Evaluating Token-Level and Passage-Level Dense Retrieval Models for Math Information Retrieval

Wei Zhong, Jheng-Hong Yang, Yuqing Xie, and **Jimmy Lin**

David R. Cheriton School of Computer Science
University of Waterloo

{w32zhong,jheng-hong.yang,yuqing.xie,jimmylin}@uwaterloo.ca

Abstract

With the recent success of dense retrieval methods based on bi-encoders, studies have applied this approach to various interesting downstream retrieval tasks with good efficiency and in-domain effectiveness. Recently, we have also seen the presence of dense retrieval models in Math Information Retrieval (MIR) tasks, but the most effective systems remain classic retrieval methods that consider handcrafted structure features. In this work, we try to combine the best of both worlds: a well-defined structure search method for effective formula search and efficient bi-encoder dense retrieval models to capture contextual similarities. Specifically, we have evaluated two representative bi-encoder models for token-level and passage-level dense retrieval on recent MIR tasks. Our results show that bi-encoder models are highly complementary to existing structure search methods, and we are able to advance the state-of-the-art on MIR datasets.

1 Introduction

Math Information Retrieval (MIR) is a special information retrieval domain that deals with heterogeneous data. The core task in this field is to retrieve relevant information from documents that contain math formulas. As digitized math content (mostly in LaTeX markup or MathML format) becomes readily available nowadays, being able to index and retrieve formulas (or equations) in those documents effectively is possibly one of the hard nuts left to be cracked before we can search freely for scientific documents, educational materials, and other math content.

The need to measure similarities for highly structured formulas with special semantic properties and to model their connections to the surrounding text have a few interesting consequences: (1) Heuristic scores like the term-frequency factor in tf–idf scoring variants become less relevant in formula similarity assessment because symbols in a math formula can be interchangeable and similarity may depend on expression structure rather than frequency of co-occurrence. (2) At the same time, the same math content can be expressed differently, e.g., $\{1, 2, ...\}$ and $\mathbb{N}^+$ represent the same concept but they are made of totally different tokens. We also need to capture similar math expressions with different structure due to math transformations, e.g., $1 + \frac{1}{x}$ and $\frac{1+x}{x}$. These have made structure search approaches alone suboptimal. (3) Many existing methods score math and text separately because they are of different modalities; however, failing to catch cross references between text and math will penalize retrieval effectiveness. For example, a top effective math-aware search engine adopting traditional ad-hoc search techniques (Fraser et al., 2018; Ng et al., 2020, 2021) tunes a hyperparameter to weight text and formulas in two separate passes, which provides little awareness of the connections between formulas and their surrounding text. The aforementioned challenges have put limitations on further advances in this field.

On the other hand, recent bi-encoder dense retrieval models (Karpukhin et al., 2020; Santhanam et al., 2021; Hofstätter et al., 2021; Formal et al., 2021; Gao and Callan, 2021) have been shown to be highly effective for in-domain retrieval while remaining efficient for large corpora in practice. Compared to traditional retrieval methods, these models use dual deep encoders, usually built on top of a Transformer encoder architecture (Vaswani et al., 2017; Devlin et al., 2019), to encode query and document passages separately and eventually output contextual embeddings. Similarity scores can be efficiently computed given these embeddings, which limits costly neural inference to indexing time. The effectiveness of these models can be attributed to the encoder's ability to capture contextual connections or even high-level semantics without the necessity for exact lexical matching. This very complementary benefit compared to

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 1092–1102
December 7-11, 2022

more rigorous structure search methods motivates us to investigate whether dense retrieval models can improve MIR results when combined with existing structure search methods. We summarize the contributions of this work as follows:

- We have performed a fair effectiveness comparison of a token-level and a passage-level dense retrieval baseline in the MIR domain. To our knowledge, this is the first time that a DPR model has been evaluated in this domain.
- We have successfully combined dense retrievers with a structure search system and have been able to achieve new state-of-the-art effectiveness in recent MIR datasets.
- A comprehensive list of dense retrievers and strong baselines for major MIR datasets are covered and compared. We believe our well-trained models and data pipeline[1] can serve as a stepping stone for future research in this domain, which suffers from a scarcity of resources.

2 Background and Related Work

2.1 Classic and Structure Search

Research on math information retrieval started with the DLMF project from NIST decades ago (Miller and Youssef, 2003). Naturally, early studies (Miller and Youssef, 2003; Youssef, 2005) directly converted math symbols to textualized tokens (e.g., "+" will be converted to "plus") so they can be easily retrieved with existing IR systems. Later, a line of studies (Hijikata et al., 2009; Sojka and Líška, 2011; Lin et al., 2014; Zanibbi et al., 2015; Kristianto et al., 2016; Fraser et al., 2018) utilizing full-text search engines additionally introduced various intermediate tree representations to extract features that capture more structure information.

The MathDowsers system (Fraser et al., 2018; Ng et al., 2020, 2021; Ng, 2021) stands out in retrieval effectiveness by the incorporation of a mature full-text search engine and a curated list of over 5 types of features extracted from the Symbol Layout Tree (SLT) representation (Zanibbi and Blostein, 2012). Other features like the leaf-root paths extracted from the Operator Trees or representational MathML DOMs are also popular among researchers (Hijikata et al., 2009; Yokoi and Aizawa, 2009; Zhong, 2015; Zhong and Fang, 2016); these features are invariant to operand position mutation (e.g., due to commutativity) and require less storage. More strict and top-down approaches (Kohlhase et al., 2012; Schellenberg et al., 2012; Zanibbi et al., 2016b; Zhong and Zanibbi, 2019; Mansouri et al., 2020) have also been proposed by evaluating well-defined math formula structure similarity or edit distance, resulting in higher precision in top-ranked results generally. Furthermore, Zhong et al. (2020) have shown that a top-down structure search method can be accelerated to achieve practically efficient first-stage retrieval as well.

