# FINAL ASSIGNMENT: Cross-Encoder re-Rankers and Query Expansion with an LLM

ISIS VAN WOLDE (S2888114) and EKANSH KHANULIA (S4336089)*

CCS Concepts: • **Information systems** → **Language models**; *Relevance assessment*; *Combination, fusion and federated search.*

## Introduction

Cross-Encoder Re-Rankers are powerful transformer based models that are used to analyze and accurately score the relevance of documents and queries. Within the search process a Cross-Encoder Re-Ranker comes into play in the second stage, the re-ranking stage. In this stage a the top results of a previously quickly estimated determined ranking, using for example a term matching model like BM25, are re-evaluated with a Cross-Encoder Re-Ranker to create a more accurate ranking based on a deeper semantic understanding of the query.

In order to get a better understanding of the workings of these Cross-Encoder Re-Rankers and their accuracy and efficiency we will be finetuning and evaluating 3 different Cross-Encoder Re-Rankers. Then, using these, 5 ensemble methods will be used to combine the results from the Cross-Encoder Re-Rankers to create the most accurate result and get a grip on the workings of such ensemble methods.

## 1 Task 1

In order to understand and evaluate a Cross-Encoder Re-Ranker it first needs to be trained on a test dataset. The data set used for the training is the MS Marco dataset. The three models we trained were: miniLLM, tinyBERT and Roberta. We trained each model for 1 hour total. With the trained models we can evaluate their performance. The evaluation was done based on 43 queries from TREC DL'19. The NDCG@10, Recall@100 and MAP@1000 were noted and for each model these are presented in table 1 along with the number of steps used for the training of the model.

| # | Model | Training Steps | Recall@100 | MAP@1000 | NDCG@10 |
|---|---|---|---|---|---|
| 1 | miniLLM | 56,998 | 49.74 | 43.97 | 67.92 |
| 2 | tinyBERT | 70,038 | 49.37 | 43.97 | 67.05 |
| 3 | Roberta | 12,000 | 49.75 | 41.89 | 63.17 |

Table 1. Performance of Three Models on IR Evaluation Metrics

From these results we see that the models excel for different metrics. For Recall@100 Roberta is the best performer. This has to do with the large size of about 125 million parameters, allowing recognizing complex semantic relations that enable generalization with fewer training steps of 12000 steps. Compared to miniLLM and tinyBERT, Roberta might have a higher recall due to the deeper understanding increasing the chance of retrieving at least one relevant document. As for the MAP@1000 metric both miniLLM and tinyBERT perform best. MiniLLM and tinyBERT both have smaller sizes and can process more steps in the same time. This enables these two models to efficiently differentiate between relevant and irrelevant documents, because of their exposure to more examples allowing these models to learn a larger collection of negative examples. Then the NDCG@10 points to miniLLM as the best performer, where miniLLM balances the number of semantic understanding and efficiency. With 22 million parameters miniLLM is still small enough to allow for a considerable number of training steps while still maintaining efficiency. Whereas tinyBERT has reduced learning efficiency due to its even larger number of steps and Roberta's lack of a decent amount of training steps hinders a accurate fine-tuning of the top of the ranking.

*Both authors contributed equally to this research.

While scoring high on these benchmark metrics indicates a high performance, it does not necessarily translate to a excelling efficiency across different queries, due to the diversity of the training data set. MS Marco is based on fact style questions, so our trained models might struggle with opinion based or domain specific queries. Besides this when focusing on one specific metric there is a risk of overfitting. For example optimizing recall tends to pull in many documents losing precision and accuracy. So, also considereing the previously mentioned tradeoff between step numbers and capacity, just using the metrics used on the MS Marco dataset is not enough to draw a conclusion on the best performing model. It takes broader variety within a dataset to test the excellence of a model beyond the original training distribution.

Models of larger size like Roberta take more time for each iteration allowing, so more GPU is needed. On the other hand miniLLM and tinyBERT can process much faster, but often miss the subtle semantic linguistic patterns. When the models are applied to queries outside the MS Marco set, the performance drops significantly, so hybrid retrieval and further fine tuning and training is needed for better performance. Another disadvantage is that they can cause latency on a larger scale, like a websearch, because they calculate a score for every document to a query. Unless multiple machines are used or cheaper filters are applied beforehand, this creates problematically low time efficiency. Also the training of these large models to keep them up to date is expensive. Lastly, the induced bias from the training data these Cross-Encoders have poses a problem to the fair treatment of different subjects and user groups.

