PseudoSeer: a Search Engine for Pseudocode

Levent Toksoz
Computer Science and Engineering
The Pennsylvania State University
lkt5297@psu.edu

Mukund Srinath*
Amazon.com, Inc.
Seattle, WA, USA
mus824@psu.edu

Gang Tan
Computer Science and Engineering
The Pennsylvania State University
gtan@psu.edu

C. Lee Giles
Information Sciences and Technology
The Pennsylvania State University
clg20@psu.edu

Abstract

PseudoSeer is a novel search engine for academic pseudocode, enabling retrieval over 320,000 algorithm implementations extracted from the arXiv. Using the system's caption-reference pairs, we study asymmetric retrieval, matching short queries with a median length of five words against long documents of roughly 300 words composed primarily of natural language with limited LaTeX notation. Our evaluation reveals scaling limitations in embedding models: a 149M parameter encoder outperforms 1.5B parameter alternatives, while BM25 remains competitive with pretrained models. Analyzing attention patterns over 33,000 caption document pairs, we identify two factors driving these results: attention efficiency and attention concentration. Models that significantly attend to sinks or non-discriminative tokens leave less attention for discriminative content, while models with overly diffuse attention fail to form discriminative representations. Guided by these findings, PseudoSeer's embedding model, trained via contrastive learning with efficient attention patterns, outperforms the best pretrained model by 8.7 points. A hybrid approach combining learned embeddings with BM25 reaches 66.5% R@10. PseudoSeer is deployed at pseudoseer.ist.psu.edu as both a practical search system and a benchmark for retrieval evaluation.

1 Introduction

The ever growing volume of academic literature necessitates efficient search tools that cater to the specific needs of researchers. In many disciplines, pseudocode plays a vital role in communicating algorithms, describing computational procedures, and promoting reproducibility. Pseudocode serves as a critical bridge between natural language algorithm descriptions and executable implementations, yet no existing search infrastructure enables researchers to efficiently locate pseudocode within the millions of papers published annually. This gap motivated our development of PseudoSeer, a search engine that indexes over 320,000 pseudocode samples extracted from 2.2 million arXiv papers.

We introduce PseudoSeer, a search engine for academic pseudocode that indexes over 320,000 algorithm implementations extracted from arXiv papers spanning three decades. Beyond its utility as a practical tool, PseudoSeer also provides an evaluation framework for studying retrieval over scientific documents, which pose an asymmetric matching problem: short natural language queries with a median length of five words must be matched against substantially longer algorithm descriptions of roughly 300 words that mix prose with limited LaTeX notation. Individual caption words typically appear within their corresponding documents, rewarding term matching while still requiring semantic understanding to identify the correct passage. This setting exposes surprising limitations in current embedding models. Evaluating a range of architectures, including both encoder only and decoder only models spanning 22M to 1.5B parameters, we find that scaling provides no clear benefit: a 149M parameter model (GTE-ModernBERT) outperforms 1.5B parameter alternatives, while BM25 remains competitive with all pretrained embedding models. This diverges from general expectations, under which larger models typically dominate.

To understand these patterns, we analyze attention distributions across 33,000 caption document pairs and identify two factors that explain model performance. The first is *attention efficiency*: whether models direct attention to content or divert it to sinks and non-discriminative tokens. The second is *attention concentration*: whether attention to content is focused on discriminative terms or diffused across the document. These factors interact in revealing ways: Snowflake (568M) directs 77% of attention to natural language yet achieves

*Work done while at The Pennsylvania State University.

Findings of the Association for Computational Linguistics: ACL 2026, pages 39706–39716

lower recall than ModernBERT (149M), which allocates only 47% but concentrates it on discriminative terms. Qwen2 (1.5B) shows decent concentration (Gini 0.74) but diverts substantial attention to sinks and non-discriminative tokens.

