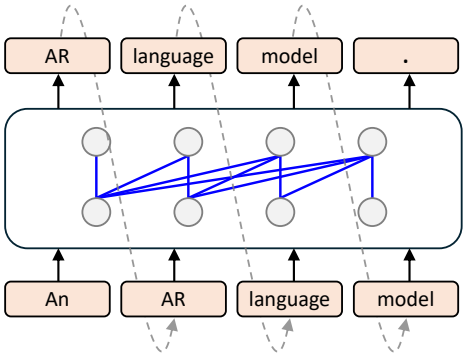
Diffusion vs. Autoregressive Language Models: A Text Embedding Perspective

Siyue Zhang[N A S] **Yilun Zhao**[Y] **Liyuan Geng**[S] **Arman Cohan**[Y]
Anh Tuan Luu[N] **Chen Zhao**[S C]

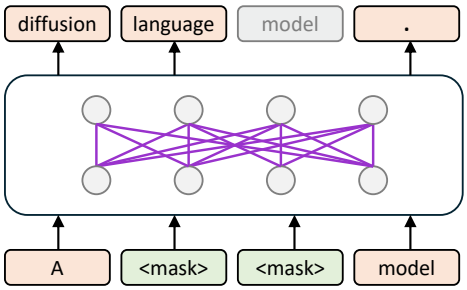
[N]Nanyang Technological University [Y]Yale University [S]NYU Shanghai
[A]Alibaba-NTU Singapore Joint Research Institute
[C]Center for Data Science, New York University

Abstract

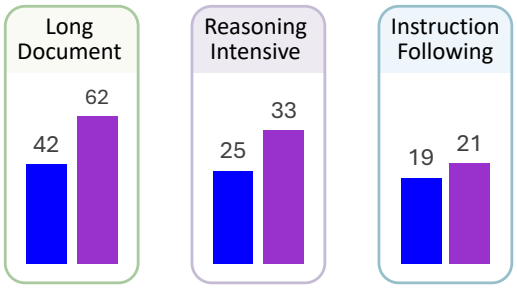
Large language model (LLM)-based embedding models, benefiting from large scale pre-training and post-training, have begun to surpass BERT and T5-based models on general-purpose text embedding tasks such as document retrieval. However, a fundamental limitation of LLM embeddings lies in the unidirectional attention used during autoregressive pre-training, which misaligns with the bidirectional nature of text embedding tasks. To this end, we propose adopting diffusion language models for text embeddings, motivated by their inherent bidirectional architecture and recent success in matching or surpassing LLMs especially on reasoning tasks. We present the first systematic study of the diffusion language embedding model, which outperforms the LLM-based embedding model by 20% on long-document retrieval, 8% on reasoning-intensive retrieval, 2% on instruction-following retrieval, and achieve competitive performance on traditional text embedding benchmarks. Our analysis verifies that bidirectional attention is crucial for encoding global context in long and complex text.[1]



(a) Autoregressive Modelling



(b) Diffusion Modelling in DREAM



(c) Llama3 + LLM2Vec (blue) vs. DREAM (purple)

Figure 1: **(a)** Unidirectional attention in Autoregressive LM. **(b)** Bidirectional attention in Diffusion LM, *i.e.,* DREAM (Ye et al., 2025). **(c)** Retrieval performance comparison between the diffusion embedding model and the LLM embedding model enhanced with LLM2Vec adaptation (BehnamGhader et al., 2024).

1 Introduction

Learning text embeddings is a fundamental NLP problem that supports a wide range of downstream applications such as retrieval-augmented generation (RAG). Traditionally, text embedding models (Karpukhin et al., 2020; Izacard et al., 2021; Thakur et al., 2021) have been trained using contrastive learning on top of pre-trained bidirectional language models such as BERT (Devlin et al., 2019) and T5 (Raffel et al., 2020). More recently, several studies (Ma et al., 2024; Wang et al., 2024b; Li et al., 2023; Lee et al., 2024; Du et al., 2025; Muennighoff et al., 2025) have adapted decoder-only large language models (LLMs), achieving notable improvements on embedding benchmarks such as MTEB (Muennighoff et al., 2023).

Despite their strong empirical performance,

[1]Our code and data are available at https://github.com/siyue-zhang/diffusion_embedder.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 4273–4303
November 4-9, 2025 ©2025 Association for Computational Linguistics

LLM-based embedding models struggle to capture global context due to the causal attention structure used during large-scale training (Springer et al., 2025). As illustrated in Figure 1a, contextualized token embeddings—the final-layer hidden states at each position—are computed without access to future tokens. This leads to a mismatch between the unidirectional nature of LLM training paradigm and the bidirectional context understanding required for text embedding tasks. Echo Embeddings (Springer et al., 2025) attempts to address this issue by duplicating the input and extracting embeddings from the second copy, though this adds inference overhead, particularly for long texts. LLM2Vec (BehnamGhader et al., 2024) adapts LLMs with bidirectional attention through continue training, but is limited with scale.

We present DIFFEMBED, a novel approach that leverages diffusion language models (LMs) for text embedding. Recent diffusion LMs design discrete diffusion processes using forward masking and reverse unmasking within a bidirectional attention architecture. Trained at scale similarly to autoregressive LLMs, these models have demonstrated competitive performance across a variety of tasks (Sahoo et al., 2024; Nie et al., 2025; Ye et al., 2025). *Our hypothesis is that diffusion embeddings, when trained at scale with a bidirectional attention architecture, are better suited to capturing global context and thus achieve superior performance on long and complex documents.*

To test this hypothesis, we evaluate DIFFEMBED and LLM embedding models[2] over a wide range of tasks, including long-document retrieval, reasoning-intensive (logical) retrieval, instruction-following retrieval, and general text embedding tasks. As no existing training dataset is effective for the reasoning-intensive task, we develop a new training set REASONAUG using LLMs, which contains 10,896 pairs of logically related positives and unrelated hard negatives. These documents cover a range of domains, from mathematics and physics theorems to code.

Our experimental results show that DIFFEMBED, based on the state-of-the-art diffusion LM DREAM-7B (Ye et al., 2025), outperforms the Llama3-8B based model by 20% on the long-document retrieval benchmark LONGEMBED (Zhu et al., 2024), 8% on the reasoning-intensive retrieval benchmark BRIGHT (Su et al., 2025), and 2% on instruction-following retrieval benchmark FOLLOWIR (Weller et al., 2024). Notably, with REASONAUG, DIFFEMBED surpasses the state-of-the-art performance on TheoremQA tasks in BRIGHT by 16.4%. Our ablation study shows that attention in both directions (causal and reverse) is crucial for encoding long and complex documents. The reverse attention has a greater impact in DIFFEMBED than LLM-based models.

To summarize, our contributions include:

- We propose to leverage the diffusion LMs for text embedding, which are trained at scale with a bidirectional architecture. This approach is intuitively motivated and avoids adaptations typically required for LLM embeddings (BehnamGhader et al., 2024).
- To the best of our knowledge, we are the first to systematically evaluate diffusion embeddings, which demonstrate superior long-document encoding and logical reasoning capabilities, as well as competitive performance on general text embedding tasks.
- We present REASONAUG, a new dataset which is constructed solely using LLMs and significantly improves the retrieval performance for logical documents, which requires intensive reasoning.

2 Background

Text Embedding Tasks. The goal of text embedding models is to map a sequence of tokens $x = x_1, ..., x_n$ to a vector $\phi(x) \in \mathbb{R}^d$ that preserves semantic similarity within a low-dimensional space. Embeddings are used in a wide range of downstream applications such as document retrieval, clustering, classification, among others (Muennighoff et al., 2023). Specifically, in document retrieval, for each query embedding, document embeddings are ranked by semantic similarity; in clustering, embeddings form semantically coherent groups; and in classification, embeddings are directly mapped to class labels.

Text Embedding Models. Text embedding models are typically adapted from pre-trained language model to leverage their contextual understanding of text. Embeddings are obtained from language model's final layer, where each input token x_j at position j is associated with a contextualized representation $\phi_j(x)$. To obtain a fixed-size sequence

[2]For brevity, we refer to the autoregressive and diffusion LM-based embedding models as the LLM and diffusion embedding models, respectively.

embedding, token representations are aggregated—commonly via mean pooling or by selecting the final token's representation.

These embedding models are further fine-tuned through contrastive learning to produce more effective representations (Gao et al., 2021; Wang et al., 2024a; Chen et al., 2024; Jina, 2024). Given a query q, a positive text p^+, and a set of negative text $p_1^-, \ldots, p_m^-$, the model is trained to increase the similarity between the query and the positive text while decreasing the similarity with negative text. This is achieved by optimizing the following objective (Karpukhin et al., 2020):

$$\mathcal{L}(q, p^+, p_1^-, \ldots, p_m^-) = \frac{e^{f(q,p^+)}}{e^{f(q,p^+)} + \sum_{j=1}^{m} e^{f(q,p_j^-)}}, \tag{1}$$

where $f(q, p)$ denotes a similarity function, e.g., dot product.

For years, bidirectional LMs such as BERT (Devlin et al., 2019) and T5 (Raffel et al., 2020) have dominated as backbone for text embedding models. ModernBERT (Warner et al., 2024) scales up BERT training with a longer sequence length. Recent works have made plentiful advances to adapt decoder-only LLMs for text embeddings such as Repllama (Ma et al., 2024), E5-Mistral (Wang et al., 2024b), and NV-Embed (Lee et al., 2024). To overcome the limitation of unidirectional attention in LLMs, LLM2Vec (BehnamGhader et al., 2024) introduce the self-supervised training to adapt LLMs to use bidirectional attention for embedding tasks.

Diffusion Language Models. Diffusion models (Sohl-Dickstein et al., 2015; Song et al., 2021) excel in generative tasks, especially in image and video generation. Extending these models to the discrete domain of natural language offers a promising direction for addressing key limitations of autoregressive LMs, including incoherent output (Holtzman et al., 2020), limited controllability (Zhang et al., 2023), and slow inference speed (Leviathan et al., 2023).

