set of top-$k$ candidates from the

| Datasets | Train | Valid | Test | # of Candidates per Query |
|---|---|---|---|---|
| MS MARCO | 498970 | 6898 | 6837 | 1000 |
| DSTC 7 | 100000 | 10000 | 5000 | 100 |

Table 6: Statistics of MS MARCO & Conversation Ranking Datasets.

retriever. In contrast, our approach adopts hard negative sampling, a technique derived from studies focused on training retrievers (Zhang and Stratos, 2021). Some negative candidates are sampled based on the retriever's scoring for query-candidate pair $(q, c_{q,j})$:

$$\forall j \in \{1, \dots, K\} \setminus \{\text{gold index}\}, \quad \tilde{c}_{q,j} \sim \frac{\exp(s_{\text{retriever}}(q, \tilde{c}_{q,j}))}{\sum_{\substack{k=1 \\ k\neq \text{gold index}}}^{K} \exp(s_{\text{retriever}}(q, \tilde{c}_{q,k}))} \tag{7}$$

To provide competitive and diverse negatives for the reranker, p% of the negatives are fixed as the top-$k$ negatives, while the others are sampled following the score distribution.

As detailed in Table 7, we implement a hard negative mining strategy for training CMC and comparable baseline methods. Specifically, for the MS MARCO dataset, hard negatives are defined as the top 63 negatives derived from the CoCondenser model, as outlined in Gao and Callan (2022). In the case of entity linking datasets, we adhere to the approach established by Zhang and Stratos (2021), where hard negatives are selected from the top 1024 candidates generated by a bi-encoder. Meanwhile, for dialogue ranking datasets, we do not employ hard negative mining, owing to the absence of candidate pool within these datasets.

**Sentence Encoder Initialization** The initial starting point for both the query and candidate encoders can significantly impact performance. The sentence encoders for late interaction models including CMC are initialized using either vanilla huggingface BERT (Devlin et al., 2018) or other BERT-based, fine-tuned models. These models include those fine-tuned on the Wikipedia dataset (BLINK-bi-encoder; Wu et al. (2020)) or MS MARCO (Cocondenser; Gao and Callan (2022)). As the cross-encoder is the only model without sentence encoder, we initialize cross-encoder using pre-trained BERT (BLINK-cross-encoder; Wu et al. (2020)) or vanilla BERT.

We initialize the sentence encoder for CMC and other baselines using (1) vanilla BERT and (2) the BLINK bi-encoder for Wikipedia entity linking datasets, and the MS-MARCO fine-tuned Cocondenser for other datasets. After conducting experiments with both starting points, we selected the best result among them. If more favorable results for baselines are found from prior works that conduct reranking over the same candidates, we sourced the numbers from these works.

**Optimization** Our model employs multi-class cross-entropy as the loss function, regularized by Kullback-Leibler (KL) divergence between the reranker's scores and the retriever's scores. The loss function is formulated as follows:

$$\begin{aligned}\mathcal{L}(q, \tilde{C}_q) = -\lambda_1 \sum_{i=1}^{K} y_i \log(p_i) \\ + \lambda_2 \sum_{i=1}^{K} p_i \log\left(\frac{p_i}{r_i}\right)\end{aligned} \tag{8}$$

For the query $q$, $y_i$ represents the ground truth label for each candidate $\tilde{c}_{q,i}$, $p_i$ is the predicted probability for candidate $\tilde{c}_{q,i}$ derived from the score function $s_\theta$, $r_i$ is the probability of the same candidate from the retriever's distribution, and $\lambda_1$ and $\lambda_2$ are coefficients forming a convex combination of the two losses.

**Extra Skip Connection** CMC is trained end-to-end, where the self-attention layer is trained concurrently with the query and candidate encoders. In addition to the inherent skip connections present in the transformer encoder, we have introduced an extra skip connection following He et al. (2016) to address the vanishing gradient problem commonly encountered in deeper network layers. Specifically, for an encoder layer consisting of self-attention layer $\mathcal{F}(\mathbf{x})$, the output is now formulated as $\mathbf{x} + \mathcal{F}(\mathbf{x})$, with $\mathbf{x}$ being the input embedding. This training strategy ensures a more effective gradient flow during backpropagation, thereby improving the training stability and performance of our model.