Guided by these findings, we develop PseudoSeer's embedding model through contrastive learning, initializing from GTE-ModernBERT based on its efficient attention patterns. This reaches 64.9% Recall@10, outperforming the best pretrained model by 8.7 points. Hybrid retrieval combining learned embeddings with BM25 reaches 66.5% R@10. Our contributions are as follows:

- We release PseudoSeer, the first search engine for academic pseudocode, deployed at pseudoseer.ist.psu.edu, serving as both a practical tool and an evaluation framework for scientific pseudocode retrieval.
- We provide a systematic attention analysis identifying efficiency and concentration as key factors explaining why smaller models outperform larger alternatives on structured scientific content.
- We demonstrate that initializing from models with efficient attention patterns enables effective domain adaptation, achieving state-of-the-art performance through contrastive learning.

2 Related Work

Several pseudocode datasets have been developed to support research in various code related translation and generation tasks. Notably, the SPOC dataset Kulal et al. (2019) contains around 20,000 programs with human-authored pseudocodes and test cases, designed for translating pseudocode to C++ code. Another significant dataset is by Oda et al. (2015), which includes approximately 16,000 manually crafted pseudocodes. The dataset is used for statistical machine translation. Additionally, Zavershynskyi et al. (2018) provide a dataset of about 2,000 manually written pseudocodes for program synthesis tasks.

While these datasets are valuable for specific tasks, they represent a limited variety of pseudocodes and are hand-crafted by programmers. Moreover, their relatively small size and lack of diversity make them less suitable for use in search engines, which require more extensive and varied datasets to handle a broad range of pseudocode search queries effectively. To that end, Toksoz et al. (2024) provide a much larger dataset containing 320,000 pseudocode samples extracted from scholarly papers on arXiv, which will serve as the foundation for the dataset we use to build our pseudocode search engine.

Recent advances in retrieval have been largely driven by Sentence Transformers by Reimers and Gurevych (2019), which learn dense vector representations of natural language sentences optimized for similarity-based tasks. These models have demonstrated strong performance in semantic search by mapping semantically similar inputs to nearby points in the embedding space, making them well-suited for our task of retrieving pseudocode from natural language descriptions. This shift toward neural retrieval aligns with broader trends in integrating large language models (LLMs) into search infrastructures, as surveyed by Xiong et al. (2024) and Xu et al. (2025), which highlight both the potential and challenges of combining semantic understanding with efficient retrieval services. In our experiments, we evaluate a diverse set of Sentence Transformer models, including Qwen2-1.5B-instruct (Yang et al., 2024), Stella-en-1.5B-v5 (Zhang et al., 2025), Snowflake-arctic-embed-l-v2.0 (Yu et al., 2024), GTE Modern BERT (Zhang et al., 2024; Li et al., 2023), bge-m3 (Chen et al., 2024), e5-large-v2 (Wang et al., 2022), allenai-specter (Cohan et al., 2020), all-MiniLM-L6-v2 (Wang et al., 2020), and all-mpnet-base-v2 (Song et al., 2020).

Recent work has examined how transformers allocate attention. Clark et al. (2019) analyzed BERT's attention heads, finding patterns such as attending to delimiter tokens and positional offsets, with certain heads corresponding to syntactic relations. Kovaleva et al. (2019) showed that a limited set of attention patterns repeat across heads, suggesting overparameterization. Voita et al. (2019) demonstrated that specialized heads play linguistically-interpretable roles and are last to be pruned when removing redundant heads. More recently, Xiao et al. (2024) identified attention sinks, tokens that disproportionately attract attention regardless of semantic importance, and showed that preserving these tokens enables stable streaming inference. Our work extends this line of analysis to retrieval encoders, distinguishing between active structural processing and sink behavior, and connecting attention patterns to retrieval performance on scientific documents.

Search engines for structured scientific content

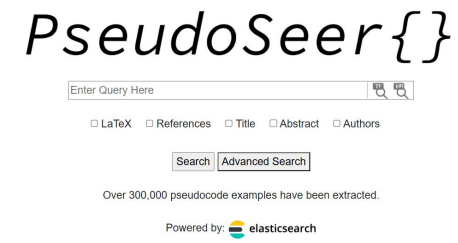

Figure 1: Landing page of the search engine with field-specific toggles for LaTeX, references, title, abstract, and authors. Supports symbolic (BM25), semantic (Transformer-based), and hybrid (RRF) retrieval.