To apply diffusion models to text, one line of approaches transforms discrete text into a continuous latent space, applies a diffusion process and then decodes the output back into discrete text (Wu et al., 2023; Karimi Mahabadi et al., 2024), while another line of approaches designs discrete diffusion processes with new forward and reverse dynamics tailored to discrete tokens (Austin et al., 2021; Sahoo et al., 2024; Lou et al., 2024; Nie et al., 2025). Specifically, for a model distribution $p_\theta(x_0)$, the forward process gradually masks tokens in x_0 independently until yielding a fully masked sequence at $t = 1$. During the masking process $t \in (0, 1)$, each token is masked with probability t. The reverse process recovers the data distribution by iteratively predicting masked tokens as t decreases from 1 to 0 (Nie et al., 2025). A parametric mask predictor $p_\theta(\cdot \mid x_t)$ takes a partially masked x_t as input and simultaneously predicts all masked tokens, denoted $\mathbf{M}$. The model is trained using cross-entropy loss computed only over the masked tokens:

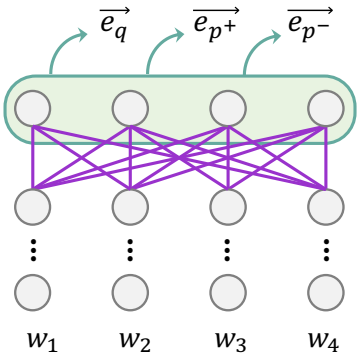


Figure 2: Overview of DIFFEMBED. Final-layer token representations from the backbone diffusion LM are mean-pooled to obtain text embeddings.

$$\mathcal{L}(\theta) \triangleq -\mathbb{E}_{t,x_0,x_t}\left[\frac{1}{t}\sum_{i=1}^{L}\mathbf{1}[x_t^i = \mathbf{M}]\log p_\theta(x_0^i \mid x_t)\right], \tag{2}$$

where x_0 is sampled from the training data, t is sampled uniformly from $[0, 1]$, and x_t is sampled from the forward process. The indicator function $\mathbf{1}[\cdot]$ ensures that the loss is computed only for masked tokens. Recent masked diffusion models such as DREAM (Ye et al., 2025) scale to over one billion parameters and has shown strong performance on multiple math and code reasoning tasks.

3 Diffusion Embedding Model

In this section, we propose a new type of text embedding model, the Diffusion Embedding Model (DIFFEMBED). In line with LLM-based text embedding models introduced in Section 2, DIFFEMBED first extracts contextualized token representations from the backbone diffusion LM and aggregate them using mean pooling[3] (see Figure 2). Then DIFFEMBED learns effective text representa-

[3]As recommended by BehnamGhader et al. (2024); Springer et al. (2025), mean pooling offers better performance and robustness than alternatives like last-token pooling.

Training Task	Dataset	Tot. Number		Query Length		Doc. Length	
		$\mathbb{N}$	**N**	Avg.	Std.	Avg.	Std.
Generalist embedding	Public E5	2M	16k	39	19	366	1062
Theorem reasoning	REASONAUG	10k	10k	230	244	508	427
Instruction following	MS MARCO†	1M	16k	83	76	92	41

Table 1: Statistics of datasets used for training embedding models. For each dataset, we show the original number of samples ($\mathbb{N}$), the amount for training subsets (N), rounded average length of queries and documents (measured by GPT-2 tokenizer (Radford et al., 2019)), and standard deviation. †MS MARCO with Instructions.

tions through contrastive learning on embedding tasks as in Equation (1). Unlike LLMs, diffusion LMs are inherently bidirectional, removing the need for intermediate steps such as enabling bidirectional attention and continue pre-training (Lee et al., 2024; BehnamGhader et al., 2024).

The primary distinction between DIFFEMBED and LLM-based embedding models lies in the mechanism by which embeddings are generated within the language models. LLM-based embeddings are obtained through large-scale pretraining using causal attention masking and a next-token prediction objective, which encourages the model to understand past context to predict future tokens (OpenAI et al., 2024; Dubey et al., 2024). In contrast, diffusion LM embeddings are learned through a denoising objective, where the model is trained to recover noisy or corrupted inputs (*e.g.,* masked tokens) simultaneously (Nie et al., 2025; Ye et al., 2025). This process enables the embeddings to capture bidirectional semantics and potentially more robust representations of input text.

4 Evaluating Text Embedding Models

To systematically evaluate DIFFEMBED, we compare it with LLM embedding models on diverse set of tasks including long-document retrieval (§4.1), reasoning-intensive retrieval (§4.2), instruction-following retrieval (§4.3), and traditional text embedding tasks (§4.4).

4.1 Long-document Retrieval

Encoding long documents is crucial for retrieval, yet prior embedding models are typically limited to inputs of 512 tokens (Zhu et al., 2024). To adapt diffusion LMs and LLMs for long-document retrieval, we train the DIFFEMBED and LLM embedding models on a subset[4] of the Public E5 dataset (Springer et al., 2025), using an input length of 4,096 tokens. This dataset is designed for developing general-purpose text embedding models and spans a wide range of tasks and document lengths (Table 1). We evaluate embedding models on the LONGEMBED benchmark Zhu et al. (2024), which includes two synthetic tasks (*i.e.,* finding the document containing a personalized passkey or fact needle) and four real-world tasks (*i.e.,* finding the document containing a correct answer or summarization), featuring documents of varying lengths and dispersed target information.

4.2 Reasoning-intensive Retrieval

Many real-world search queries require in-depth reasoning beyond lexical or semantic matching. Su et al. (2025) introduces the BRIGHT benchmark for this type of reasoning-intensive retrieval, *e.g.,* finding a tutorial for a math question related to the same theorem (TheoQ.) or reference code that implements the same algorithm as the programming problem (Leet.). However, at the time of this work, no existing dataset effectively supports this task.[5] Thus, we develop a new dataset REASONAUG using LLMs[6] for training embedding models for logical reasoning. REASONAUG mainly contains two types of task samples: (1) question-to-concept: retrieving the concept that is helpful for solving the given question; (2) question-to-question: retrieving the question that can be solved using the same concept as the given question. The trained models are evaluated on the math and code subsets of BRIGHT (*e.g.,* TheoremQA and Leetcode). The dataset construction includes following steps:

[4]Simulating low-resource settings, we train on the first N samples, which shows to capture most of performance.

[5]ReasonIR (Shao et al., 2025) recently released a dataset built on the BRIGHT corpus, whereas REASONAUG uses no BRIGHT documents. Our experiments on ReasonIR in Appendix F show findings consistent with REASONAUG.

[6]We experimented with GPT-4o-mini and DeepSeek-V3 for query generation. GPT-4o-mini produced more diverse and complex questions, so it was used for the final dataset.

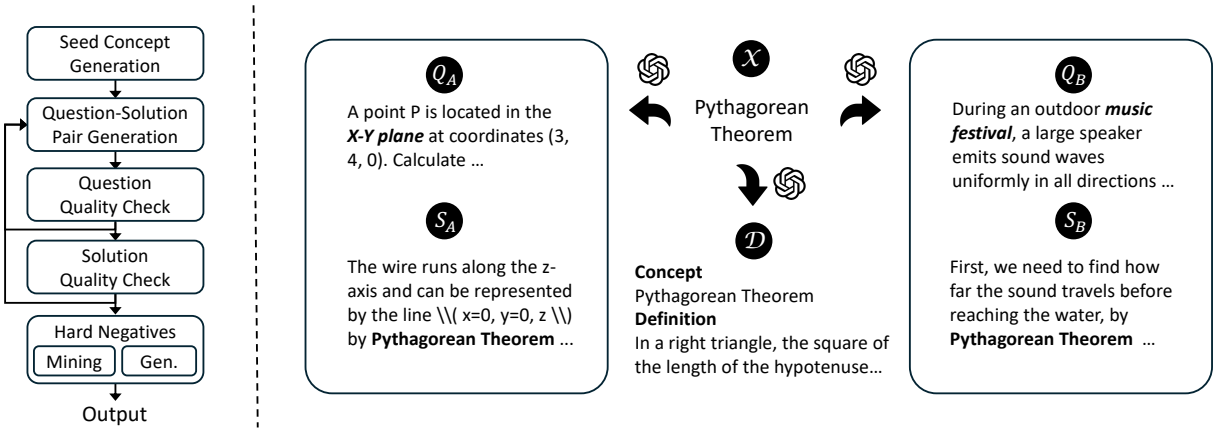


Figure 3: **Left**: data augmentation pipeline. **Right**: qualitative examples of seed concepts, their definitions, and associated question–solution pairs. A question-to-question retrieval sample can be constructed using a query question Q_A, a positive document (Q_B, S_B) generated from the same concept X, and a hard negative document (Q_C, S_C) generated from a different concept X'. A question-to-concept retrieval sample can consist of a query question Q_B, a positive document D (the definition of the relevant concept), and a hard negative document D' (the definition of a different concept X').

Concept, Question, and Solution Generation. As shown in Figure 3 Left, we begin by prompting LLMs to generate a list of concepts such as mathematical and physics theorems, programming algorithms, and financial equations (see Appendix A). For each seed concept, we create one definition and eight question-answer pairs as shown in Figure 3 Right, with the option of generating more. Our prompts are carefully crafted to produce questions that are novel, challenging, and varied in type (theoretical or applied), background context, and length—while remaining focused on the seed concept. In question-to-question retrieval, each question is used as a query, with other questions from the same concept serving as positives. For question-to-concept retrieval, the corresponding definition is used as the positive document.

Finally, to ensure there is no data leakage, we compare our generated questions against the BRIGHT queries using fuzzy string matching. We compute the longest contiguous matching subsequence and derive a similarity ratio. The highest similarity score observed is 0.5—well below the commonly used threshold of 0.8—indicating no meaningful overlap with the test data in BRIGHT.

Question and Solution Quality Check. We conduct a quality examination with two primary objectives: (1) Relevance — the generated question and solution should involve the seed concept; and (2) Correctness — the solution should correctly address the generated question. LLMs are instructed to discard any irrelevant or incorrect generations and produce appropriate replacements.

Hard Negative Mining and Generation. Hard negative documents are crucial for promoting fine-grained relevance discrimination. We construct hard negatives through both document mining and LLM generation. For mining, we retrieve lexically similar questions from the entire generation set using BM25 (MacAvaney et al., 2020), ensuring they are based on irrelevant and diverse concepts. For generation, we prompt LLMs to create novel questions with similar background but unrelated to the seed concept, ensuring they are not helpful for answering the query.

In total, we curate 10,896 triplets (query, positive, hard negative), with statistics in Table 1 and examples in Appendix G. The data creation prompts are described in Appendix B.

4.3 Instruction-Following Retrieval

Inspired by progress in instruction-tuned LMs, there is a growing demand for instruction-following retrievers. To this end, Weller et al. (2025) develop a instance-level instruction training set MS MARCO with Instructions. We train models using a subset of MS MARCO with Instructions and evaluate their instruction-following capabilities on the FOLLOWIR (Weller et al., 2024) benchmark.