2.2 Data-Driven Methods

More recently, data-driven approaches that incorporate word embeddings (Gao et al., 2017; Mansouri et al., 2019), GNNs (Song and Chen, 2021), or Transformer models (Peng et al., 2021; Reusch et al., 2021a,b) have also been proposed for the MIR domain. By observing token co-occurrence and structure features during training, these models can discover synonyms or high-level semantic similarities, making them a good enhancement to strict structure matching. However, previous Transformer-based retrievers in this domain (Mansouri et al., 2021a; Reusch et al., 2021a,b) either only evaluate partial collections due to the adoption of expensive cross encoders, or cover only a token-level bi-encoder retriever, i.e., using the ColBERT model (Khattab and Zaharia, 2020; Santhanam et al., 2021). The effectiveness of a fine-tuned bi-encoder Transformer retriever for passage-level semantic similarity remains unknown.

In this work, we examine the DPR model (Karpukhin et al., 2020) as a passage-level dense retriever baseline for the MIR domain. We also fine-tune a ColBERT model (Khattab and Zaharia, 2020) that greatly outperforms the same type of models previously described in this domain. Furthermore, previous efforts (Mansouri et al., 2019; Peng et al., 2021) to consider structure features using data-driven models have achieved good levels of effectiveness; we will follow this path and evaluate the combination of structure-matching methods and dense retrieval.

Finally, some previous effective cross-encoder math retrieval runs (Reusch et al., 2021b) are based on further-pretrained backbone models in this domain. However, this *domain-adaptive pretraining*

[1] Our model checkpoints and source code are made publicly available: https://github.com/approach0/math-dense-retrievers/tree/emnlp2022

(DAPT) (Gururangan et al., 2020) shows inconsistent benefits to downstream tasks (Zhu et al., 2021). In this work, we wish to investigate and compare different bi-encoder backbones on downstream retrieval effectiveness in a fair manner.

2.3 Dense Retrieval Models

DPR In the Dense Passage Retriever (DPR) architecture (Karpukhin et al., 2020), a Transformer encoder $E(\cdot)$ is applied to the query or passage: the output embedding corresponding to the `[CLS]` token is used to calculate a similarity score. To facilitate retrieval efficiency, a simple dot product is used:

$$S(q,p) = E(q)^T \cdot E(p) \tag{1}$$

where $S(q,p)$ represents the similarity between a query q and a passage p.

During training, a pretrained model is used as the initial encoder state, and the encoder is optimized through the objective of a contrastive loss consisting of a query and a pair of positive and negative passages, p^+ and p^-. A common practice in training a batch of queries $\{q_i\}_1^B$ is to utilize passages of other training instances from the batch as additional *in-batch negatives* in the loss function:

$$\begin{aligned}&\mathcal{L}^{(i)}(q_i, p_i^+, p_{i,1}^-, ..., p_{i,2B-1}^-) \\ &= -\log \frac{\exp\left(S(q_i, p_i^+)\right)}{\exp\left(S(q_i, p_i^+)\right) + \sum_{j=1}^{2B-1} \exp\left(S(q_i, p_{i,j}^-)\right)}\end{aligned} \tag{2}$$

ColBERT Instead of using a single passage-level embedding, the ColBERT model (Khattab and Zaharia, 2020; Santhanam et al., 2021) preserves all output embeddings for the similarity calculation. Since each Transformer encoder is pretrained using the MLM objective (Devlin et al., 2019), the model provides fine-grained contextualized semantics for individual tokens.

Given a query token sequence $q = q_0, q_1, ...q_l$ and a passage token sequence $p = d_1, d_2, ...d_n$, ColBERT uses either dot product or L2 distance of normalized embeddings for computing the token-level similarity score $s(q_i, d_j)$. During scoring, it locates the highest-scoring token of a passage d_j for each query token q_i (i.e., the MaxSim operator), and a summation is taken over these partial scores as the overall similarity between query q and passage p:

$$S(q,p) = \sum_{i \in [E(q)]} \max_{j \in [E(p)]} s(q_i, d_j). \tag{3}$$

Similar to the DPR model, given a triple of query and a contrastive passage pair, i.e., (q, p^+, p^-), the ColBERT model optimizes a pairwise softmax cross-entropy loss.

Because ColBERT uses all passage token embeddings, it applies a linear pooling layer on top of its backbone encoder to obtain smaller fixed-size ($d = 128$ by default) embedding outputs for more efficient score computation or token-level indexing. In addition, the model prepends two different special tokens, `[Q]` or `[D]`, to distinguish the encoding of a query or a passage. In practice, the authors also demonstrate improved effectiveness via *query augmentation* by rewriting `[PAD]` query tokens to `[MASK]` tokens before query encoding.

In end-to-end retrieval, however, ColBERT typically relies on two-stage query processing for efficiency: (1) A candidate set of tokens is retrieved using highly efficient approximate nearest neighbors (ANN) search techniques (e.g., Jégou et al., 2011). (2) Then, the passages containing these tokens need to be located. (3) Finally, the candidate passages are then sent to the GPU for fast matrix multiplication to calculate token similarities for each query and passage in the candidate pool. Due to the candidate selection pipeline, this process is regarded as an approximate version of retrieval for the similarity search specified by Eq. 3.