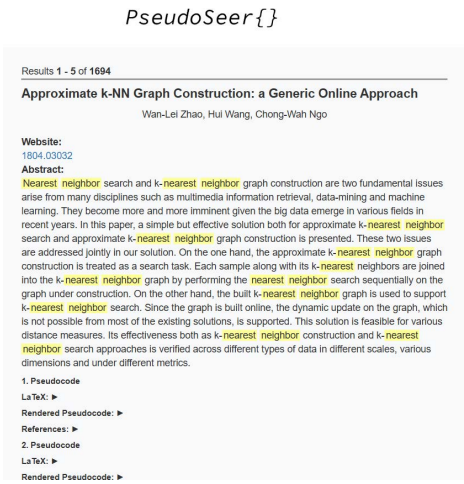

Figure 2: Results page when searching for `nearest neighbor` in the abstract field.

have made significant strides in specialized domains. MathWebSearch (Kohlhase and Şucan, 2006; Kohlhase et al., 2012) pioneered structural indexing of mathematical expressions, with subsequent work on normalization techniques for detecting logically equivalent formulas (Normann and Kohlhase, 2007). Tools such as GROBID (Lopez, 2008–2023) and Nougat (Blecher et al., 2023) have advanced scientific document processing and metadata extraction. Our work complements these efforts by targeting a different modality: matching natural language queries against documents containing pseudocode with mixed prose and LaTeX notation, where semantic understanding and lexical matching work together.

3 Pseudoseer System

3.1 Data Collection

The pseudocode search engine is built on data from Toksoz et al. (2024), extracted from LaTeX source files of arXiv papers from 1991 to June 2023. Pseudocode blocks were identified by locating `\begin{algorithm}` and `\end{algorithm}` tags, resulting in nearly 320,000 pseudocode examples across 2.2 million papers. Each sample is enriched with metadata including its arXiv ID, publication year, arXiv weblinks, and reference snippets, which are text snippets that mention the extracted pseudocode.

Reference snippets are central to the retrieval task. These snippets are extracted by aligning `\ref` tags with a surrounding text window of up to 1200 characters on each side, trimmed to sentence boundaries when detected (Toksoz et al., 2024), and reflect how researchers naturally describe algorithms within scientific papers. While we index all available fields, our retrieval, evaluation, and fine tuning procedures are primarily centered on reference snippets. This design reflects a system wide choice, as pseudocode blocks are dominated by LaTeX markup, mathematical symbols, and variable definitions that diverge substantially from the structure of natural language queries. Reference snippets provide a middle ground: they are primarily natural language, with limited LaTeX notation. Some pseudocode entries were excluded due to missing or unresolvable reference tags. To the best of our knowledge, no prior work has focused specifically on extracting and enabling semantic search over pseudocode content in scientific papers, making this framework a novel and valuable resource for information retrieval.

3.2 Search Architecture

PseudoSeer supports sparse, dense, and hybrid retrieval over indexed paper fields (Figure 1).

Sparse Retrieval. We use Elasticsearch with standard Unicode tokenization to index all extracted fields: title, abstract, authors, pseudocode blocks, and reference snippets. BM25 serves as the sparse retrieval baseline. For multi field queries, we apply field specific weights (2.0 for title and abstract, 1.0 for others), with reference snippets receiving elevated weight to reflect their alignment with natural language queries.

Dense Retrieval. For semantic search, we embed each reference snippet independently using PseudoSeer's embedding model, a Sentence Transformer trained on caption reference pairs (Section 6). We adopt per snippet embeddings rather than concatenating all snippets into a single vector,

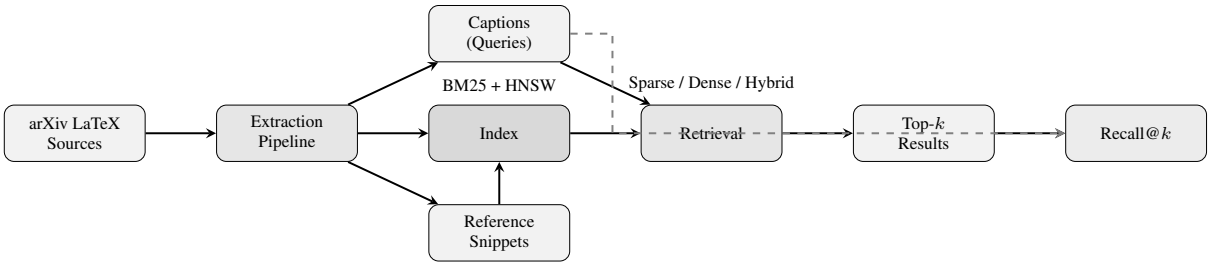

Figure 3: System overview. Captions and reference snippets are extracted from arXiv LaTeX sources. Reference snippets are indexed using BM25 and HNSW. Captions act as evaluation queries; Recall@k measures whether the correct reference appears among the top-k retrieved results. Dashed arrows indicate ground-truth matching.