4.4 Traditional Text Embedding Tasks

In addition to retrieval, we compare diffusion and LLM embedding models on the traditional text em-

	N	Synthetic (Acc@1)				Real (nDCG@10)				Avg.
		Pk≤4k	**Pk>4k**	**Nd≤4k**	**Nd>4k**	**NQA**	**QMS**	**SFD**	**WQA**	
E5-Mistral	2M	95.6	30.0	68.8	14.0	44.6	43.6	96.8	82.0	59.4
Llama3	16k	54.0	3.3	51.2	4.0	31.9	30.8	67.6	47.3	32.3
+ LLM2Vec	16k	59.6	4.0	59.2	6.7	36.5	34.4	84.5	51.1	42.0
Mistral	16k	87.6	22.0	72.4	20.0	35.0	36.8	85.4	62.9	52.8
+ LLM2Vec	16k	98.8	**30.0**	69.2	**21.3**	39.5	40.0	91.6	**78.0**	58.6
Qwen2.5	16k	86.8	24.6	68.0	19.3	31.7	35.6	88.0	68.3	52.8
+ LLM2Vec	16k	94.0	28.7	72.4	20.0	36.5	37.3	90.5	68.1	55.9
DIFFEMBED	16k	**100**	29.3	**86.8**	20.0	**42.1**	**43.8**	**98.0**	77.2	**62.2**

Table 2: Results on long document retrieval (LONGEMBED) for embedding models trained with **E5** text embedding data. *N* is the amount of E5 samples used for training. *Pk≤4k* refers to *Passkey* Retrieval with an evaluation length less than or equal to 4,096. *Nd>4k* refers to *Needle-in-a-haystack* Retrieval with length larger than 4,096. *NQA*, *QMS*, *SFD*, *WQA* is short for *NarrativeQA*, *QMSum*, *SummScreenFD*, *2WikiMultihopQA*, respectively. The best results with 16k training data are in **bold**.

bedding tasks. Following BehnamGhader et al. (2024), we train the models using the same subset of the Public E5 dataset as in Section 4.1, and evaluate them on the Massive Text Embedding Benchmark (MTEB) (Muennighoff et al., 2023), which includes tasks such as reranking, clustering, classification, semantic textual similarity, and more.

5 Experiments

As detailed in Section 4, we compare DIFFEMBED and LLM embeddings in four setups for different capabilities. In each setup, we fine-tune the models using one dedicated dataset.

5.1 Models

We take DREAM-v0-Instruct-7B (Ye et al., 2025) as the backbone model for DIFFEMBED. Initialized from Qwen2.5-7B (Yang et al., 2024), DREAM is pre-trained on 580 billion tokens[7] covering text, code, and math, and achieves performance competitive with leading AR LMs of similar scale. For comparison, we include similarly sized autoregressive models: Llama-3-8B-Instruct (Dubey et al., 2024), Mistral-7B-Instruct-v0.2 (Jiang et al., 2023), and Qwen2.5-7B-Instruct (Yang et al., 2024). We also include LLM2Vec (BehnamGhader et al., 2024), which adapts LLMs to use bidirectional attention via two intermediate pre-training steps: Masked Next Token Prediction (MNTP) and unsupervised contrastive learning (Gao et al., 2021) using Wikitext data. We focus on MNTP, which has a greater impact on final performance in line with BehnamGhader et al. (2024), and report unsupervised learning results in Appendix C.

To ensure fair comparisons, we adopt the same model configuration for all embedding models, including the contrastive learning objective, mean pooling, bidirectional attention masking, cosine similarity, and other settings. For training with Public E5 and REASONAUG datasets, we follow the configuration of LLM2Vec (BehnamGhader et al., 2024), while for the training with MS MARCO with Instructions dataset, we adopt the setup from Promptriever (Weller et al., 2025).

5.2 Long Document Retrieval

Table 2 compares DIFFEMBED and LLM embedding models on the LONGEMBED benchmark. We find that LLM2Vec brings an average 6.2% gain for LLM embedding models, this is because learning to use bidirectional attention introduces potential benefits such as capturing long-range dependencies. Our synthetic experiments further reveal that, although models are trained with an input length of 4,096 tokens, LLM embedding models may still fail to encode key information effectively (Kamradt, 2024). As expected, DIFFEMBED with bidirectional attention during large-scale pretraining, aces the synthetic tasks (with 100% and 86.8% accuracy) and outperforms all LLM embedding models on real-world tasks.

[7]Considering this corpus is significantly smaller than the multi-trillion-token pre-training of Qwen2.5, we believe this phase primarily serves to adapt the generation paradigm from autoregressive to diffusion, rather than to introduce substantial new knowledge. It does not overlap with benchmark data.

	N	TheoT.	TheoQ.	AoPS	Leet.	Average
E5-Mistral	-	26.8	26.1	7.1	28.7	22.2
ReasonIR		27.2	31.9	14.7	35.0	27.2
Llama3	10k	31.0	34.6	5.8	18.9	22.6
+ LLM2Vec	10k	28.3	33.8	12.6	23.6	24.6
Mistral	10k	32.4	33.7	9.4	20.7	24.1
+ LLM2Vec	10k	30.8	30.4	9.1	22.9	23.3
Qwen2.5	10k	34.7	40.2	**15.5**	**32.0**	30.6
+ LLM2Vec	10k	32.3	37.2	10.3	31.0	27.7
DIFFEMBED	10k	**38.9**	**48.3**	15.4	30.0	**33.2**

Table 3: Results on theorem-based benchmarks in reasoning-intensive retrieval (BRIGHT) for embedding models trained using REASONAUG. We report nDCG@10 for *TheoremQA* with with theorem retrieval (*TheoT.*) and question retrieval (*TheoQ.*), *AoPS*, and *LeetCode* (*Leet.*). *N* is the amount of REASONAUG samples used for training. The best results with 10k training data are in **bold**.

	N	Robust04		News21		Core17		Average	
		MAP	MRR	nDCG	MRR	MAP	MRR	Score	MRR
Prompt -riever	1M	28.3	+11.7	28.5	+6.4	21.6	+15.4	26.1	+11.2
Llama3	16k	12.7	+2.3	17.6	-1.0	11.7	+1.7	14.0	+1.0
+ LLM2Vec	16k	17.6	+1.9	21.6	+1.2	16.7	+4.7	18.6	+2.6
Mistral	16k	**21.4**	**+7.0**	27.4	+1.1	16.0	+7.2	**21.5**	**+5.1**
+ LLM2Vec	16k	20.7	+5.7	25.1	+1.1	**16.3**	**+7.5**	20.7	+4.8
Qwen2.5	16k	15.6	+1.4	20.4	+1.7	13.9	+4.4	16.6	+2.5
+ LLM2Vec	16k	14.5	+1.6	19.7	+0.1	13.8	+2.8	16.0	+1.5
DIFFEMBED	16k	18.9	+5.7	**27.7**	**+3.6**	16.2	+6.0	20.9	**+5.1**

Table 4: Results on instruction-following document retrieval (FOLLOWIR) for embedding models trained using **MS MARCO with Instructions**. *N* is the amount of MS MARCO samples used for training. MAP@1000/nDCG@5 range from 0-100. MRR is short for the pairwise evaluation metric measuring instruction following when instructions change, ranging from -100 to 100 (higher is better). The best results trained with 16k data are in **bold**.

5.3 Reasoning-Intensive Retrieval

Compared to LONGEMBED, the logical reasoning-based tasks[8] in BRIGHT involve shorter queries but more complex documents, require stronger reasoning capabilities. According to Table 3, LLM2Vec is ineffective, likely due to limited data scale and complexity in its continue pre-training. All fine-tuned embedding models outperform the general-purpose E5-Mistral and perform on par with the reasoning-focused retriever ReasonIR (Shao et al., 2025), highlighting the effectiveness of our lightweight dataset REASONAUG. Among LLM embeddings, Qwen2.5 achieves the highest overall performance, likely benefiting from its strong mathematical and coding capabilities. DIFFEMBED further improves on Qwen2.5, with gains of +4.2 and +8.1 points on the TheoT. and TheoQ. tasks, substantially outperforming the rest models. These improvements suggest that the ability to attend to both past and future context is essential for understanding complex logic, theorems and equations. On AoPS and Leet., DIFFEMBED shows relatively smaller improvements, as discussed further in Section 6 Q3.

5.4 Instruction-Following Retrieval

As shown in Table 4, Mistral achieves the best performance among the models. As noted by BehnamGhader et al. (2024), bidirectional atten-

[8]Other BRIGHT tasks, such as StackExchange (non-logical content) and Pony (a niche programming language), differ significantly from the domain of REASONAUG, thus we have excluded them. Our results show that there is no significant performance gains on these tasks using REASONAUG.

tion may be effective for Mistral even without LLM2Vec pre-training. DIFFEMBED demonstrates comparable instruction-following ability to Mistral (*i.e.,* +5.1 pairwise MRR). We hypothesize that the limited length[9] and low complexity of queries and documents constrain the benefits of DIFFEMBED's bidirectional embedding approach.

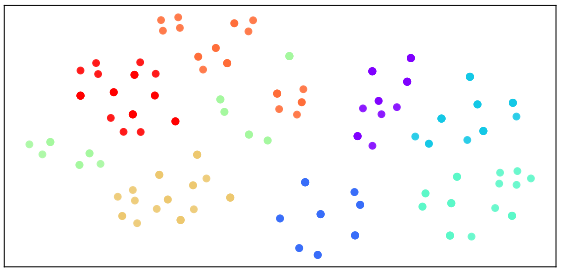
5.5 General Text Embedding Tasks

We evaluate the DIFFEMBED and LLM embedding models on 15 general text embedding tasks on MTEB. According to Table 6, DIFFEMBED performs on par with LLM embedding models, as expected. Similar to Section 5.4, most traditional tasks involve shorter inputs and less reasoning, limiting DIFFEMBED's benefits.

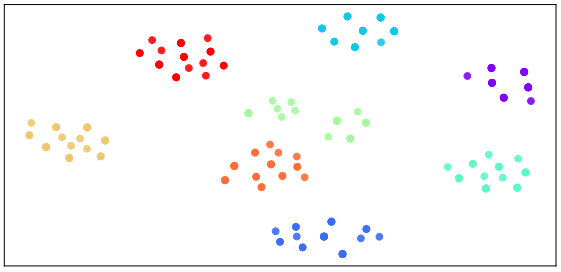
6 Analysis

Q1: Why do generalist embedding models underperform in reasoning-intensive retrieval? Although identifying questions that share the same theorem resembles a clustering task, traditional embedding models are not trained to recognize theorems or use them as clustering signals. As shown in Figure 4, the E5-Mistral embeddings of REASONAUG documents are more dispersed, lacking clear cluster boundaries[10]. Furthermore, the case study in Table 11 shows that E5-Mistral often relies on superficial lexical cues (e.g., exact numbers like "10" or keywords like "smallest") and shallow semantic patterns (e.g., "what is the count" vs. "how many") when matching questions. Detailed studies are presented in Appendix D.

[9]Following Promptriever, we use inputs of max 304 tokens.



(a) E5-Mistral without REASONAUG fine-tuning.



(b) DIFFEMBED fine-tuned on REASONAUG.

Figure 4: t-SNE visualization of document embeddings from REASONAUG. The documents are grouped and color-coded by concept (see legend in Figure 6). Mathematical theorems include *Vieta's Formulas*, *Pigeonhole Principle*, *Euler's Identity*, and *Central Limit Theorem*. Algorithmic concepts include *Two Pointers*, *N-Queens Problem*, *Sweep Line Algorithm*, and *Kahn's Algorithm*.