2.4 Fusing Dense and Structure Signals

Although Peng et al. (2021) have performed structure mask pretraining for better matching formula substructures, their method is still based on additional structure embeddings generated from a different system. However, we argue that a dense retrieval model may excel at adding fuzziness and recall to math retrieval without being constrained to require a structure match in candidates. Given that previous math retrieval systems (Zhong et al., 2020, 2021) have already incorporated effective formula structure matching, we wish to combine existing well-defined structure similarity search systems with more fuzzy and higher-level semantic search capabilities from dense retrieval models.

3 Evaluation Setup

3.1 Datasets

Evaluations in this paper are conducted on two recent MIR tasks:

NTCIR-12 Wiki-Formula (Zanibbi et al., 2016a) A formula-only retrieval task made from math-related pages in Wikipedia. Both queries and documents are isolated formulas encoded using LaTeX. We consider all 20 concrete queries (no wildcards for formula variables) and index all (around 591,000) formulas as documents in this task. Judgment ratings are provided on a scale of 0 to 3. For each judged formula, the ratings are mapped to *fully relevant* (≥ 2), *partially relevant* (≥ 1), or *irrelevant* ($= 0$).

ARQMath-2 (main task) (Mansouri et al., 2021b) A CLEF answer retrieval task for math-related questions. The collection includes roughly 1 million questions, containing 28 million formulas extracted from the MSE (Math StackExchange) website.[2] There are 100 question posts sampled from MSE, where 71 of these questions are sufficiently evaluated (an average of 450 answers per topic are assessed by human experts). The official evaluation measurements in ARQMath are prime-versions of NDCG, MAP, and Precision at 10. They differ from the original metrics in that unjudged documents from the ranked lists are removed before evaluation. Relevance levels include *High* ($= 3$), *Medium* ($= 2$), *Low* ($= 1$), and *Irrelevant* ($= 0$). High and Medium relevance are collapsed for binary evaluation metrics.

We use the official evaluation metrics and protocols for both tasks. Each run contains a ranked list of 1000 documents per query.

3.2 Pretraining Configurations

We consider three types of pretrained Transformer backbones for downstream math retrieval tasks.

BERT (Devlin et al., 2019) A Transformer encoder pretrained using MLM and NSP objectives on a large corpus comprising the Toronto Book Corpus and English Wikipedia.

SciBERT (Beltagy et al., 2019) A further pretrained Transformer encoder built on the BERT base model using 1.14M scientific papers with additional vocabularies for scientific content.

[2] https://math.stackexchange.com

Example: Inequality between norm 1, norm 2 and norm ∞ of matrices: $\|A\|_2 \leq \sqrt{\|A\|_1 \|A\|_\infty}$

Output: Inequality between norm 1, norm 2 and norm <infty> of matrices: <Vert> <A> <Vert> <subscript> <2> <le> <root> <{> <Vert> <A> <Vert> <subscript> <1> <Vert> <A> <Vert> <subscript> <infty> <}>

Figure 1: Example of pre-tokenized math content containing LaTeX markup. The output has all meaningful or syntactic LaTeX tokens preserved.

Our further-pretrained BERTs We further pretrain the BERT base model on 1.69M math-related documents composed of texts and math formulas using the MLM and NSP objectives proposed by Devlin et al. (2019). Specifically, we crawl MSE and the Art of Problem Solving[3] websites. Out of 9M sentences from these documents, we extracted 2.2M sentence pairs for training. All LaTeX markup are pre-tokenized using the PyA0 toolkit (Zhong and Lin, 2021), which unifies semantically identical tokens (e.g., `\frac` and `\dfrac`), adding 539 new tokens into the vocabulary space. An example of the pre-tokenizing process can be found in Figure 1. We treat these LaTeX tokens as regular text during training. Our own backbone is trained on eight A100 GPUs with a batch size of 240 for 3 and 7 epochs.

3.3 Fine-Tuning Configurations

On top of different backbones, we fine-tune our bi-encoder models using the ARQMath collection as training data (we use Q&A posts prior to the year 2018). Given a query q, we sample a positive passage p^+ from accepted answers, duplicate questions, or any answer posts to the query receiving more than 7 upvotes. A random answer passage related to the same tags is treated as a hard negative sample p^-. We obtain 607K (q, p^+, p^-) triplets for training dense retrieval models.

We use the AdamW optimizer (Loshchilov and Hutter, 2017) in all our experiments, with a weight decay of 0.01, and a learning rate of 1×10^{-6} for ColBERT and 3×10^{-6} for DPR. Following Reusch et al. (2021b), we set the maximum number of input tokens to 512.

DPR models trained for 1 epoch To validate the effectiveness of our pretrained backbones, we design a comparative experiment where DPR models on different backbones are trained for the same

[3] https://artofproblemsolving.com/community

number of steps (∼550K iterations, approximately one epoch) with a batch size of 15. The goal of these conditions is to quickly compare the effectiveness of different backbones.

Fully-trained models To maximize effectiveness, we fine-tune our DPR and ColBERT models on our backbone that has been further pretrained for 7 epochs. We fine-tune DPR for 10 epochs with a batch size of 36, and ColBERT with a batch size of 30 for 3 epochs, both on A6000 GPUs.

3.4 Structure Search Fusion

The best way to add structure similarity awareness to dense retrieval models remains an important and open problem. In this work, we make the first attempt to simply merge dense retrieval results with results generated from a structure search system, Approach0 (Zhong and Zanibbi, 2019; Zhong et al., 2020). Approach0 takes a top-down approach to evaluate formula similarities, and thus it is very complementary to more fuzzy semantic retrieval.