as longer inputs exceed typical Transformer context limits and tend to degrade retrieval quality. Embeddings are 768-dimensional and L2-normalized for cosine similarity, indexed via Elasticsearch's HNSW based k-NN plugin for efficient approximate nearest neighbor search over 320,000 snippets. At query time, we embed the query and retrieve the top k matching snippets. Since a single paper may contain multiple reference snippets, we collapse results to the document level by taking the maximum similarity score across all snippets belonging to the same paper. This document level ranking aligns retrieval with how users interact with results, each returned entry represents a complete paper with its pseudocode, metadata, and reference snippets (Figure 2).

Hybrid Retrieval. We combine sparse and dense signals using Reciprocal Rank Fusion (RRF) (Cormack et al., 2009). Both BM25 and the Transformer model retrieve over reference snippets independently, and their rankings are fused. This mode achieves the highest retrieval quality at the cost of increased latency, and allows us to evaluate the complementarity of lexical and semantic signals.

Ranking. For sparse retrieval, BM25 scores documents by term frequency with saturation and document length normalization. For dense retrieval, we rank by cosine similarity between the query embedding and snippet embeddings, selecting the maximum score per document. For hybrid retrieval, we apply Reciprocal Rank Fusion (Cormack et al., 2009), which combines ranked lists by summing reciprocal ranks.

3.3 Caption-Based Evaluation

In our evaluation, we use captions of pseudocode as evaluation queries. The benefit of this design is to avoid manual ground-truth annotation, since the reference snippets for some pseudocode are automatically treated as the ground truth label for the caption of the pseudocode. Caption reference pairs are structurally co-located within the same paper, and captions, which are normalized to natural language with a median length of five words, resemble the short keyword style queries commonly used in search engines (Jansen et al., 2000). Captions of three words or fewer were filtered as they tend to be generic and lack discriminative content.

The document side, reference snippets with occasional LaTeX notation, represents our target domain of scientific literature search. Crucially, captions are lexically distinct from their corresponding reference snippets, and authors do not usually copy caption text into the paper body, so there is no guaranteed phrase level overlap between query and target. However, individual caption words typically appear somewhere in the reference, rewarding term matching while requiring semantic understanding to identify the correct passage. This evaluation yields a retrieval setting that is neither purely NL to NL nor subject to the extreme modality gap of NL to LaTeX, but instead an asymmetric task involving short queries matched against longer, structure rich documents. This setting is described in Section 5.

Given this setup, for each caption, we retrieve the top k snippets and check whether any snippet from the same paper appears in the results. We use Recall@k as our primary metric, measuring the proportion of queries for which the correct document is retrieved within the top k results. We additionally report nDCG@k (normalized discounted cumulative gain), which applies a logarithmic position discount to reward relevant documents retrieved at higher ranks; since each query has exactly one relevant document, alternative rank-based metrics such as MRR yield similar orderings over models. To prevent data contamination, we split caption reference pairs into train, validation, and test sets before any model training. Figure 3 illus-

Table 1: Retrieval performance across models on the full corpus. Sorted by Recall@10. Arch.: Enc=Encoder, Dec=Decoder, Lex=Lexical.

Model	Params	Arch.	R@10 (%)	nDCG
BM25	—	Lex	51.4	0.461
GTE-ModernBERT	149M	Enc	48.2	0.432
Snowflake-arctic-l-v2.0	568M	Enc	46.8	0.405
Qwen2-1.5B-instruct	1,540M	Dec	46.6	0.416
Stella-en-1.5B-v5	1,540M	Dec	46.4	0.411
bge-m3	569M	Enc	41.1	0.349
e5-large-v2	335M	Enc	35.6	0.286
all-mpnet-base-v2	109M	Enc	30.3	0.241
all-MiniLM-L6-v2	22.7M	Enc	25.6	0.198
allenai-specter	110M	Enc	13.8	0.101

trates the complete pipeline from data extraction through evaluation.