	Bidirectional	Unidirectional
	Mistral	
TheoT.	32.4	4.0 (-28.4)
TheoQ.	33.7	9.6 (-24.1)
AoPS	9.4	7.8 (-1.6)
Leet.	20.7	34.9 (+14.2)
	DIFFEMBED	
TheoT.	38.9	1.1 (-37.8)
TheoQ.	48.3	0.7 (-47.6)
AoPS	15.4	2.8 (-12.6)
Leet.	30.0	29.5 (-0.5)

Table 5: Ablation study comparing BRIGHT performance when using bidirectional vs. unidirectional attention at test time. Both models are trained with bidirectional attention and REASONAUG data.

Q2: How important is bidirectional attention for different models and tasks? To assess the role of bidirectional attention, we ablate two embedding models, Mistral and DIFFEMBED on BRIGHT. For each model, we compare testing performance using full bidirectional (causal and reverse) attention versus unidirectional (causal only) attention. According to Table 5, one notable finding is that DIFFEMBED exhibits a substantially larger performance drop when reverse attention is disabled, which suggests that DIFFEMBED depends more heavily on bidirectional context, likely due to its bidirectional pre-training. Furthermore, tasks differ in their sensitivity to reverse attention. Performance on the Leet. task remains relatively stable (even increase) without reverse attention, while performance on the TheoQ. task degrades significantly. This indicates the bidirectional attention is critical

[10]Most REASONAUG questions focus on a single concept, which is generally distinct from others; therefore, clear boundaries among concept clusters are expected.

for logical reasoning tasks like TheoremQA, supporting the performance gains observed for DIFFEMBED on TheoT. and TheoQ. in Section 5.3.

Q3: Why are improvements less evident in Leet. and AoPS compared to TheoQ. in BRIGHT? To better understand the difference between TheoQ., Leet. and AoPS, we conduct a comparative analysis in Table 8 and a human re-evaluation for gold documents in Table 9. Our analysis reveals notable noise in the gold annotations and corpora of Leet. and AoPS, impacting the evaluation results (see Appendix E). In particular, we observe multiple types of relevance (*e.g.,* contextual and algorithmic) annotated in Leet., while it is more consistent in TheoT. and TheoQ..

Q4: Does the performance gain diminish as the training dataset size increases? Figure 5 illustrates the TheoQ. performance of DIFFEMBED and its base model, Qwen2.5, as the training size varies from 2k to 10k. The performance gap remains substantial within this range, suggesting the LLM-based models may need extensive data to learn effective bidirectional attention as DIFFEMBED.

7 Conclusion

This paper presents the first exploration of using diffusion language models as text embedding models—a intuitively motivated direction. Through extensive experiments on long-document retrieval (LONGEMBED), reasoning-intensive retrieval (BRIGHT), instruction-following retrieval (FOLLOWIR), and general embedding tasks (MTEB), we demonstrate the advantages of diffusion embeddings over LLM embeddings in capturing global context for long and complex text. We attribute these gains to large-scale bidirectional pre-training. We hope this work provides meaningful insights for both the text embedding community and the development of diffusion LMs.

Limitations

We evaluate only the state-of-the-art diffusion LM, DREAM (Ye et al., 2025). Other models, such as LLaDA (Nie et al., 2025), are expected to exhibit inferior text embedding performance given their comparatively weaker generative and reasoning abilities. Due to resource constraints, we limit the training scale within 20k samples. Larger-scale experiments with millions of examples could reveal further insights. In REASONAUG, irrelevant or incorrect documents may still remain when LLMs fail to identify them during the quality check steps. However, contrastive training is resilient to a certain degree of noise in the data.

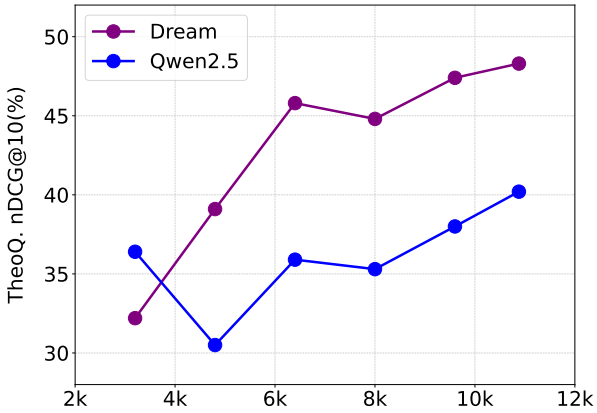


Figure 5: Retrieval performance on TheoQ. for Dream and Qwen2.5 models trained with varying amounts of REASONAUG data.

Acknowledgements

This research is supported by the RIE2025 Industry Alignment Fund – Industry Collaboration Projects (IAF-ICP) (Award I2301E0026), administered by A*STAR, as well as supported by Alibaba Group and NTU Singapore through Alibaba-NTU Global e-Sustainability CorpLab (ANGEL). Siyue Zhang and Chen Zhao were supported by NYU Shanghai Center for Data Science. This work was supported in part through the NYU IT High Performance Computing resources, services, and staff expertise.

References

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021. Structured denoising diffusion models in discrete state-spaces. In *Thirty-Fifth Annual Conference on Neural Information Processing Systems*.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. LLM2Vec: Large language models are secretly powerful text encoders. In *First Conference on Language Modeling*.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of

deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*.

Enjun Du, Siyi Liu, and Yongqi Zhang. 2025. Mixture of length and pruning experts for knowledge graphs reasoning. ArXiv preprint arXiv:2507.20498.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaoqing Ellen Tan, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aaron Grattafiori, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alex Vaughan, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Franco, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, Danny Wyatt, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Firat Ozgenel, Francesco Caggioni, Francisco Guzmán, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Govind Thattai, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Karthik Prasad, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kun Huang, Kunal Chawla, Kushal Lakhotia, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng

Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Maria Tsimpoukelli, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikolay Pavlovich Laptev, Ning Dong, Ning Zhang, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Rohan Maheswari, Russ Howes, Ruty Rinott, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Kohler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vítor Albiero, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaofang Wang, Xiaojian Wu, Xiaolan Wang, Xide Xia, Xilun Wu, Xinbo Gao, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yuchen Hao, Yundi Qian, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, and Zhiwei Zhao. 2024. The llama 3 herd of models.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *The Eighth International Conference on Learning Representations*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

Jina. 2024. Jina reranker v2 for agentic rag: Ultra-fast, multilingual, function-calling and code search. https://jina.ai/news/jina-reranker-v2-for-agentic-rag-ultra-fast-multilingual-function-calling-and-code-search/. Accessed: 2024-11-26.

Greg Kamradt. 2024. Llm test - needle in a haystack. https://github.com/gkamradt/LLMTest_NeedleInAHaystack. Accessed: 2025-05-19.

Rabeeh Karimi Mahabadi, Hamish Ivison, Jaesung Tae, James Henderson, Iz Beltagy, Matthew Peters, and Arman Cohan. 2024. TESS: Text-to-text self-conditioned simplex diffusion. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2024. Nv-embed: Improved techniques for training llms as generalist embedding models.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*.

Chaofan Li, Zheng Liu, Shitao Xiao, and Yingxia Shao. 2023. Making large language models a better foundation for dense retrieval.

Aaron Lou, Chenlin Meng, and Stefano Ermon. 2024. Discrete diffusion language modeling by estimating the ratios of the data distribution.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2024. Fine-tuning llama for multi-stage text retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, Nazli Goharian, and Ophir Frieder. 2020. Efficient document re-ranking for transformers by precomputing term representations. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Niklas Muennighoff, Hongjin Su, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. 2025. Generative representational instruction tuning. In *The Thirteenth International Conference on Learning Representations*.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. Large language diffusion models.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*.

Subham Sekhar Sahoo, Marianne Arriola, Aaron Gokaslan, Edgar Mariano Marroquin, Alexander M Rush, Yair Schiff, Justin T Chiu, and Volodymyr Kuleshov. 2024. Simple and effective masked diffusion language models.

Rulin Shao, Rui Qiao, Varsha Kishore, Niklas Muennighoff, Xi Victoria Lin, Daniela Rus, Bryan Kian Hsiang Low, Sewon Min, Wen tau Yih, Pang Wei Koh, and Luke Zettlemoyer. 2025. Reasonir: Training retrievers for reasoning tasks.

Zhang Siyue, Xue Yuxiang, Zhang Yiming, Wu Xiaobao, Luu Anh Tuan, and Zhao Chen. 2024. Mrag:

A modular retrieval framework for time-sensitive question answering.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*.

Tingyu Song, Guo Gan, Mingsheng Shang, and Yilun Zhao. 2025. Ifir: A comprehensive benchmark for evaluating instruction-following in expert-domain information retrieval. In *Proceedings of 2025 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2021. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*.

Jacob Mitchell Springer, Suhas Kotha, Daniel Fried, Graham Neubig, and Aditi Raghunathan. 2025. Repetition improves language model embeddings. In *The Thirteenth International Conference on Learning Representations*.

Hongjin Su, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han yu Wang, Haisu Liu, Quan Shi, Zachary S. Siegel, Michael Tang, Ruoxi Sun, Jinsung Yoon, Sercan O. Arik, Danqi Chen, and Tao Yu. 2025. Bright: A realistic and challenging benchmark for reasoning-intensive retrieval. In *The Thirteenth International Conference on Learning Representations*.

Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. 2021. Augmented SBERT: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2024a. Text embeddings by weakly-supervised contrastive pre-training.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024b. Improving text embeddings with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. 2024. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference.

Orion Weller, Benjamin Chang, Sean MacAvaney, Kyle Lo, Arman Cohan, Benjamin Van Durme, Dawn Lawrie, and Luca Soldaini. 2024. Followir: Evaluating and teaching information retrieval models to follow instructions.

Orion Weller, Benjamin Van Durme, Dawn Lawrie, Ashwin Paranjape, Yuhao Zhang, and Jack Hessel. 2025. Promptriever: Instruction-trained retrievers can be prompted like language models. In *The Thirteenth International Conference on Learning Representations*.

Tong Wu, Zhihao Fan, Xiao Liu, Yeyun Gong, Yelong Shen, Jian Jiao, Hai-Tao Zheng, Juntao Li, Zhongyu Wei, Jian Guo, Nan Duan, and Weizhu Chen. 2023. Ar-diffusion: Auto-regressive diffusion model for text generation. In *Thirty-Seventh Annual Conference on Neural Information Processing Systems*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. 2024. Qwen2 technical report.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. 2025. Dream 7b: Introducing dream 7b, the most powerful open diffusion large language model to date.

Honghua Zhang, Meihua Dang, Nanyun Peng, and Guy Van Den Broeck. 2023. Tractable control for autoregressive language generation. In *Proceedings of the 40th International Conference on Machine Learning*.