## 2 Task 2

In order to increase model performance one can opt to combine multiple models. Models have different excel points, so by combining these strong points of the different Cross-Encoders their complementary properties can be exploited. In order to find the best combination of models we will use 5 ensemble methods. The evaluation of these methods is presented in table 2.

| # | Ensemble Method | Recall@100 | MAP@1000 | NDCG@10 |
|---|---|---|---|---|
| 1 | CombANZ | 51.18 | 45.55 | 69.54 |
| 2 | CombSUM | 51.18 | 45.55 | 69.54 |
| 3 | CombMNZ | 51.18 | 45.55 | 69.54 |
| 4 | CombMIN | 50.20 | 43.76 | 68.88 |
| 5 | CombMAX | 50.38 | 43.15 | 64.83 |

Table 2. Performance of Five Ensemble Methods on IR Evaluation Metrics

Table 2 shows that CombANZ, CombSUM, and CombMNZ yield identical gains, reflecting a strong consensus among the models on top-ranked documents. CombMIN is more conservative—any model's low confidence vetoes a document—causing its slight performance drop. CombMAX fares worst: by taking the highest score, it elevates passages that only one model favors, injecting noise into the ranking.

Sum-based methods (CombSUM, CombANZ, CombMNZ) perform best by reinforcing documents that all models rank highly, amplifying consensus and smoothing out individual errors. CombANZ further divides by a document's rank frequency, and CombMNZ multiplies by it—both softly normalizing scores so no single model dominates. By contrast, CombMIN uses the lowest model score, which vetoes any document a model rates poorly and thus lowers recall, while CombMAX uses the highest score, allowing a single model's overenthusiastic ranking to introduce false positives. These trade-offs explain why sum-based fusion outperforms the more extreme min- and max-based approaches.

Ensembling markedly improves performance at minimal development cost—no additional training data or architecture changes are needed—and leverages each model's strengths to mitigate individual errors. By relying on consensus, these methods guard against outlier predictions on atypical queries. The trade-off is higher inference

cost, since every model must score each candidate; in practice, fusion may be best applied after an initial top-k filtering step to balance accuracy and efficiency

## 3 Task 3

In order to determine the best combination of models the CombSUM method is used. CombSUM is then applied to all possible combinations of our three Cross-Encoders. The performance metrics NDCG@10, Recall@100 and MAP@1000 for the tree possible combinations of our models are presented in table 3.

| # | Model combination | Recall@100 | MAP@1000 | NDCG@10 |
|---|---|---|---|---|
| 1 | miniLLM & tinyBERT | 50.61 | 45.56 | 69.66 |
| 2 | miniLLM & DistilRoBERTa | 50.71 | 44.60 | 68.56 |
| 3 | tinyBERT & DistilRoBERTa | 50.34 | 44.43 | 68.08 |

Table 3. Performance of Three Pairs of Models on IR Evaluation Metrics

From table 3 we see that for both MAP@1000 and NDCG@10 the combination of miniLLM and tinyBERT scores the highest and it's recall@100 score is close to the score of miniLLM and Roberta, making the top combination in the table the best performing combination. Besides these models being individually the strongest re-rankers seeing from their NDCG@10 score in table 1. They also complement each other. TinyBERT finds relevant passages missed by miniLLM due to its lighter and distilled architecture and the other way around. So their combination cancels out idiosyncratic errors yielding the highest MAP@1000 and NDCG@10 scores. From table 3 we see that tinyBERT combined with Roberta performs the worst, this is due to the shared lacking aspects and their individual mediocre scores.

In table 3 the NDGC@10 scores range over about 1.6 points and the MAP@1000 scores difference about 1.1 points. While these differences are not massive, their influence on high-performing ranking systems is. Since all models are trained on the same MS Marco set explaining the seemingly limited gains. These differences reflect how the models complement and overlap with each other. One thing stood out, miniLLM and Roberta score slightly higher in recall, this could be due to Roberta finding more subtle documents that are missed by both miniLLM and tinyBERT.

So as already suspected combining models using ensemble methods yields better performance. This is confirmed from the results showing that diverse models with differences in model size, distillation and embedding space is crucial.