4 Evaluation Framework

We evaluate a range of pretrained embedding models spanning two orders of magnitude in parameter count, from 22.7M up to 1.5B parameters. As PseudoSeer is a deployed search system, model selection must balance retrieval quality against inference latency and resource constraints. Table 1 summarizes the models evaluated, including both encoder based architectures (BERT-style bidirectional attention) and decoder based architectures (causal attention adapted for embedding). We include BM25 as a lexical baseline.

For each model, we encode all reference snippets in the corpus and retrieve the top k results by cosine similarity to the encoded caption query. All models are evaluated using their default settings with a maximum input length of 512 tokens. Our dataset contains approximately 33,000 caption reference pairs. Crucially, the retrieval corpus includes all reference snippets, not just those with associated captions. This setting reflects real world use, where queries must match against the full corpus. For BM25, Elasticsearch's default implementation is used. Recall@10 and nDCG are reported.

4.1 Scaling and Performance Relationship

Table 1 presents retrieval performance for all models. The results reveal several noteworthy patterns. First, BM25 achieves the highest performance among all pretrained methods, reaching 51.4% compared to 48.2% for the best embedding model (GTE-ModernBERT), a notable result given the dominance of embedding methods in modern retrieval systems and the growing adoption of decoder based models in RAG pipelines. Second, model size shows no clear correlation with performance. The largest models do not achieve the highest scores: Qwen2 (1.5B parameters) reaches 46.6%, while GTE-ModernBERT (149M) achieves 48.2%, a model with 10× fewer parameters performs 1.6 points higher. Even Snowflake (568M) matches Qwen2 despite having one third of the parameters. More broadly, the best performing embedding model (GTE-ModernBERT at 149M) is smaller than half of the models evaluated. Third, decoder based models (Qwen2, Stella) perform below encoder based alternatives of smaller size. These patterns differ from typical expectations in general purpose retrieval, where larger models tend to outperform smaller ones. On our evaluation framework, the relationship between scale and performance is less clear, and lexical matching remains highly competitive. The following section investigates potential explanations for these observations.

5 Attention Analysis

Attention patterns are analyzed for the four highest performing models across 33,000 caption document pairs for two factors: placement, how attention is distributed across token categories, and concentration, how attention is distributed within natural language tokens. Attention to caption words is also tracked because our dataset uses content based natural language captions that describe the semantic content of each pseudocode document. These captions contain terms that also appear in the documents they describe, with an average of 3.3 cap-

Table 2: Attention Placement by Token Category

Model	NL %	Structural %	LaTeX %	Special %	Stopwords %
ModernBERT	47.1±7.1	4.6±3.7	**18.9±5.0**	20.3±2.2	2.0±1.3
Snowflake	**77.2±4.1**	7.4±2.1	6.4±2.8	2.2±1.7	6.5±3.1
Qwen	47.9±7.5	15.1±4.6	7.2±3.6	16.8±4.0	6.2±2.8
Stella	14.7±2.2	**16.8±5.7**	9.4±2.7	**42.6±2.2**	**11.1±5.8**

Table 3: Attention Redistribution When LaTeX Removed

Model	Category	Original	Cleaned
ModernBERT	NL	47.1%	**55.8%**
	Punctuation	2.9%	11.9%
	Newlines	0.9%	1.2%
	Stopwords	2.0%	2.4%
Qwen	NL	47.9%	49.2%
	Punctuation	3.3%	4.8%
	Newlines	11.4%	**14.2%**
	Stopwords	6.2%	6.8%

tion words per document. For example, a caption such as "Two-neighborhood Iterated Local Search" includes discriminative terms ("two," "neighborhood," "iterated," "local," "search") that appear in the corresponding reference snippet and carry semantic meaning relevant to retrieval. Caption attention therefore serves as a direct measure of retrieval focus. A model that captures document semantics should attend to words that describe its content. Gini coefficient and entropy measure whether attention is peaked or diffuse across content words in general, caption attention evaluates whether that concentration targets query relevant terms.