Dawei Zhu, Liang Wang, Nan Yang, Yifan Song, Wenhao Wu, Furu Wei, and Sujian Li. 2024. LongEmbed: Extending embedding models for long context retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.

Task	Llama3	Mistral	Dream
ArguAna	57.33	54.46	58.47
Banking77Classification	87.35	87.03	87.50
BiorxivClusteringS2S	38.45	35.80	35.56
EmotionClassification	47.81	50.81	51.44
MassiveIntentClassification	78.69	76.89	75.64
MedrxivClusteringS2S	33.46	31.26	31.42
NFCorpus	38.04	39.27	35.44
SciDocsRR	85.19	84.41	82.00
SciFact	75.93	71.21	71.42
SICK-R	80.64	78.92	76.49
SprintDuplicateQuestions	92.30	95.28	95.30
StackOverflowDupQuestions	55.32	54.32	50.69
STS17	87.30	88.15	87.83
STSBenchmark	84.76	85.47	83.43
TwentyNewsgroupsClustering	53.28	49.50	52.56
Average	66.39	65.52	65.01

Table 6: Evaluation on 15 general text embedding tasks selected by BehnamGhader et al. (2024) from MTEB for Llama3 + LLM2Vec, Mistral + LLM2Vec, and Dream models, which are trained with 16k Public E5 data and bidirectional attention.

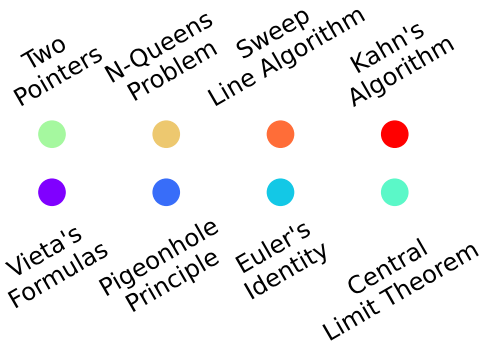


Figure 6: The concept color mapping for Figure 4.

A REASONAUG Seed Concept List

60 Algorithms: Sweep Line Algorithm, Kahn's Algorithm, Dijkstra's Algorithm, Game Theory, Two Pointers, N-Queens Problem, Depth First Search (DFS), Prefix Sum, Greedy Algorithm, Bucket Sort, Breadth First Search (BFS), Longest Common Subsequence (LCS), Huffman Coding, Manhattan Distance, Topological Sorting, Rod Cutting Problem, Binary Search, Knapsack Algorithm (0/1 Knapsack), Floyd-Warshall Algorithm, Bellman-Ford Algorithm, Merge Sort, Quick Sort, Heap Sort, Bubble Sort, Insertion Sort, Selection Sort, Kruskal's Algorithm, Prim's Algorithm, Kadane's Algorithm, Rabin-Karp Algorithm, Knuth-Morris-Pratt (KMP) Algorithm, Boyer-Moore Algorithm, Longest Increasing Subsequence (LIS), Edit Distance, Sieve of Eratosthenes, Tarjan's Algorithm, Kosaraju's Algorithm, Z Algorithm, LRU Cache Algorithm, A-star search algorithm, Hamiltonian Path, Substring Search Algorithm, Permutations, Combinations, Knapsack DP with Bitmasking, Matrix Exponentiation, Square Root Decomposition, Mo's Algorithm, Strassen's Algorithm, K-Means Clustering, Gradient Descent, Support Vector Machines (SVM), Aho-Corasick Algorithm, Ford-Fulkerson Algorithm, Trapping Rain Water, Matrix Chain Multiplication, Coin Change Problem, Palindrome Partitioning, Sudoku Solver, Newton's Method.

90 Math Theorems: Newton's Sums, Pigeonhole Principle, Chicken McNugget Theorem, Simon's Favorite Factoring Trick, Fermat's Little Theorem, Ptolemys theorem, Euler's Identity, Euclidean algorithm, Cauchy-Riemann Equations (Complex Analysis), Vieta's Formulas, Triangle Inequality, Power of a Point, Central Limit Theorem, Pick's Theorem, Shoelace Theorem, Legendre's formula, Principle of Inclusion Exclusion, Ceva's Theorem, Logarithm: Change of Base Formula, Stars and Bars formula, Eigenvalue equation, Intermediate Value Theorem, Mass point geometry theorem, Geometric probability in 2D, Fourier Transform, Cramer's Rule, Vertex cover in graph theory, One-sample t-test, Z-transform, Ramsey's Theorem, Pollard's Rho Algorithm (Factorization), Chinese Remainder Theorem, Taylor's Theorem, Addition of Multiindices, Bayes' Theorem, Binomial Theorem, Mean Value Theorem for Derivatives, Pythagorean Theorem, Lagrange's Theorem (Group Theory), The Chain Rule in calculus, Green's Theorem, Cauchy-Schwarz Inequality, Divergence Theorem, Second Part of the Fundamental Theorem of Calculus (FTC2), Quadratic Formula (for polynomials of degree 2), Fundamental Theorem of Arithmetic, Rolle's Theorem, De Moivre's Theorem, Law of Large Numbers, Cayley-Hamilton Theorem, L'Hôpital's Rule, Singular Value Decomposition (SVD) Theorem, The Squeeze Theorem, Brouwer Fixed-Point Theorem, Tychonoff's Theorem, Bézout's Theorem, Vandermonde's Identity, Wilson's Theorem, Markov Property, Invertible Matrix Theorem, Sylow Theorems, Cantor's Theorem, Heron's Formula (for the area of a triangle), Laplace Transform, Bolzano-Weierstrass Theorem, Weierstrass Approximation Theorem, Cauchy's Mean Value Theorem, Lindelof's Theorem, Poisson Limit Theorem, Mertens' Theorem, Chebyshev's Inequality, Markov's Inequality, Jensen's Inequality, Borel-Cantelli Lemma, Chauvenet's Criterion, Helly's Theorem, Holder's Inequality, Minkowski's Inequality, Euler's Formula for Planar Graphs, Hahn Decomposition Theorem, Radon-Nikodym Theorem, Kakutani Fixed-Point Theorem, Sum of a Geometric Series Formula, The Isoperimetric Inequality, Spectral Theorem, Power Rule (for derivatives), Hadamard's Determinant Theorem, Borel's Theorem, Runge's Theorem, Euler's Formula for Polyhedra, Integral of a Power Function Formula, Poincaré Recurrence Theorem, Extreme Value Theorem, Dirichlet's Theorem on Primes in Arithmetic Progressions, Lefschetz Fixed-Point Theorem, Seifert-van Kampen Theorem, Hurewicz Theorem, Frobenius' Theorem, Formula for Permutations (without repetition), Formula for Combinations (with repetition).

80 Physics Theorem: Center-of-Mass Energy, Planck's energy-frequency relation, Magnification theorem, Maximum Entropy Principle, Heron's Formula, Gibbs Free Energy, Ideal Gas Law, Torricelli's Law, Coulomb's Law, Gauss's Law for Electricity, Kirchhoff's Laws, Ohm's Law, Millma's Theorem, Carnot's Theorem, Beer-Lambert Law, Newton's Laws of Motion, Lorentz Force, First Law of Thermodynamics, Work-Energy Theorem, Maxwell's Equations, Conservation of Mechanical Energy, Kinetic Energy Theorem for Rotational Motion, Conservation of Angular Momentum, Torque-Angular Momentum Theorem, Centripetal Force Formula (for an object in circular motion), Euler's Rotation Theorems, Parallel Axis Theorem, Elastic Collision, Boucherot's Theorem (Power Factor Theorem), Tellegen's Theorem, Law of Reflection, Malus's Law, Specific Heat Capacity Formula, Optical Path Length (OPL) Theorem, Snell's Law, Huygens' Principle, Young's Double-Slit Experiment, Fraunhofer Diffraction Theory, Fresnel Equations, Planck's Law of Blackbody Radiation, Stefan-Boltzmann Law, Wien's Displacement Law, Rayleigh-Jeans Law, Compton Scattering Formula, Electric Field Formula (from a point charge), Speed of Sound in air (at temperature T), Heat Transfer via Conduction (Fourier's Law), Pauli Exclusion Principle, Energy Stored in a Capacitor Formula, Einstein's Photoelectric Equation, Bragg's Law, Gauss's Law for Magnetism, Faraday's Law of Induction, Lenz's Law, Work Done in an Adiabatic Process (Thermodynamics) Formula, Ampère's Circuital Law, Hooke's Law (for Springs), Laplace's Equation, Poisson's Equation, D'Alembert's Principle, Lagrange's Equations of Motion, Hamilton's Principle, Virial Theorem, Kepler's Laws of Planetary Motion, Newton's Law of Universal Gravitation, Magnetic Force on a Moving Charge Formula, Schwarzschild Metric, Lorentz Transformation, Einstein's Energy-Mass Equivalence, Shannon Entropy, Dirac Equation, Feynman Path Integral Formulation, Landauer's Principle, Onsager Reciprocity Relations, Bernoulli's Equation (for fluid flow), Stokes' Law, Reynolds Transport Theorem, Conservation of Mass, Thermal Conductivity Formula, Lens Equation (for a thin lens).

30 Finance Formula: Binomial Model in finance, Net Present Value (NPV), Future Value (FV) Formula, Present Value (PV) Formula, Discounted Cash Flow (DCF) Model, Dividend Discount Model, Capital Asset Pricing Model (CAPM), Gordon Growth Model (GGM), Binomial Option Pricing Model, Bond Pricing Formula, Yield to Maturity (YTM), Sharpe Ratio, Macauley Duration, Modified Duration, Internal Rate of Return (IRR), Return on Equity (ROE), Value at Risk (VaR) Formula, Z-Spread Formula, Inventory Turnover Ratio, GDP (Expenditure Approach), DuPont Analysis Formula, Weighted Average Cost of Capital (WACC), Derivatives Forward Price, Dividend Discount Model (DDM), Earnings Yield Formula, Sustainable Growth Rate (SGR), Operating Leverage Formula, Covariance Formula, Variance of a Two-Asset Portfolio, Profitability Index (PI).

B Data Augmentation Prompts for LLM

Theorem Definition Prompt

Your task is to provide a definition for the {domain}: {theorem}. Write the equation in LaTex format.

Here are some examples:
Concept
Pigeonhole Principle
Definition
Let S be a finite set whose cardinality is n. Let $S_1, S_2, \ldots, S_k$ be a partition of S into k subsets. Then, at least one subset S_i of S contains at least $\left\lceil \frac{n}{k} \right\rceil$ elements, where $\lceil \cdot \rceil$ denotes the ceiling function.

Here is your task:
Concept
{theorem}
Definition

Question Quality Check Prompt

Question
{question}

Is this problem testing or requiring {domain} {theorem}? If yes, please answer "YES". If no, please response with a new problem and solution about {theorem} with similar context and difficulty. Do not provide any explanation.