# D Additional Results and Analysis

## D.1 Reranking Latency of cross-encoders and CMC

In Figure 4, we present the plot of runtime against the number of candidates. For CMC, the model can handle up to 16,384 candidates per query, which is comparable to the speed of cross-encoders for running 64 candidates. Running more than 128 and

|  | Entity Linking<br>AIDA-train | <br>ZeSHEL | Passage Ranking<br>MS MARCO | Dialogue Ranking<br>DSTC7 |
|---|---|---|---|---|
| max. query length | 32 | 128 | 32 | 512 |
| max. document length | 128 | 128 | 128 | 512 |
| learning rate | {**1e-5**,5e-6,2e-6} | {**1e-5**,2e-5,5e-5} | {1e-5,5e-6,**2e-6**} | {**1e-5**,2e-5,5e-5} |
| batch size | 4 | 4 | 8 | 8 |
| hard negatives ratio | 0.5 | 0.5 | 1 | - |
| # of negatives | 63 | 63 | 63 | 7 |
| training epochs | 4 | 5 | 3 | 10 |

Table 7: Hyperparameters for each dataset. We perform a grid search on learning rate and the best-performing learning rate is indicated as bold.

|  | Test |  |  |  |  |  | Valid |  |
|---|---|---|---|---|---|---|---|---|
| Method | R@1 | R@4 | R@8 | R@16 | R@32 | R@64 | R@1 | R@64 |
| Bi-encoder | 52.94 | 64.51 | 71.94 | 81.52 | 84.98 | 87.95 | 55.45 | 92.04 |
| BE + CMC(64) | **59.22** | **77.69** | 82.45 | 85.46 | 87.28 | 87.95 | **60.27** | 92.04 |
| BE + CMC(128) | 59.13 | 77.65 | 82.72 | 85.84 | 88.29 | 89.83 | 60.24 | 93.22 |
| BE + CMC(256) | 59.13 | 77.6 | 82.86 | 86.21 | 88.96 | 90.93 | 60.13 | 93.63 |
| BE + CMC(512) | 59.08 | 77.58 | **82.91** | **86.32** | **89.33** | **91.51** | 60.1 | **93.89** |

Table 8: Retrieval performance by the number of candidates from the initial retriever. The numbers in parentheses (e.g., 128 for CMC(128)) indicate the number of candidates which CMC compares, initially retrieved by the bi-encoder. The best result is denoted in bold and the second-best result is underlined.

16,384 candidates cause memory error on GPU for cross-encoders and CMC, respectively.

### D.2 Effect of Number of Candidates on Retrieval Performance

In Table 8, we present detailed results of retrieval performance on varying numbers of candidates from the initial bi-encoder. Recall@k increased with a higher number of candidates. It indicates that CMC enables the retrieval of gold instances that could not be retrieved by a bi-encoder, which prevents error propagation from the retriever. It is also noteworthy that CMC, which was trained using 64 candidates, demonstrates the capacity to effectively process and infer from a larger candidate pool (256 and 512) while giving an increase in recall@64 from 82.45 to 82.91.

### D.3 Detailed Information of Entity Linking Performance

In Table 9, we present detailed results for each dataset in Wikipedia entity linking task. Also, in table 10, we present detailed results for each world in ZeSHEL test set.

|  | Method | Valid (A) | Test (B) | MSNBC* | WNED-CWEB* | Average |
|---|---|---|---|---|---|---|
| High-Latency | Cross-encoder | 82.12 | 80.27 | 85.09 | 68.25 | 77.87 |
|  | Cross-encoder † | 87.15 | 83.96 | 86.69 | 69.11 | 80.22 |
| Intermediate-Latency | Sum-of-max † | 90.84 | **85.30** | 86.07 | **70.65** | 80.67 |
|  | Deformer† | 90.64 | 84.57 | 82.92 | 66.97 | 78.16 |
| Low-Latency | Bi-encoder | 81.45 | 79.51 | 84.28 | 67.47 | 77.09 |
|  | Poly-encoder† | 90.64 | 84.79 | 86.30 | 69.39 | 80.16 |
|  | MixEncoder† | 89.92 | 82.69 | 78.24 | 64.00 | 76.27 |
|  | CMC† | **91.16** | 85.03 | **87.35** | 70.34 | **80.91** |

Table 9: Unnormalized accuracy on Wikipedia entity linking dataset (AIDA (Hoffart et al., 2011), MSNBC (Cucerzan, 2007), and WNED-CWEB (Guo and Barbosa, 2018)). *Average* means macro-averaged accuracy for three test sets. The best result is denoted in bold and the second best result is denoted as underlined. * is out of domain dataset. † is our implementation.