We categorize tokens into five groups: *natural language* (content words including nouns, verbs, and domain-specific terms), *structural* (punctuation and newlines), *LaTeX* (commands, environments, and variables), *special* (CLS, SEP, end-of-text markers), and *stopwords* (function words). We separate LaTeX from structural tokens because LaTeX includes variable names that have semantic content. Models also differ in whether LaTeX attention reflects active processing or serves as an attention sink, tokens that absorb excess attention without contributing to the representation (Xiao et al., 2024).

The models we analyze use different pooling strategies to produce the final embedding vector for retrieval, which determines how attention weights are extracted. GTE-ModernBERT and Snowflake-Arctic use CLS token pooling: the CLS token representation serves as the document embedding, so attention is extracted from the CLS token to all other positions. GTE-Qwen uses last token pooling, where the final token representation serves as the embedding; attention is therefore extracted from the last token position. Stella uses mean pooling, averaging all token representations to produce the embedding; since every position contributes equally, attention patterns are averaged across all sequence positions. For all models, we analyze attention at the final layer and aggregate subword tokens to the word level using the tokenizer's builtin word mapping.

5.1 Placement and Concentration

Table 2 shows attention allocation across token categories. We analyze these patterns through the lens of attention efficiency: whether attention actively contributes to the representation or serves as an attention sink. We identify sink behavior through two analyses: (1) layer by layer attention to non-content tokens (stopwords, structural, special, LaTeX) in original text to identify persistent patterns, and (2) attention redistribution when LaTeX is removed to see whether freed attention goes to NL or reroutes to other sinks (Table 3).

Models exhibit distinct efficiency patterns. Snowflake allocates minimal attention to structural tokens (7.4%) and maximal attention to NL (77.2%) consistently across layers, suggesting efficient allocation with no apparent sink behavior. ModernBERT shows selective structural processing: LaTeX attention spikes only in specialized layers (L17-18: 61-65%) while remaining low elsewhere (2-11%), and other structural tokens (punctuation, newlines) stay consistently low throughout (<3%). When LaTeX is removed, +8.7% redirects to NL, consistent with active processing rather than sink behavior. In contrast, Qwen has persistent non-content attention: newline attention is high

Table 4: Non-Content Attention Across Layers

Model	Token Type	Early	Middle	Late	Pattern
ModernBERT	LaTeX	2-6%	9-11%	**61-65%*** $\rightarrow$ 19%	Localized spike
ModernBERT	Punct/Newline	<3%	<3%	<3%	Consistently low
Qwen	Newlines	8-30%	**46-55%**	11-29%	Persistent (min 11%)
Qwen	LaTeX/Punct	6-8%	6-8%	6-8%	Consistently low
Snowflake	All structural	<5%	<5%	<5%	Consistently low
Stella	Punctuation	8-10%	15-28%	27-36%	Persistent (min 8%) with fluctuations

*L17-18 only; returns to 19% at final layer

Table 5: Attention Concentration Metrics

Model	NL % Total	NL Gini↑	NL Entropy↓	Caption %	Caption Gini↑	Caption Entropy↓
ModernBERT	47.1±7.1%	**0.77±0.05**	**0.75±0.06**	**6.1±5.4**	**0.45±0.26**	**0.59±0.31**
Qwen	47.9±7.5%	0.74±0.05	0.77±0.06	5.7±5.1	0.39±0.23	0.65±0.32
Snowflake	**77.2±4.1%**	0.57±0.04	0.87±0.02	4.4±4.4	0.25±0.20	0.62±0.40
Stella	14.7± 2.2%	0.43±0.03	0.93±0.02	1.9±2.6	0.29±0.19	0.72±0.34

from early layers (L6-10: 46-55%) and never drops below 11% in any layer (Table 4). When LaTeX is removed, attention shifts to newlines (+2.8%) rather than NL (+1.3%), suggesting sink behavior where attention reroutes to alternative non-content tokens. Stella also exhibits persistent non-content attention (8%) with high variance (±14.8%), suggesting document-dependent sink patterns.