Solution Quality Check Prompt

Question
{question}
Solution
{solution}

Is this a correct solution to the problem and using the {domain} {theorem}? Response "YES" or "No".'

Math Question-Solution Generation Prompt

Your task is to create one {question type} problem with a correct solution.

- The problem should be new and unique, not similar to common existing problems.
{"- The problem should be based on real world human activities but not a proof problem.", "- The problem should a multi-choice problem but not a proof problem.", "- The problem should be theoretical and mathematical but not a proof problem."}
- The problem should involve numerical operations.
- Most importantly, the problem should require or test about the {domain}: {theorem}.
- The problem should not explicitly mentioning theorem.
- The problem should be as difficult as {"American Mathematics Competitions", "International Mathematical Olympiad", "Scholastic Assessment Test Math Exam"}.
- The problem should be solved {"in multiple steps", "by multiple domains"}.
{"- The problem should be around four sentences long.", None}
{"- The solution should not explicitly mention {theorem}.", None}
- The solution should include reasoning or calculation steps.

Write the problem after the **Problem** tag and the solution after the **Solution** tag. Do not write any explanation.

Coding Question-Solution Generation Prompt

Your task is to create one {question type} problem with a correct solution.

- The problem should be new and unique, not similar to common existing problems.
{ "- The problem should be based on real world human activities.", "- The problem should be based on a theoretical coding context.", "- The problem should be about a company or a factory.", "- The problem should be about a game or a puzzle.", "- The problem should be about designing a system.", "- The problem should be about a mathematical task needing automation.", "- The problem should be about traffic or logistics.", "- The problem should be about a city or a community.", "- The problem should be about fiance or business.", "- The problem should be about software or mobile applications.", "- The problem should be about education or e-learning.", "- The problem should be about e-commerce or online marketplaces.", "- The problem should be about agriculture or food production.", "- The problem should be about health or fitness.", "- The problem should be about customer service.", "- The problem should be about environmental sustainability.", "- The problem should be based on real world human activities.", None}
- Most importantly, the problem should require or test about the {domain}: {theorem}.
- The problem should be as difficult as {"LeetCode", "Codeforces Contests", "Google Code Jam"}.
- The solution code should be written in the programming language {language}.

Write the problem after the **Problem** tag and the solution after the **Solution** tag. Do not write any explanation.

Physics Question-Solution Generation Prompt

Your task is to create one {question type} problem with a correct solution.

- The problem should be new and unique, not similar to common existing problems.
{"- The problem should be based on real world human activities.", None}
- Most importantly, the problem should require or test about the {domain}: {theorem}.
- The problem should be as difficult as {"International Physics Olympiad (IPhO)", "American Invitational Physics Exam (AIPMT)", "Scholastic Assessment Test (SAT) Physics Subject Test"}.
- The problem should be solved {"in multiple steps", "by multiple domains"}.
- The solution should include reasoning or calculation steps.

Write the problem after the **Problem** tag and the solution after the **Solution** tag. Do not write any explanation.

Hard Negative Generation Prompt for LLM

You have been assigned a retrieval task: {instruction}

You will be given a user query and a positive document. Your mission is to write one hard negative document. The hard negative document must:
- Have the similar context background as the user query but test or require a different {domain}.
- Follow the format of the positive document.
- Should not be related to {theorem}.
- Should not be helpful for solving the user query problem.

User Query
{query}

Positive Document
{pos}

Directly response with the content of hard negative document.
Hard Negative Document

B.1 Training Details

For all experiments, we use language models with a bidirectional attention mask, mean pooling, and contrastive learning objectives. Training is conducted on 4 A6000 GPUs using LoRA with the specified ranks. For models with MNTP, we initialize the models from pre-trained MNTP checkpoints provided by BehnamGhader et al. (2024).

MS MARCO with Instructions. We train for 1 epoch using a batch size of 8 per GPU, 4 gradient accumulation steps, a learning rate of 1e-4 with 20 warmup steps, and LoRA rank 32. Only explicit negatives are used. Maximum input and document lengths are 304 and 196 tokens, respectively.

Public E5 and REASONAUG. We train for 1 epoch with a batch size of 4 per GPU, 1 gradient accumulation step, a learning rate of 1e-4 with 100 warmup steps, and LoRA rank 16. Both hard negatives and in-batch negatives are employed. Maximum input and document lengths are set to 4096 tokens.

B.2 Evaluation Metrics

We use the MTEB (Muennighoff et al., 2023) implementations to evaluate LONGEMBED, BRIGHT, FOLLOWIR, and general text embedding benchmarks (e.g., Table 6). The reported scores follow the main evaluation metrics defined in the original benchmark papers: Acc@1 for synthetic tasks in LONGEMBED, nDCG@10 for real tasks in LONGEMBED and all tasks in BRIGHT, nDCG@5 for News21 in FOLLOWIR, MAP@1000 for Robust04 and Core17 in FOLLOWIR, and p-MRR for all subsets in FOLLOWIR. The definitions of these metrics refer to original papers.

Task instructions used in BRIGHT are as follows:

- **TheoT.**: Given a problem, retrieve the relevant theorems that help solve the given problem.
- **TheoQ.**: Given a problem, retrieve the relevant problems that can be solved by the similar theorems.
- **AoPS**: Given a problem, rretrieve the relevant problems that can be solved by the similar math theorems.
- **Leet.**: Given a coding problem, retrieve the relevant problems that can be solved by the similar algorithms or data structures.

We follow LLM2Vec (BehnamGhader et al., 2024) for other task instructions.

C Unsupervised Contrastive Learning

We conduct additional experiments for three base models to evaluate whether unsupervised learning methods, such as SimCSE (Gao et al., 2021), enhance retrieval performance on reasoning-intensive tasks like BRIGHT (Su et al., 2025). As shown in Table 7, no significant improvement is observed, consistent with findings by BehnamGhader et al. (2024). This can be attributed to the fundamental mismatch between semantic similarity objectives in unsupervised sentence embeddings and the theorem-centric nature of BRIGHT, which demands deeper reasoning beyond surface-level semantics. Therefore, we exclude these results in the main paper.

D Case Study

To further analyze models' behaviors, we collect statistics on the top 10 retrieved documents from DIFFEMBED and E5-Mistral for the Leet. and TheoQ. tasks. As shown in Table 8, while both models perform similarly on Leet., each model retrieves the correct document in approximately 70 cases, but there are about 20 cases where the other model fails, suggesting the presence of multiple relevance definitions in Leet. This discrepancy is less pronounced in TheoQ., indicating more consistent relevance annotations. Notably, DIFFEMBED demonstrates more significant improvements on TheoQ. and TheoT..

Qualitative analysis, detailed in Tables 11 to 13, provides further insights: Table 11 reveals that E5-Mistral prioritizes semantic matching over theorem matching; Table 12 presents a failure case for E5-Mistral in identifying the correct algorithm; Table 13 illustrates a failure case for DIFFEMBED, where a noisy document receives a high score due to the relevance of its first half. Identifying incon-

	Leet.	TheoQ.
Overall Statistics		
Total number of queries	142	194
nDCG@10		
DIFFEMBED	30.0	48.3
E5-Mistral	28.7	26.1
Gold Passage Retrieval		
DIFFEMBED found at least one gold	74	124
E5-Mistral found at least one gold	73	64
Both found at least one same gold	53	56
Disjoint Gold Passage Found		
DIFFEMBED found E5-Mistral failed	19	67
E5-Mistral found DIFFEMBED failed	18	7

Table 8: Comparative analysis of top 10 documents retrieved by DIFFEMBED and E5-Mistral models on Leet. and TheoQ. subsets of BRIGHT.

	N	TheoT.	TheoQ.	AoPS	Leet.	Avg.
Llama3	10k	31.0	34.6	5.8	18.9	22.6
+ LLM2Vec	10k	28.3	33.8	12.6	23.6	24.6
+ SimCSE	10k	29.0	36.0	10.4	24.1	24.9
Mistral	10k	32.4	33.7	9.4	20.7	24.1
+ LLM2Vec	10k	30.8	30.4	9.1	22.9	23.3
+ SimCSE	10k	25.5	32.0	10.3	23.2	22.8
DIFFEMBED	10k	**38.9**	**48.3**	**15.4**	**30.0**	**33.2**

Table 7: Results on theorem-related reasoning-intensive retrieval (BRIGHT) for LLM-based embedding models trained with **REASONAUG** data. We report nDCG@10 for *TheoremQA* with with theorem retrieval (*TheoT.*) and question retrieval (*TheoQ.*), *AoPS*, and *LeetCode* (*Leet.*). SimCSE refers to the unsupervised contrastive learning with Wikipedia sentences (Gao et al., 2021). *N* is the amount of REASONAUG samples used for training. The best results with 10k training data are in **bold**.

sistencies within a single document presents a new challenge for embedding models.

E Noisy Data in BRIGHT

E.1 TheoQ.

We re-examined the gold annotations for the first ten test queries in the BRIGHT tasks. As shown in Table 9, TheoQ. exhibits a lower annotation error rate compared to Leet and AoPS. Given that TheoQ. and TheoT. originate from the same underlying dataset, we expect higher sample quality in both. Accordingly, our analysis places greater emphasis on TheoQ. and TheoT., where we conduct more detailed studies.

E.2 AoPS

Through manual evaluation of DIFFEMBED retrieved documents in the AoPS task, we observe a significant number of relevant documents that are not annotated as gold. In some queries, this number exceeds 10, undermining the effectiveness of nDCG@10 as a primary evaluation metric. This misalignment partially explains why most embedding-based retrieval models report low nDCG@10 scores (around 10%)—the actual performance is underestimated. Consequently, comparing models using this metric becomes unreliable. For such scenario with a large number of relevant documents in the corpus, we recommend to use the LLM-as-judge approach for evaluation like Siyue et al. (2024); Song et al. (2025).

E.3 Leet.

The Leet. subset often exhibits inconsistencies between query relevance and gold annotation. A notable example is the pair Trapping Rain Water I and Trapping Rain Water II, which are regarded as relevant to each other. However, they differ significantly:

- **Trapping Rain Water I** is a 1D problem solved with two pointers or a stack by tracking left and right boundaries.
- **Trapping Rain Water II** is a 2D problem requiring a min-heap and BFS to simulate water flow from the lowest boundaries inward.

Such cases illustrate a deviation from the intended task goal—retrieving problems and solutions that reflect the same underlying algorithmic design.

Additionally, we identify documents like Listing 4, where the problem statement and solution in the same document correspond to different tasks. In the situation like Table 13, documents may include a relevant question but an irrelevant solution, yet are not marked as relevant. According to the benchmark's definition, corpus documents should offer accurate reference demonstrations or tutorials, which is not always upheld.