We evaluate search fusion against the NTCIR-12 and ARQMath-2 tasks. Based on structure search results, which are generated by Approach0 and tuned on a different dataset (i.e., ARQMath-1), we perform one of two alternatives: (1) rerank the baseline using inference scores from DPR or ColBERT; (2) linearly combine scores from the baseline and from DPR or ColBERT. In the second case, we perform 5-fold cross validation to tune a weight $\alpha \in \{0.1, ..., 0.9\}$. The final fusion score S_f is interpolated by a convex combination:

$$S_f = \alpha \cdot S_d + (1 - \alpha) \cdot S_a,$$

where S_d, S_a are the scores from the dense retrievers and the structure search, respectively. Original scores are rescaled using min-max normalization, and when a document is missing from the other source during fusion, we set its score to zero.

3.5 Baselines

For the NTCIR-12 dataset, we compare our scores to the only Transformer retriever reported on this dataset, i.e., MathBERT (Peng et al., 2021), a BERT model with specialized structure-aware further pretraining. However, their results should be regarded as an ensemble run because they are generated from reranking a highly effective run produced by Tangent-CFT (Mansouri et al., 2019).

For the ARQMath-2 dataset, we select other bi-encoder Transformer runs submitted or reported on the ARQMath-2 main task (Mansouri et al., 2021b). This includes ColBERT runs based on different backbones from the TU_DBS team (Reusch et al., 2021b), using the weights of the original BERT-base, SciBERT (Beltagy et al., 2019), and a Col-ARQBERT pretrained from scratch on the ARQ-Math corpus. Furthermore, two additional effective bi-encoder models—CompuBERT (Novotnỳ et al., 2021) from MIRMU and FormulaEmb (Dadure et al., 2021)—are also compared. The former uses averaged token embeddings of SentenceBERT fine-tuned by minimizing the cosine distance of questions to their accepted or high-ranking answers, while the latter uses pretrained Transformer embeddings directly for similarity computations.

In our fusion results, we compare top-effective existing systems. For the NTCIR-12 dataset, we include: MCAT (Kristianto et al., 2016) – an expensive MIR system that takes on average over 25 seconds per query to run; the Tangent-S system (Davila and Zanibbi, 2017) using low-granularity structure node pairs; and its successor Tangent-CFT (Mansouri et al., 2019) based on FastText embeddings of local structures from the SLT representation; a GNN model for formula retrieval (Song and Chen, 2021); and finally, MathBERT (Peng et al., 2021). However, the two most effective ensemble runs, TanAPP (Mansouri et al., 2019) and MathAPP (Peng et al., 2021), are excluded because their linear fusion weights are tuned directly on the complete NTCIR-12 dataset.

For the ARQMath-2 dataset, we include the most effective systems for comparison: the MathDowsers primary system (Fraser et al., 2018; Ng et al., 2020, 2021; Ng, 2021) and the up-to-date Approach0 system. Additionally, two cross-encoder dense retrievers are included: the TU_DBS primary retriever based on ALBERT (Reusch et al., 2021a) and QASim (Mansouri et al., 2021a), which combines two Transformers, for question-question and question-answer similarity assessment. We also consider ensemble systems including the most effective run (WIBC) from the MIRMU team (Novotnỳ et al., 2021) and the official tf–idf and tf–idf+Tangent-S baselines provided in the ARQMath-2 main task. The tf–idf+Tangent-S baseline is an unweighted average fusion between the results produced by the Terrier system (Ounis et al., 2005) and a structure-search system, Tangent-S (Davila and Zanibbi, 2017). In the Terrier pass, LaTeX strings are directly used for retrieval.

Table 1: Effectiveness comparisons of bi-encoder Transformers. Rows (1)–(4) show DPR models fine-tuned for one epoch, starting from different pretrained backbones. Our fully-trained passage-level and token-level models (DPR and ColBERT, respectively), rows (11) and (12), are compared with existing Transformer models in rows (5)–(10). */** denotes that the compared row performs weaker than the bottom row in each block, i.e., the 1-epoch fine-tuned and 7-epoch further-pretrained BERT in row (4) or our fully-pretrained ColBERT model in row (12), at $p < 0.05/0.01$ level using the two-tailed pairwise t-test. Underlined scores are not involved in any test of significance due to unavailable run files.

Runs	NTCIR-12 Wiki-Formula			CLEF ARQMath-2				
	Full BPref	Part. BPref	Judged ‰	NDCG'	MAP'	P'@10	BPref	Judged ‰
DPR models fine-tuned for 1 epoch								
(1) BERT (2019)	0.505	0.393	21.9	0.174*	0.051	0.116	0.073	45.3
(2) SciBERT (2019)	0.512	0.363	21.1	0.176*	0.056	**0.134**	0.073	42.3
(3) further-pretrained BERT (3 epochs)	0.486	0.392	21.8	0.195	0.058	0.126	0.073	45.0
(4) further-pretrained BERT (7 epochs)	**0.522**	**0.439**	23.2	**0.200**	**0.060**	0.130	**0.081**	46.8
Other bi-encoder Transformers								
(5) MathBERT † (2021)	**0.614**	**0.736**	-	-	-	-	-	-
(6) TU_DBS ColSciBERT (2021b)	-	-	-	0.028**	0.004**	0.009**	0.009**	17.5
(7) TU_DBS ColBERT (2021b)	-	-	-	0.183	0.053	0.110	-	-
(8) TU_DBS ColARQBERT (2021b)	-	-	-	0.225	0.073	0.131	-	-
(9) MIRMU CompuBERT (2021)	-	-	-	0.262**	0.083**	0.135**	0.087**	69.2
(10) FormulaEmb (2021)	-	-	-	0.161	0.059	0.197	-	-
Our fully-trained models								
(11) Our DPR	0.516	0.427	23.4	0.270**	0.087**	0.152**	0.097**	66.3
(12) Our ColBERT	0.545	0.483	25.1	**0.329**	**0.128**	**0.213**	**0.136**	69.5

†: Ensemble system.