However, avoiding sink behavior does not guarantee effective retrieval. Table 5 reveals that Snowflake, despite efficient attention allocation (77.2% to NL, with no apparent sink behavior), shows low concentration (Gini 0.57) and achieves lower caption attention (4.4%) than ModernBERT (6.1%). For a 5-word query matched against $\sim$300 content words, identifying discriminative terms requires concentrated attention. ModernBERT's high Gini (0.77) indicates sharp peaks on specific words; Stella's low Gini (0.43) indicates more uniform distribution. NL and caption entropy values corroborate this pattern. While Stella shows slightly higher average caption Gini than Snowflake (0.29 vs 0.25), Qwen shows a slightly larger difference over Snowflake (caption Gini 0.39 vs 0.25), which may partially explain its slightly higher nDCG (0.416 vs 0.405) despite slightly lower Recall@10 (46.6% vs 46.8%), as concentrated attention on discriminative terms can improve ranking quality even when retrieval rates are similar. However, the differences in Recall@10 and nDCG among these three models are small; further investigation with additional evaluations would be helpful to draw stronger conclusions.

ModernBERT also develops specialized content focused layers that may contribute to its concentration advantage. At layers L13, L16, L19, and L22, ModernBERT shows elevated NL attention (31%, 46%, 52%, and 47% respectively) and caption attention (3.7%, 5.4%, 6.2%, and 6.1%), substantially higher than surrounding layers and other models at comparable depths. In contrast, Snowflake maintains uniformly high NL attention (68-91%) across all layers but never exceeds 4.9% caption attention at any layer. Qwen only reaches comparable caption attention (4.4-5.7%) in its final layers (L21-28), after recovering from middle-layer sink dominance. Stella's caption attention peaks mid-network (L15-22: 4.1-5.1%) then drops sharply at the final layer (L28: 1.9%), potentially due to mean pooling diluting concentrated signal across all positions.

The two factors, attention efficiency and concentration, interact in revealing ways. Snowflake and Qwen achieve nearly identical recall (46.8% vs 46.6%) through different limitations: Snowflake directs attention to NL but distributes it diffusely; Qwen concentrates attention better (Gini 0.74) but wastes capacity on sinks. ModernBERT achieves both efficient allocation and high concentration.

Table 6: BM25 vs Embedding Models by Query Length in Words. Emb columns show average and maximum over ModernBERT, Qwen, Snowflake, and Stella.

	R@10				nDCG			
Length	BM25	Emb(avg)	Emb(max)	Gap	BM25	Emb(avg)	Emb(max)	Gap
4–5	0.508	0.462	0.486	4.6%	0.447	0.401	0.435	4.6%
6–7	0.521	0.483	0.496	3.8%	0.477	0.431	0.451	4.5%
8–10	0.538	0.507	0.523	3.0%	0.506	0.465	0.490	4.1%
11–15	0.547	0.523	**0.636**	2.4%	0.518	0.484	**0.620**	3.5%

Table 7: Retrieval Performance on the held-out test set.

Method	**R@10**	**nDCG**
BM25	59.4%	0.519
GTE-ModernBERT (pretrained)	56.2%	0.488
PseudoSeer Embedding	64.9%	0.593
PseudoSeer Hybrid	**66.5%**	**0.610**

5.2 Embedding Models vs BM25

BM25 achieves 51.4% R@10, outperforming all embedding models on average. This result reflects our dataset characteristics: queries are short (median 5 words) while documents are long (median 292 words), and caption words appear frequently in reference snippets. BM25 exploits this through IDF weighting, which concentrates weight on discriminative query terms while downweighting common words. With short queries, there is limited semantic context for embedding models to leverage, and they must learn comparable concentration, which proves difficult when balancing attention to query relevant content against other demands: architectural overhead (e.g., CLS/SEP tokens receive ∼20% attention in GTE-ModernBERT), attention sinks (e.g., Qwen allocates up to 30% to structural, LaTeX, and stopword tokens), and active structural processing (e.g., GTE-ModernBERT's 18% LaTeX attention for document understanding).