## 4 Task 4

Task 4 uses Zephyr-7B-beta to generate Chain-of-Thought query expansions for a fixed set of 43 test queries and their top-1K candidates, ensuring each reformulated query stays under BERT's 512-token limit. We then rerank with TinyBERT before and after expansion and compare NDCG@10, Recall@100, and MAP@1000 to assess how LLM-driven reformulation impacts retrieval effectiveness .In this task we use two prompt to compare effectiveness

Prompt 1:

```
"Identify the central topics and synonyms for this search query:"
{query}
"List 5–8 comma-separated keywords:"
```

Prompt 2:

```
"Query: {query}\n"
"What could be the intent behind this query?"
```

Evaluating the TinyBERT on the 43 queries revealed that the baseline achieved better results in table 4. Applying Prompt 1 (table 4 row 2nd) and truncating to the first 50 words caused performance to fall sharply

, while Prompt 2(table 4 row third) yielded a smaller drop. Prompt 1's broad thematic lists lacked the precise, discriminative terms that the re-ranker requires, whereas Prompt 2's intent-focused framing elicited somewhat more targeted paraphrases. In both cases, truncating at 50 words often cut off high-value keywords. These results diverge from Jagerman *et al.*'s reported gains—obtained using their exact Chain-of-Thought keyword template within a two-stage BM25 + re-rank pipeline—because single-stage re-ranking amplifies noisy or verbose expansions and coarse truncation discards the most informative generated terms. Aligning exactly with the paper's CoT prompt, extracting only generated keywords, truncating by token budget rather than word count, and restoring a BM25 retrieval first stage should be adopted to realize the positive lifts in evaluation metrics demonstrated in the literature.

| # | Model | Recall@100 | MAP@1000 | NDCG@10 |
|---|---|---|---|---|
| 1 | Original | 51.28 | 46.17 | 69.84 |
| 2 | CoT e.g 1 | 43.13 | 36.43 | 54,13 |
| 3 | Cot e.g 2 | 45.9 | 41.71 | 62.96 |

Table 4. TinyBERT performance on 43 queries: baseline vs. two CoT-based query expansions on evaluation metrics

## 5 Task 5

Pseudo-Relevance Feedback (PRF) refines a query by treating its top-retrieved documents as relevant and extracting new terms from them. In Task 5, the top 3 passages for each of the 43 queries are combined with the original query in the CoT/PRF prompt to produce reformulations via an LLM. These expanded queries are then re-ranked by the TinyBERT cross-encoder. The aim is to quantify how PRF-augmented LLM expansion affects NDCG@10, Recall@100, and MAP@1000, relative to both the unexpanded baseline and the pure CoT reformulation without PRF. We use the following prompt for this task :

Prompt:

```
"Answer the following query based on the context:"
"Context:" {docs[0]}, {docs[1]}, {docs[2]}
"Query:"   {query}
"Give the rationale before answering"
```

Contrary to the paper's findings, where CoT prompts (especially with PRF context) yielded modest gains in recall and top-heavy metrics, both the CoT and CoT+PRF expansions here markedly degraded performance. The standalone CoT expansion undercuts the baseline by 8–5 points across metrics, while the PRF-enhanced variant collapses to single-digit scores. This divergence likely stems from several implementation and modeling factors. First the Zephyr-7B model generates overly verbose rationales rather than focused expansion terms, and truncating to the final line may have captured justification instead of actual keywords. Second, using only three context passages without fine-tuning the prompt to distill salient terms can mislead the LLM. Third the TinyBERT 512-token limit may still have been exceeded or misaligned by the expanded queries, reducing re-ranking effectiveness. And fourth the original paper's experiments employ much larger, instruction-tuned Flan models whose richer representations better support query expansion. Thus, both model choice and prompt engineering critically influence whether LLM-driven query reformulations improve or impair downstream re-ranking.

| # | Model | Recall@100 | MAP@1000 | NDCG@10 |
|---|---|---|---|---|
| 1 | Original | 51.28 | 46.17 | 69.84 |
| 2 | CoT | 47.13 | 43.11 | 61.46 |
| 3 | Cot/PRF | 16.02 | 12.29 | 9.28 |

Table 5. Evaluation on 43 queries: baseline vs. CoT vs. CoT+PRF for NDCG@10, Recall@100, and MAP@1000

## References