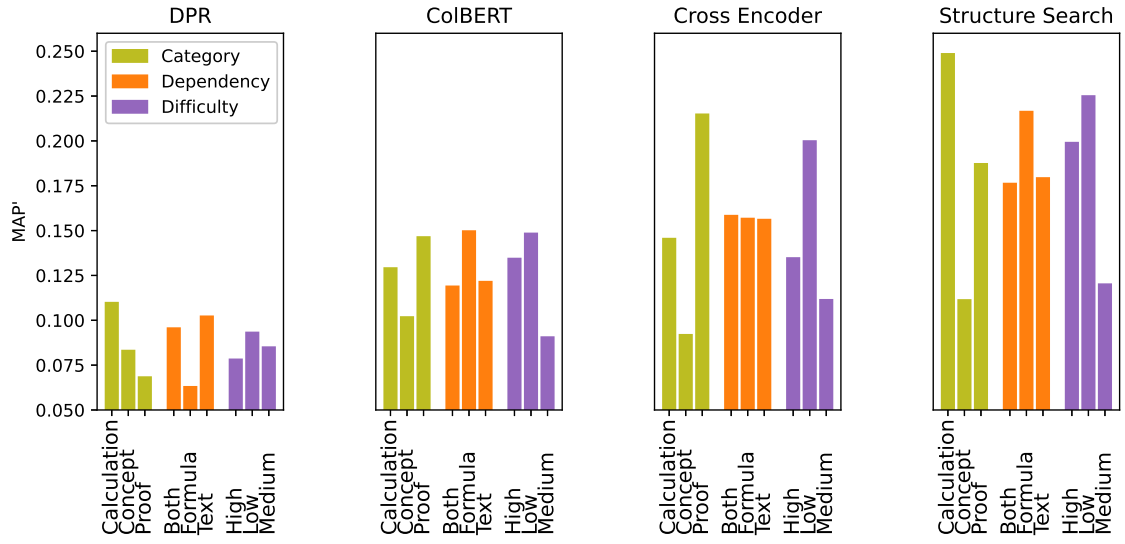

Figure 2: The MAP′ scores produced by our fully-trained DPR, ColBERT, a cross encoder represented by the TU_DBS primary run, and the structure matching retriever Approach0, all evaluated on the ARQMath-2 dataset. Results are divided by different topic categories (Calculation, Concept, or Proof), semantic dependencies (Text, Formula, or Both), and different difficulty levels (Low, Medium, and High). Note that the y-axes of all plots have the same scale.

4 Results

4.1 Overall Comparisons

Evaluation results for Transformer-based dense models are shown in Table 1. Across both formula-only retrieval (NTCIR-12) and math-aware full-text retrieval (ARQMath-2), our pretrained backbones can generally boost downstream DPR retrieval effectiveness compared to DPR models based on vanilla BERT, row (1), or SciBERT, row (2). This is presumably because we have further pretrained on more domain-specific data (unlike SciBERT, which also includes scientific text like biomedical articles) with a much larger batch size, i.e., 240 compared to SciBERT's 32 batch size. According to rows (2)–(4) in Table 1, with pretraining only for 3 epochs, our model reaches a similar level of effectiveness as SciBERT; and more pretraining results in better downstream effectiveness.

Our fully-trained ColBERT model, row (12), achieves the best scores among other Transformer models. Compared to other ColBERT variants from rows (6)–(8) submitted by the TU_DBS team, we also achieve higher scores. Our DPR model, row (11), is generally more effective than previous bi-encoder systems, so it can be considered a cost-effective alternative to ColBERT, since the latter requires a much bigger index. For ARQMath-2, our DPR model requires a 5.3G index (at full precision), while our ColBERT model requires a 77G index (at half precision). However, our dense models are not on par with the MathBERT run on the NTCIR-12 dataset, row (5). This is because MathBERT reranks a highly effective run generated by Tangent-CFT; the latter is directly tuned on complete NTCIR-12 data.

4.2 Comparisons on Different Topics

To further investigate the strengths and weaknesses of different architectures for math-aware retrieval, we break down results by different types of topics, e.g., *computation*, *concept*, or *proof*. Topic categories are labeled by the ARQMath-2 task organizers (Mansouri et al., 2021b).

As shown in Figure 2, the DPR model, compared to itself, is good at text retrieval but bad at formula retrieval, while ColBERT is the opposite. The cross encoder (from left to right, the 3rd plot), on the other hand, handles all types of dependencies equally well, and it shows sufficient understanding for easy math question retrieval and proof-related topics. On the other hand, structure search (the rightmost plot) excels at the calculation category and formula-dependent retrieval, with most categories performing even better than the cross encoder. This demonstrates that matching formula structure is still crucial for effective math-aware search, especially for formula-heavy content such as the calculation category.

Table 2: Comparison of effectiveness on the NTCIR-12 Wiki-Formula dataset. We combine a structure search system (Approach0) with our fully-trained DPR and ColBERT models. End-to-end fusion weights are tuned via cross-validation. */** denotes that the compared row performs weaker than the bottom row (i.e., Approach0 + DPR in end-to-end fusion) at $p < 0.05/0.01$ level using the two-tailed pairwise t-test. Underlined scores are not involved in any test of significance.