	✓	×	Error Rate
TheoQ.	14	4	22%
AoPS	9	12	57%
Leet.	10	8	44%

Table 9: Human re-evaluation of gold evidence in BRIGHT subsets. For each subset (TheoQ., AoPS, Leet.), we manually inspect the gold documents associated with the first 10 test examples. A gold document is marked with a checkmark (✓) if it uses a theorem or algorithm similar to that in the query, and with a cross (×) otherwise. We report the proportion of incorrect gold documents as the error rate. Higher error rates are observed in AoPS and Leet. compared to TheoQ..

F Evaluating BRIGHT Using ReasonIR Data

While ReasonIR (Shao et al., 2025) is trained on 1.4M Public E5, 250k VL, and 100k HQ samples, we fine-tune using a subset of 100k samples from each dataset due to resource constraints. In Table 10, we compare DIFFEMBED with Llama3 and Qwen2.5 with LLM2Vec adaptation. Consistent with the findings in the main paper, DIFFEMBED achieves the highest performance, particularly on logic-intensive tasks such as TheoT. and TheoQ..

G REASONAUG Examples

We illustrate four retrieval examples:

- Math: Question-to-Theorem (Table 14)
- Math: Question-to-Question (Table 15)
- Physics: Question-to-Theorem (Table 16)
- Physics: Question-to-Question (Table 17)

	Bio.	Ear.	Eco.	Psy.	Rob.	Sta.	Sus.	Leet.	Pony	AoPS	TQ.	TT.	Avg.
E5-Mistral	18.8	26.0	15.5	15.8	16.4	9.8	18.5	28.7	4.8	7.1	26.1	26.8	17.9
ReasonIR	26.2	31.4	23.3	30.0	18.0	23.9	20.5	35.0	10.5	14.7	31.9	27.2	24.4
Llama3	7.5	28.2	18.1	24.5	**16.1**	**21.7**	17.0	37.0	**4.3**	6.3	26.2	21.7	19.1
Qwen2.5	12.5	**32.1**	**19.6**	20.1	14.2	19.4	13.7	40.6	1.8	**10.1**	36.5	28.8	20.8
DIFFEMBED	**14.7**	31.5	**19.6**	**25.9**	**16.1**	20.2	**17.2**	**41.3**	2.5	8.4	**39.6**	**32.6**	**22.5**

Table 10: Results on reasoning-intensive retrieval (BRIGHT) for embedding models trained using ReasonIR data. We report nDCG@10 for each subset task. We fine-tune the LLMs using LoRA instead of full parameter updates like in ReasonIR, leveraging 100k samples each from Public E5, ReasonIR VL, and ReasonIR HQ datasets.

Query Question: Mary is planning to bake exactly 10 cookies, and each cookie may be one of three different shapes – triangle, circle, and square. Mary wants the cookie shapes to be a diverse as possible. What is the smallest possible count for the most common shape across the ten cookies?

Gold Theorem: Pigeonhole principle

DIFFEMBED Retrieved Document:

In a group of 1000 people, at least how many people have to share the same birthday?

Using the **Pigeonhole Principle**, we can determine the minimum number of people who must share the same birthday. In this case, the "pigeons" are the 1000 people, and the "holes" are the 365 possible birthdays (ignoring leap years) ...

E5-Mistral Retrieved Document:

Let n represent the **smallest** integer that satisfies the following conditions:

$\frac{n}{2}$ is a perfect **square**.

$\frac{n}{3}$ is a perfect cube.

$\frac{n}{5}$ is a perfect fifth.

How many divisors does n have that are not multiples of **10**?

The first condition implies that the power of each prime factor of n must be an even power (excluding 2, which must be an odd power). The second condition implies that the power of each prime factor of n must be divisible by 3 ...

Table 11: Qualitative comparison on BRIGHT TheoremQA Questions: DIFFEMBED retrieves a question related to the gold theorem "Pigeonhole Principle", whereas the most relevant question retrieved by E5-Mistral is about "Least Common Multiple", highlighting that E5-Mistral tends to prioritize semantic matching than theorem matching.

```
def fallingSquares(positions):
    """
    There are several squares being dropped onto the X-axis of a 2D plane.

    You are given a 2D integer array ‘positions‘ where ‘positions[i] = [lefti,
    sideLengthi]‘ represents the ‘ith‘ square with a side length of ‘sideLengthi‘
    that is dropped with its left edge aligned with X-coordinate ‘lefti‘.

    Each square is dropped one at a time from a height above any landed squares. It
    then falls downward (negative Y direction) until it either lands **on the top
    side of another square** or **on the X-axis**. A square brushing the left/right
    side of another square does not count as landing on it. Once it lands, it
    freezes in place and cannot be moved.

    After each square is dropped, you must record the **height of the current
    tallest stack of squares**.

    Return _an integer array_ ‘ans‘ _where_ ‘ans[i]‘ _represents the height
    described above after dropping the_ ‘ith‘ _square_.

    **Example 1:**

    **Input:** positions = \[\[1,2\],\[2,3\],\[6,1\]\]
    **Output:** \[2,5,5\]
    **Explanation:**
    After the first drop, the tallest stack is square 1 with a height of 2.
    After the second drop, the tallest stack is squares 1 and 2 with a height of 5.
    After the third drop, the tallest stack is still squares 1 and 2 with a height
    of 5.
    Thus, we return an answer of \[2, 5, 5\].

    **Example 2:**

    **Input:** positions = \[\[100,100\],\[200,100\]\]
    **Output:** \[100,100\]
    **Explanation:**
    After the first drop, the tallest stack is square 1 with a height of 100.
    After the second drop, the tallest stack is either square 1 or square 2, both
    with heights of 100.
    Thus, we return an answer of \[100, 100\].
    Note that square 2 only brushes the right side of square 1, which does not count
     as landing on it.

    **Constraints:**

    *   ‘1 <= positions.length <= 1000‘
    *   ‘1 <= lefti <= 108‘
    *   ‘1 <= sideLengthi <= 106‘
    """

    ans = []
    intervals = []

    for p in positions:
        L, size = p
        R = L + size
        h = size
        for h2, R2 in intervals:
            if R2 > L and R > R2:
                h = max(h, size + h2)

        maxHeight = max((h2 for h2, R2 in intervals), default=0)
        ans.append(max(maxHeight, h))
        intervals.append((h, R))

    return ans
```

Listing 1: DIFFEMBED Retrieved Code Document from from BRIGHT

```
def countSmaller(nums):
    """
    Given an integer array ‘nums‘, return _an integer array_ ‘counts‘ _where_ ‘
    counts[i]‘ _is the number of smaller elements to the right of_ ‘nums[i]‘.

    **Example 1:**

    **Input:** nums = \[5,2,6,1\]
    **Output:** \[2,1,1,0\]
    **Explanation:**
    To the right of 5 there are **2** smaller elements (2 and 1).
    To the right of 2 there is only **1** smaller element (1).
    To the right of 6 there is **1** smaller element (1).
    To the right of 1 there is **0** smaller element.

    **Example 2:**

    **Input:** nums = \[-1\]
    **Output:** \[0\]

    **Example 3:**

    **Input:** nums = \[-1,-1\]
    **Output:** \[0,0\]

    **Constraints:**

    *   ‘1 <= nums.length <= 105‘
    *   ‘-104 <= nums[i] <= 104‘
    """

    def merge_sort(indices):
        if len(indices) <= 1:
            return indices
        mid = len(indices) // 2
        left = merge_sort(indices[:mid])
        right = merge_sort(indices[mid:])
        return merge(left, right)

    def merge(left, right):
        merged, count = [], 0
        while left and right:
            if nums[left[0]] <= nums[right[0]]:
                counts[left[0]] += count
                merged.append(left.pop(0))
            else:
                count += len(left)
                merged.append(right.pop(0))
        for i in left:
            counts[i] += count
        return merged + left + right

    counts = [0] * len(nums)
    merge_sort(list(range(len(nums))))
    return counts
```

Listing 2: E5-Mistral Retrieved Code Document from from BRIGHT

Query Question:

A city's skyline is the outer contour of the silhouette formed by all the buildings in that city when viewed from a distance. Given the locations and heights of all the buildings, return _the skyline formed by these buildings collectively_.

The geometric information of each building is given in the array `buildings` where `buildings[i] = [lefti, righti, heighti]`:

* `lefti` is the x coordinate of the left edge of the `ith` building.

* `righti` is the x coordinate of the right edge of the `ith` building.

* `heighti` is the height of the `ith` building.

You may assume all buildings are perfect rectangles grounded on an absolutely flat surface at height `0`. The skyline should be represented as a list of "key points" sorted by their x-coordinate in the form `[[x1,y1],[x2,y2],...]`. Each key point is the left endpoint of some horizontal segment in the skyline except the last point in the list, which always has a y-coordinate `0` and is used to mark the skyline's termination where the rightmost building ends. Any ground between the leftmost and rightmost buildings should be part of the skyline's contour.

Note: There must be no consecutive horizontal lines of equal height in the output skyline. For instance, `[...,[2 3],[4 5],[7 5],[11 5],[12 7],...]` is not acceptable; the three lines of height 5 should be merged into one in the final output as such: `[...,[2 3],[4 5],[12 7],...]`

...

Gold Theorem: Sweep Line Algorithm

DIFFEMBED Retrieved Document: Listing 1

E5-Mistral Retrieved Document: Listing 2

Table 12: Qualitative comparison on BRIGHT Leetcode: DIFFEMBED retrieves code related to the gold theorem "Sweep Line Algorithm" (see Listing 1), while the most relevant code retrieved by E5-Mistral pertains to"Merge Sort" (see Listing 2).

Query Question:

Convert a non-negative integer `num` to its English words representation.

Example 1:

Input: num = 123

Output: One Hundred Twenty Three

Example 2:

Input: num = 12345

Output: Twelve Thousand Three Hundred Forty Five

Example 3:

Input: num = 1234567

Output: One Million Two Hundred Thirty Four Thousand Five Hundred Sixty Seven

...