|  |  | Valid | Test (By Worlds) |  |  |  |  |
|---|---|---|---|---|---|---|---|
|  | Method |  | Forgotten Realms | Lego | Star Trek | Yugioh | Avg. |
| High-Latency | Cross-encoder | 67.41 | 80.83 | 67.81 | 64.23 | 50.62 | 65.87 |
|  | Cross-encoder (w/ CMC) | 70.22 | 81.00 | 67.89 | 64.42 | 50.86 | 66.04 |
| Intermediate-Latency | Sum-of-max | 59.15 | 73.45 | 58.83 | 57.63 | 45.29 | 58.80 |
|  | Deformer | 56.95 | 73.08 | 56.98 | 56.24 | 43.55 | 57.46 |
| Low-Latency | Bi-encoder | 55.45 | 68.42 | 51.29 | 52.66 | 39.42 | 52.95 |
|  | Poly-encoder | 57.19 | 71.95 | 58.11 | 56.19 | 43.60 | 57.46 |
|  | MixEncoder | 58.64 | 73.17 | 56.29 | 56.99 | 43.01 | 57.36 |
|  | CMC(Ours) | 60.05 | 73.92 | 58.96 | 58.08 | 45.69 | 59.16 |

Table 10: Detailed Reranking Performance on Zero-shot Entity Linking (ZeSHEL) valid and test set (Logeswaran et al., 2019). Macro-averaged unnormalized accuracy is measured for candidates from Bi-encoder (Yadav et al., 2022).The best result is denoted in **bold**.

### D.4 Ranking Performance on ZeSHEL BM25 candidate sets

In many previous works (Wu et al., 2020; Xu et al., 2023), the performance of models over BM25 candidates (Logeswaran et al., 2019) has been reported. In Table 11, we present the performance of CMC to illustrate its positioning within this research landscape.

| Methods | Forgotten Realms | Lego | Star Trek | Yugioh | Macro Acc. | Micro Acc. |
|---|---|---|---|---|---|---|
| Cross-encoder (Wu et al., 2020) | 87.20 | 75.26 | 79.61 | 69.56 | 77.90 | 77.07 |
| ReS (Xu et al., 2023) | 88.10 | 78.44 | 81.69 | 75.84 | 81.02 | 80.40 |
| ExtEnD (De Cao et al., 2020) | 79.62 | 65.20 | 73.21 | 60.01 | 69.51 | 68.57 |
| GENRE (De Cao et al., 2020) | 55.20 | 42.71 | 55.76 | 34.68 | 47.09 | 47.06 |
| Poly-encoder† | 78.90 | 64.47 | 71.05 | 56.25 | 67.67 | 66.81 |
| Sum-of-max† | 83.20 | 68.17 | 73.14 | 64.00 | 72.12 | 71.15 |
| Comparing Multiple Candidates (Ours) | 83.20 | 70.63 | 75.75 | 64.83 | 73.35 | 72.41 |

Table 11: Test Normalized accuracy of CMC model over retrieved candidates from BM25. * is reported from Xu et al. (2023). † is our implementation.

| Methods | w/ bi-encoder retriever<br>Valid | <br>Test | w/ BM25 retriever<br>Test |
|---|---|---|---|
| CMC | 65.29 | **66.83** | **73.10** |
| w/o extra skip connection | 64.78 | 66.44 | 73.07 |
| w/o regularization | 64.45 | 66.31 | 72.94 |
| w/o sampling | **65.32** | 66.46 | 72.97 |

Table 12: Normalized Accuracy on ZeSHEL test set for various training strategies

### D.5 Ablation Study on Training Strategies

In Table 12, we evaluated the impact of different training strategies on the CMC's reranking performance on the ZeSHEL test set. The removal of extra skip connections results in only a slight decrease ranging from 0.03 to 0.39 points in normalized accuracy. Also, to examine the effects of a bi-encoder retriever, we remove regularization from the loss. It leads to a performance drop but still shows higher performance than sum-of-max, the most powerful baseline in the low latency method. Lastly, we tried to find the influence of negative sampling by using fixed negatives instead of mixed negatives. The result shows a marginal decline in the test set, which might be due to the limited impact of random negatives in training CMC.

### D.6 Reranking Performance of Cross-encoders for Various Number of Candidates

In Table 13, we evaluated the impact of the different number of candidates on the cross-encoder's reranking performance on the ZeSHEL test set with a candidate set from the bi-encoder retriever. Even with a larger number of candidates, the unnormalized accuracy of the cross-encoder does not increase. Although the number of candidates from the bi-encoder increases from 64 to 512, recall@1 decreases by 0.01 points.

| # of candidates | Recall@1<br>(Unnormalized Accuracy) |
|---|---|
| 16 | 65.02 |
| 64 | 65.87 |
| 512 | 65.85 |

Table 13: Normalized Accuracy on ZeSHEL test set for various training strategies