Runs	Part. BPref	Full BPref
Previous systems		
(1) MCAT (2016)	0.57	0.57
(2) Tangent-S (2017)	0.59	0.64
(3) Tangent-CFT (2019)	0.71	0.60
(4) GNN (2021)	0.54	0.63
(5) MathBERT (2021)	**0.74**	0.61
Structure search baseline		
(6) Approach0 (2019; 2020)	0.54*	0.63
Our reranking		
(7) Approach0 + DPR	0.48**	0.53*
(8) Approach0 + ColBERT	0.52*	0.56*
Our end-to-end fusion		
(9) Approach0 + ColBERT	0.63	0.65*
(10) Approach0 + DPR	0.62	**0.70**

4.3 Fusion Results

As shown in Figure 2, even a cross encoder (without special structure pretraining) can fall short for formula retrieval compared to the structure search approach. Nevertheless, we want to learn if dense retrieval can be combined with structure search to further advance structure search effectiveness.

Our fusion results are summarized in Table 2 and Table 3. On the NTCIR-12 Wiki-Formula dataset (Table 2), comparing rows (1)–(5), our linear fusion runs in rows (9)–(10) outperform others in fully relevant BPref scores. This shows we can generate a good ranking for highly relevant formulas when linearly combining end-to-end dense retrieval and structure search. The formula-only reranking in rows (7)–(8) is not beneficial, but on the other hand, end-to-end fusion in rows (9)–(10) is helpful because dense retrieval can improve recall when structure matching is too strict (more discussion below).

On the ARQMath dataset, comparing rows (7)–

Table 3: Results from the most effective runs of previous systems in ARQMath-2 compared to our method for combining a structure search model (Approach0) with our fully-trained DPR and ColBERT models. End-to-end fusion weights are tuned via cross-validation. */** denotes that the compared row performs weaker than the bottom row (i.e., Approach0 + ColBERT in end-to-end fusion) at $p < 0.05/0.01$ level using the two-tailed pairwise t-test.

Runs	NDCG'	MAP'	P'@10	BPref
Previous systems				
(1) MathDowsers Primary (2018; 2020; 2021)	0.434	0.169*	0.211*	0.145**
(2) TU_DBS Primary (2021b)	0.377**	0.158**	0.227*	0.158*
(3) DPRL QASim (2021a)	0.388*	0.146**	0.193*	0.135**
(4) MIRMU WIBC (2021)	0.332**	0.087**	0.106**	0.069**
(5) tf–idf (Terrier) (2021b)	0.185**	0.046**	0.063**	0.046**
(6) tf–idf+Tangent-S (2021b)	0.201**	0.045**	0.086**	0.048**
Structure search baseline				
(7) Approach0 (2019; 2020)	0.381**	0.189*	0.234*	0.180*
Our reranking				
(8) Approach0 + DPR	0.372**	0.169**	0.235*	0.162**
(9) Approach0 + ColBERT	0.383**	0.182**	**0.276**	0.181*
Our end-to-end fusion				
(10) Approach0 + DPR	0.429*	0.203	0.258	0.189
(11) Approach0 + ColBERT	**0.447**	**0.215**	0.252	**0.202**

(11) in Table 3 and rows (11) and (12) in Table 1, we see that although the structure search baseline produced by Approach0 alone is generally more effective than dense retrieval models, both DPR and ColBERT can still boost the baseline results. With the assistance of structure search, we are also able to outperform cross-encoder models shown in row (2) and row (3) in Table 3. These cross encoders require costly inference over every candidate pair. In fact, due to the impractical inference times of cross encoders for the ARQMath dataset, the TU_DBS team had to limit their candidate pool prior to indexing. Similarly, the DPRL QASim run adopts a smaller TinyBERT model to practically compute similarities for all candidate pairs in a limited set.

Interestingly, across two datasets, reranking is not helpful in general, other than a precision boost in rows (8)–(9), Table 3. This is because the dense rerankers are prone to false positives at formula retrieval compared to structure search, and this is especially the case when a dense retriever is used to rerank a highly effective formula retriever baseline. We report extra experiments to support this argument in Section 5. This indicates that the dense retrievers are only complementary to the structure search approach in a way that helps recall rather than reranking.

5 Discussion

Given that linear fusion is able to produce such good results, a natural question to ask is whether other fusion methods can lead to even better results.

Table 4: Other fusion methods evaluated using the most competitive Approach0 + ColBERT model combination on the ARQMath-2 dataset. */** denotes that the compared row performs significantly weaker than the linear fusion at $p < 0.05/0.01$ level using the two-tailed pairwise t-test. ISR and RRF stand for *Inverse Square Rank* and *Reciprocal Rank Fusion*, respectively.

Fusion	NDCG'	MAP'	P'@10	BPref
Borda Count	0.443	0.213	0.280	0.197
CombSUM	0.411**	0.213	0.296	0.216
ISR	0.433**	0.203	0.263	0.191
log-ISR	0.432**	0.202	0.263	0.189
RRF ($k = 60$)	0.449	0.221	0.284	0.200
Linear	0.449	0.217	0.279	0.204

Therefore, we compare popular fusion methods[4] on the ARQMath-2 datasets and our results are summarized in Table 4. In all experiments, we directly choose the best fusion parameters tuned on the ARQMath-2 dataset to obtain an optimistic bound for each method. Table 4 shows that linear interpolation is sufficient to generate "good enough" results that are not significantly worse (sometimes better) than other popular fusion methods.