Gold Theorem: Recursive Decomposition and Mapping

E5-Mistral Retrieved Document: Listing 3

DIFFEMBED Retrieved Document: Listing 4

Table 13: Qualitative comparison on BRIGHT Leetcode: E5-Mistral retrieves the gold document (see Listing 3), which is closely related to "Mapping", though not directly about "Recursive Decomposition". In contrast, DIFFEMBED retrieves a non-gold document (see Listing 4) that includes a question requiring both "Recursive Decomposition" and "Mapping" but with an irrelevant solution code. All the embedding models in this work were not trained to assess the correctness of the solutions, which points out be a new dimension for data augmentation. We suspect these types of documents represent noisy data, as the corpus in BRIGHT's setup is intended to consist of reference documents, which are expected to be correct.

```
def int_to_roman(num):
    """
    Roman numerals are represented by seven different symbols: `I`, `V`, `X`, `L`, `
    C`, `D` and `M`.

    **Symbol**       **Value**
    I             1
    V             5
    X             10
    L             50
    C             100
    D             500
    M             1000

    For example, `2` is written as `II` in Roman numeral, just two one's added
    together. `12` is written as `XII`, which is simply `X + II`. The number `27` is
     written as `XXVII`, which is `XX + V + II`.

    Roman numerals are usually written largest to smallest from left to right.
    However, the numeral for four is not `IIII`. Instead, the number four is written
     as `IV`. Because the one is before the five we subtract it making four. The
    same principle applies to the number nine, which is written as `IX`. There are
    six instances where subtraction is used:

    *   `I` can be placed before `V` (5) and `X` (10) to make 4 and 9.
    *   `X` can be placed before `L` (50) and `C` (100) to make 40 and 90.
    *   `C` can be placed before `D` (500) and `M` (1000) to make 400 and 900.

    Given an integer, convert it to a roman numeral.

    **Example 1:**

    **Input:** num = 3
    **Output:**  "III "
    **Explanation:** 3 is represented as 3 ones.

    **Example 2:**

    **Input:** num = 58
    **Output:**  "LVIII "
    **Explanation:** L = 50, V = 5, III = 3.

    **Example 3:**

    **Input:** num = 1994
    **Output:**  "MCMXCIV "
    **Explanation:** M = 1000, CM = 900, XC = 90 and IV = 4.

    **Constraints:**

    *   `1 <= num <= 3999`
    """

        romans = [
            (1000, "M"), (900, "CM"), (500, "D"),
            (400, "CD"), (100, "C"), (90, "XC"),
            (50, "L"), (40, "XL"), (10, "X"),
            (9, "IX"), (5, "V"), (4, "IV"), (1, "I")
        ]
        roman = ""
        for value, symbol in romans:
            while num >= value:
                roman += symbol
                num -= value
        return roman
```

Listing 3: E5-Mistral Retrieved Code Document from BRIGHT

```
import heapq
"""
Given an array of ‘digits‘ which is sorted in **non-decreasing** order. You can
    write numbers using each ‘digits[i]‘ as many times as we want. For example, if ‘
    digits = [’1’,’3’,’5’]‘, we may write numbers such as ‘’13’‘, ‘’551’‘, and
    ‘’1351315’‘.

Return _the number of positive integers that can be generated_ that are less than or
     equal to a given integer ‘n‘.

**Example 1:**

**Input:** digits = \[ "1 ", "3 ", "5 ", "7 "\], n = 100
**Output:** 20
**Explanation:**
The 20 numbers that can be written are:
1, 3, 5, 7, 11, 13, 15, 17, 31, 33, 35, 37, 51, 53, 55, 57, 71, 73, 75, 77.

**Example 2:**

**Input:** digits = \[ "1 ", "4 ", "9 "\], n = 1000000000
**Output:** 29523
**Explanation:**
We can write 3 one digit numbers, 9 two digit numbers, 27 three digit numbers,
81 four digit numbers, 243 five digit numbers, 729 six digit numbers,
2187 seven digit numbers, 6561 eight digit numbers, and 19683 nine digit numbers.
In total, this is 29523 integers that can be written using the digits array.

**Example 3:**

**Input:** digits = \[ "7 "\], n = 8
**Output:** 1

**Constraints:**

*   ‘1 <= digits.length <= 9‘
*   ‘digits[i].length == 1‘
*   ‘digits[i]‘ is a digit from ‘’1’‘ to ‘’9’‘.
*   All the values in ‘digits‘ are **unique**.
*   ‘digits‘ is sorted in **non-decreasing** order.
*   ‘1 <= n <= 109‘
"""


def minRefuelStops(target: int, startFuel: int, stations: List[List[int]]) -> int:
    i, stops, curFuel = 0, 0, startFuel
    pq = []
    while curFuel < target:
        while i < len(stations) and stations[i][0] <= curFuel:
            heapq.heappush(pq, -stations[i][1])
            i += 1
        if not pq: return -1
        curFuel += -heapq.heappop(pq)
        stops += 1
    return stops
```

Listing 4: DIFFEMBED Retrieved Code Document from BRIGHT

Retrieval Task:

Given a math problem, retrieve the relevant math theorem that helps solve the given problem.

User Query:

In a garden shaped like a right triangle, one leg measures 24 meters and the other leg measures 10 meters. If a gardener wants to install a diagonal pathway that spans the triangle's hypotenuse, how many meters of paving stones will the gardener need? A) 20 m B) 30 m C) 26 m D) 34 m

Positive Document:

Pythagorean Theorem

In a right-angled triangle, the square of the length of the hypotenuse (the side opposite the right angle) is equal to the sum of the squares of the lengths of the other two sides. This relationship can be expressed mathematically as:

$$c^2 = a^2 + b^2$$

where c is the length of the hypotenuse, and a and b are the lengths of the other two sides.

Hard Negative Document:

Triangle Inequality

For any triangle with sides of lengths a, b, and c, the following inequalities hold:

$$a + b > c, \quad a + c > b, \quad \text{and} \quad b + c > a.$$

This means that the sum of the lengths of any two sides of a triangle must be greater than the length of the remaining side.

Table 14: REASONAUG Math Question-to-Concept Example.

Retrieval Task:

Given a math problem, retrieve the relevant problems that can be solved by the similar math theorem.

User Query:

A gardener is laying out a triangular flower bed in the backyard. The two sides of the triangle are 24 feet and 32 feet long. To ensure that the flower bed is perfectly shaped, the gardener wants to find the length of the fencing needed for the third side, which will form a right triangle at one of the corners. What is the length of the third side of the flower bed?

Positive Document:

A rescue team is stationed at a point 450 meters from the base of a cliff. To reach the top of the cliff, they need to scale it by climbing a rope that is secured at the top and stretches down to the base. The climber ascends along the rope, which makes an angle with the ground. If the height of the cliff is 600 meters, what is the length of the rope in meters that the climber must use?

We can visualize this situation as a right triangle, where one leg represents the height of the cliff, the other leg represents the horizontal distance from the base of the cliff to the team's position, and the hypotenuse represents the length of the rope.

Let r be the length of the rope. According to the relationship:

$$r^2 = (450)^2 + (600)^2$$

...

Taking the square root:

$$r = \sqrt{562500} = 750$$

Thus, the length of the rope the climber must use is $\boxed{750}$ meters.

Hard Negative Document:

A gardener is designing a rectangular flower bed that has an area of 120 square meters. He wants the length of the bed to be 5 meters more than twice the width. What will be the length of the flower bed in meters? A) 15 B) 20 C) 25 D) 30

Let the width of the flower bed be w meters. Then, the length l will be $2w + 5$ meters. The area of the rectangle can be expressed as:

$$l \times w = 120$$

Substituting for l:

$$(2w + 5)w = 120$$

This expands to:

$$2w^2 + 5w - 120 = 0$$

Now, we can identify the coefficients: $a = 2$, $b = 5$, and $c = -120$. Using the quadratic formula $w = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$:

$$b^2 - 4ac = 5^2 - 4 \cdot 2 \cdot (-120) = 25 + 960 = 985$$

Thus,

$$w = \frac{-5 \pm \sqrt{985}}{4}$$

...

Thus, the correct length is $\boxed{20}$.

Table 15: REASONAUG Math Question-to-Question Example.

Retrieval Task:

Given a physics problem, retrieve the relevant physics theorem that help solve the given problem.

User Query:

A laboratory experiment involves a cylindrical conductor made of an unknown material that has a length of 2.5 meters and a diameter of 0.01 meters. This conductor is connected to a circuit powered by a stable voltage source of 12 volts. During an experiment, the current flowing through the conductor is measured to be 0.5 amperes. Calculate the resistance of the conductor in ohms, given that the resistivity of the material is not known but can be derived from the current and voltage provided.

Positive Document:

Ohm's Law

Ohm's Law states that the current flowing through a conductor between two points is directly proportional to the voltage across the two points and inversely proportional to the resistance of the conductor. This relationship is fundamental in electrical circuits and can be expressed mathematically as:

$$V = I \cdot R$$

where V is the voltage (in volts), I is the current (in amperes), and R is the resistance (in ohms).

Hard Negative Document:

Energy Stored in a Capacitor Formula

The energy U stored in a capacitor is given by the formula:

$$U = \frac{1}{2}CV^2$$

where U is the energy (in joules), C is the capacitance (in farads), and V is the voltage across the capacitor (in volts). This equation describes how the energy stored in the electric field of a capacitor is related to its capacitance and the voltage applied across it.

Table 16: REASONAUG Physics Question-to-Concept Example.

Retrieval Task:

Given a physics problem, retrieve the problems that can be solved by the similar physics theorem.

User Query:

A circuit consists of a 12V battery connected in series with two resistors: R1 and R2. R1 has a resistance of 4 ohms, and R2 has a resistance of 6 ohms. Additionally, there is a variable resistor (R3) connected in parallel with R2, which can vary its resistance. When R3 is set to 3 ohms, the total current in the circuit is measured to be 1.2A. If R3 is adjusted to 8 ohms, what is the new total current flowing through the circuit?

Positive Document:

A 10-meter-long copper wire has a uniform cross-sectional area of 1.5 mm² and is used to connect a 9V battery to a small device. The resistivity of copper is $1.68 \times 10^{-8}\,\Omega \cdot m$. What is the current flowing through the device if the total resistance of the wire is considered, and the device has an internal resistance of 2 ohms?

First, we calculate the resistance of the copper wire using the formula:

$$R_{\text{wire}} = \frac{\rho L}{A}$$

where ρ is resistivity, L is length of the wire, and A is cross-sectional area.

...

Now use Ohm's law to find the current flowing through the device:

$$I = \frac{V}{R_{\text{total}}}$$

Substituting the values:

$$I = \frac{9V}{2.112\,\Omega} \approx 4.26\,A$$

Thus, the current flowing through the device is approximately $4.26\,A$.

Hard Negative Document:

An engineer is testing a prototype of a magnetic levitation train system. As the train approaches a section with a solenoid that is turned off, its speed is 15 m/s. The solenoid has an area of 0.8 m² and the magnetic field it produces, when activated, is 0.4 T. The train is designed so that when it approaches, an induced current is created in the conductive loops on the train due to a change in magnetic flux. The solenoid is activated precisely as the train is 2 m away from its entrance. Calculate the induced current (I) in the loop of the train if the resistance of the loop is 5 ohms. Treat the magnetic field as uniform.

First, calculate the time t it takes for the train to reach the solenoid:

$$t = \frac{d}{v} = \frac{2\,\text{m}}{15\,\text{m/s}} = \frac{2}{15}\,\text{s} \approx 0.1333\,\text{s}$$

Now, calculate the change in magnetic flux $\Delta\Phi$ through the loops on the train as the solenoid is activated:

...

Table 17: REASONAUG Physics Question-to-Question Example.