We further investigate the reasons why structure search and dense retrieval are highly complementary but not so in the reranking case. After probing a number of queries where fusion runs achieve much better results, we find that the structure constraint imposed on candidates by Approach0 can

[4]We use the `polyfuse` tool: https://github.com/rmit-ir/polyfuse

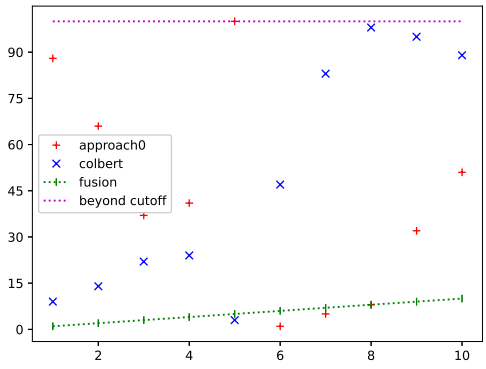

Figure 3: Retrieved document ranks of the Approach0 + ColBERT fusion run (topic A.219, cut off at 10) and their positions in the original runs. The x-axis corresponds to the ranks of retrieved documents in the fusion run, the y-axis corresponds to the ranks of retrieved documents in the original runs, and each point represents one document.

fail completely when relevant documents do not share any common substructure in math formulas, especially if the query is formula-centered, while dense retrieval has the capacity to find these relevant documents by matching contextual semantics. On the other hand, structure search is helpful to dense retrieval in cases where an obviously relevant document is found by matching a candidate formula perfectly.

We illustrate the above statement using the topic where the precision metric has the largest increase after fusion. Specifically, topic A.287, which gains the most when the fusion run is compared against the Approach0 baseline (P@10 changes from 0 to 0.8), fails for Approach0 because structure match does not occur in relevant documents and the query is formula-centered. On the other hand, when compared to the ColBERT run, the fusion run in topic A.219 shows the most gain in precision (P@10 increases from 0.1 to 0.5). Figure 3 shows the ranks of top-10 fusion results and their original positions in topic A.219. By further inspecting in detail, we discover that structure search prevents false positives in dense retrieval. Specifically, the top-3 dense retrieval hits in topic A.219 contain binomial coefficient notation, e.g., $\sum_{k=0}^{r} \binom{m}{k}\binom{n}{r-k} = \binom{m+n}{r}$, which looks similar to the query $\binom{n+r+1}{r} = \sum_{k=0}^{r} \binom{n+k}{k}$ but is not equivalent mathematically. They are ruled out or get lowered in rank in the top-10 final results because their counterparts in structure search are missing (e.g., rank 1st and 2nd results from ColBERT run) or out of sight (e.g., rank 3rd result from ColBERT run), and those ColBERT hits paired with a structure hit in the Approach0 pass stand out.

6 Conclusions

Rapid progress in dense retrieval models using deep neural networks has greatly influenced many IR tasks. In this paper, we provide a thorough evaluation of both token-level and passage-level bi-encoder models in the math information retrieval domain. Our DPR and ColBERT models adapted to this domain in both pretraining and fine-tuning are made publicly accessible to provide stepping stones for future research. Our study also highlights the importance of combining structure search with dense retrieval models for better math-aware search. We show that bi-encoder dense retrieval models alone can be less effective than cross encoders, but when combined with strong structure search methods, they can further improve state-of-the-art effectiveness. With the huge modeling capacity of dense retrieval models, we believe it is worth exploring other directions for improvements so that we can unleash the potential of deep models in this domain, for example, to better identify similarities in mathematically transformed expressions with different structures.

7 Limitations

What our evaluations suggest in this work is to build end-to-end retrievers by combining strict structure search and dense retrieval for a highly effective math-aware search. We are aware of two limitations: First, an ensemble of two different end-to-end retrieval systems imposes engineering challenges; the benefit of supporting math-aware search may be offset by the overhead of adding multiple software stacks. Second, it is unclear how to highlight matching in the case of DPR; and in the case of ColBERT, it demands larger storage (see Section 4.1) and requires intensive GPU resources to perform the MaxSim operation over the embeddings of all candidate tokens (see Section 2.3).

Acknowledgments

This research was supported in part by the Canada First Research Excellence Fund and the Natural Sciences and Engineering Research Council (NSERC) of Canada. Computational resources were provided in part by Compute Ontario and Compute Canada.

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. *arXiv:1903.10676*.

Pankaj Dadure, Partha Pakray, and Sivaji Bandyopadhyay. 2021. BERT-based embedding model for formula retrieval. In *CLEF*.

Kenny Davila and Richard Zanibbi. 2017. Layout and semantics: Combining representations for mathematical formula search. In *SIGIR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2021. SPLADE v2: Sparse lexical and expansion model for information retrieval. *arXiv:2109.10086*.

Dallas Fraser, Andrew Kane, and Frank Tompa. 2018. Choosing math features for BM25 ranking with Tangent-L. In *DocEng*.

Liangcai Gao, Zhuoren Jiang, Yue Yin, Ke Yuan, Zuoyu Yan, and Zhi Tang. 2017. Preliminary exploration of formula embedding for mathematical information retrieval: Can mathematical formulae be embedded like a natural language? *arXiv:1707.05154*.

Luyu Gao and Jamie Callan. 2021. Condenser: A pre-training architecture for dense retrieval. *arXiv:2104.08253*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv:2004.10964*.

Yoshinori Hijikata, Hideki Hashimoto, and Shogo Nishida. 2009. Search mathematical formulas by mathematical formulas. In *SHI (Symposium on Human Interface)*.

Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. 2021. Efficiently teaching an effective dense retriever with balanced topic aware sampling. In *SIGIR*.

Hervé Jégou, Romain Tavenard, Matthijs Douze, and Laurent Amsaleg. 2011. Searching in one billion vectors: Re-rank with source coding. In *IEEE ICASSP*.

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:2004.04906*.

Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *SIGIR*.

Michael Kohlhase, Bogdan A Matican, and Corneliu-Claudiu Prodescu. 2012. MathWebSearch 0.5: Scaling an open formula search engine. In *CICM*.

Giovanni Yoko Kristianto, Goran Topic, and Akiko Aizawa. 2016. MCAT math retrieval system for NTCIR-12 MathIR task. In *NTCIR*.

Xiaoyan Lin, Liangcai Gao, Xuan Hu, Zhi Tang, Yingnan Xiao, and Xiaozhong Liu. 2014. A mathematics retrieval system for formulae in layout presentations. In *SIGIR*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv:1711.05101*.

Behrooz Mansouri, Douglas W Oard, and Richard Zanibbi. 2020. DPRL systems in the CLEF 2020 ARQMath lab. In *CLEF*.

Behrooz Mansouri, Douglas W. Oard, and Richard Zanibbi. 2021a. DPRL systems in the CLEF 2021 ARQMath lab: Sentence-BERT for answer retrieval, learning-to-rank for formula retrieval. In *CLEF*.

Behrooz Mansouri, Shaurya Rohatgi, Douglas W. Oard, Jian Wu, C Lee Giles, and Richard Zanibbi. 2019. Tangent-CFT: An embedding model for mathematical formulas. In *SIGIR*.

Behrooz Mansouri, Richard Zanibbi, Douglas W. Oard, and Anurag Agarwal. 2021b. Overview of ARQMath-2 (2021): Second CLEF lab on answer retrieval for questions on math (working notes version). In *CLEF*.

Bruce R. Miller and Abdou Youssef. 2003. Technical aspects of the digital library of mathematical functions. In *AMAI*.

Yin Ki Ng. 2021. Dowsing for math answers: Exploring MathCQA with a math-aware search engine. Master's thesis, University of Waterloo.

Yin Ki Ng, Dallas Fraser, Besat Kassaie, and Frank Tompa. 2021. Dowsing for answers to math questions: Ongoing viability of traditional MathIR. In *CLEF*.

Yin Ki Ng, Dallas J. Fraser, Besat Kassaie, George Labahn, Mirette S Marzouk, Frank Tompa, and Kevin Wang. 2020. Dowsing for math answers with Tangent-L. In *CLEF*.

Vít Novotnỳ, Michal Štefánik, Dávid Lupták, Martin Geletka, Petr Zelina, and Petr Sojka. 2021. Ensembling ten math information retrieval systems. In *CLEF*.

Iadh Ounis, Gianni Amati, Vassilis Plachouras, Ben He, Craig Macdonald, and Douglas Johnson. 2005. Terrier information retrieval platform. In *ECIR*.

Shuai Peng, Ke Yuan, Liangcai Gao, and Zhi Tang. 2021. MathBERT: A pre-trained model for mathematical formula understanding. *arXiv:2105.00377*.

Anja Reusch, Maik Thiele, and Wolfgang Lehner. 2021a. An ALBERT-based similarity measure for mathematical answer retrieval. In *SIGIR*.

Anja Reusch, Maik Thiele, and Wolfgang Lehner. 2021b. TU_DBS in the ARQMath lab 2021, CLEF. In *CLEF*.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2021. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. *arXiv:2112.01488*.

Thomas Schellenberg, Bo Yuan, and Richard Zanibbi. 2012. Layout-based substitution tree indexing and retrieval for mathematical expressions. In *DRR*.

Petr Sojka and Martin Líška. 2011. The art of mathematics retrieval. In *DocEng*.

Yujin Song and Xiaoyu Chen. 2021. Searching for mathematical formulas based on graph representation learning. In *CICM*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Keisuke Yokoi and Akiko Aizawa. 2009. An approach to similarity search for mathematical expressions using MathML. In *DML (Digital Mathematics Library)*.

Abdou Youssef. 2005. Search of mathematical contents: Issues and methods. In *IASSE*.

Richard Zanibbi, Akiko Aizawa, Michael Kohlhase, Iadh Ounis, Goran Topic, and Kenny Davila. 2016a. NTCIR-12 MathIR task overview. In *NTCIR*.

Richard Zanibbi and Dorothea Blostein. 2012. Recognition and retrieval of mathematical expressions. In *IJDAR*.

Richard Zanibbi, Kenny Davila, Andrew Kane, and Frank Tompa. 2015. The Tangent search engine: Improved similarity metrics and scalability for math formula search. *arXiv:1507.06235*.

Richard Zanibbi, Kenny Davila, Andrew Kane, and Frank Tompa. 2016b. Multi-stage math formula search: Using appearance-based similarity metrics at scale. In *SIGIR*.

Wei Zhong. 2015. *A novel similarity-search method for mathematical content in LaTeX markup and its implementation*. University of Delaware.

Wei Zhong and Hui Fang. 2016. OPMES: A similarity search engine for mathematical content. In *ECIR*.

Wei Zhong and Jimmy Lin. 2021. PyA0: A Python toolkit for accessible math-aware search. In *SIGIR*.

Wei Zhong, Shaurya Rohatgi, Jian Wu, Lee Giles, and Richard Zanibbi. 2020. Accelerating substructure similarity search for formula retrieval. In *ECIR*.

Wei Zhong and Richard Zanibbi. 2019. Structural similarity search for formulas using leaf-root paths in operator subtrees. In *ECIR*.

Wei Zhong, Xinyu Zhang, Ji Xin, Jimmy Lin, and Richard Zanibbi. 2021. Approach Zero and Anserini at the CLEF-2021 ARQMath track: Applying substructure search and BM25 on operator tree path tokens. In *CLEF*.

Qi Zhu, Yuxian Gu, Lingxiao Luo, Bing Li, Cheng Li, Wei Peng, Minlie Huang, and Xiaoyan Zhu. 2021. When does further pre-training MLM help? An empirical study on task-oriented dialog pre-training. In *EMNLP Insights Workshop*.