As queries lengthen, embedding models close the gap (Table 6). BM25's advantage decreases from 4.6% at 4–5 words to 2.4% at 11–15 words, and the best embedding model exceeds BM25 at longer queries (0.636 vs 0.547). Longer queries provide richer semantic context that embeddings can exploit, while BM25 remains limited to exact term matching. A broader window for 11-15 words ensures comparable sample sizes to other windows, as longer queries are less frequent in the dataset.

6 Pseudoseer's Retrieval Model

Our attention analysis reveals that GTE-ModernBERT achieves both attention efficiency and high concentration, yielding the highest caption attention (6.1%) among pretrained models. These properties, combined with its compact size (149M parameters), make it an ideal foundation for building PseudoSeer's retrieval model.

Model Development. PseudoSeer's embedding model is developed through contrastive learning on caption reference pairs. Using GTE-ModernBERT as initialization, we train on caption reference pairs where positives are captions matched with their corresponding reference snippets, with in-batch negatives providing contrastive signal. To prevent data contamination, the dataset is split into train, validation, and test sets; the pretrained model evaluation in Section 4 used the full corpus since no training was involved. Training is for 10 epochs batch size 32, learning rate 1×10^{-5}, and 100 warmup steps.

Table 7 shows results on the held out test set. PseudoSeer's embedding model achieves 64.9% R@10, an 8.7 point improvement over the pretrained baseline and 5.5 points above BM25. BM25 is known to be a strong baseline that often outperforms neural retrievers in zero shot settings (Thakur et al., 2021); surpassing it with a dense retriever demonstrates that starting from a model with efficient attention patterns enables effective domain adaptation to structured scientific content.

Hybrid Retrieval. Section 5.2 showed that BM25 and embedding models have complementary strengths: BM25 captures exact term matches through IDF weighting, while embeddings capture semantic similarity. PseudoSeer combines these using Reciprocal Rank Fusion (RRF), running BM25 and our embedding model in parallel and merging rankings with $k = 60$. PseudoSeer Hybrid achieves 66.5% R@10, the best overall per-

formance, gaining 1.6 points over embedding alone through BM25's complementary exact matching.

Latency. While retrieval effectiveness is critical, latency matters for deployment. BM25 averages approximately 47ms, while PseudoSeer's embedding model requires ∼106ms due to real time query encoding, both acceptable for interactive use. Hybrid is slower and takes about a second, executing both pipelines before merging results, but achieves the highest recall. These measurements were obtained on a 2nd Gen AMD EPYC CPU with 64 cores. PseudoSeer offers all three options: users prioritizing speed with filtering capabilities can use BM25, those seeking better recall with minimal latency increase can use embedding, and those prioritizing recall over latency can use hybrid. Note that since pseudocode LaTeX is rendered in real time and cached, user perceived wait time may exceed query latency if images are not yet cached.

7 Conclusion

We presented PseudoSeer, a novel search engine for academic pseudocode that serves as both a practical tool for researchers and an evaluation framework for studying embedding model behavior on scientific documents. Our experiments reveal that the relationship between model scale and retrieval performance is complex: BM25 outperforms all pretrained embedding models, and a 149M parameter encoder surpasses models with 1.5B parameters.

Through attention analysis, we identify two factors that explain these results. First, attention efficiency: models that allocate substantial attention to sinks and non-discriminative tokens have reduced capacity for content processing. Second, attention concentration: models that efficiently direct attention to content but distribute it diffusely struggle to create discriminative representations for retrieval. Our analysis distinguishes between active structural processing and attention sink behavior, showing that these phenomena have different implications for retrieval.

These findings have practical implications for retrieval system design. When documents contain structured notation and queries are short, small models with efficient attention patterns may outperform larger alternatives. Starting from such models enables effective domain adaptation through contrastive learning: PseudoSeer's embedding model surpasses BM25 by 5.5 points and outperforms the best pretrained model (GTE-ModernBERT) by 8.7 points. Hybrid retrieval provides additional gains by combining learned semantic similarity with BM25's exact term matching, reaching 66.5% R@10. PseudoSeer is deployed at pseudoseer.ist.psu.edu. Beyond hybrid retrieval, the tension between attention concentration and content coverage motivates exploring alternative architectures that capture both focused and distributed document representations.

8 Limitations

The dataset only includes pseudocode from papers that contain `\begin{algorithm}` and `\end{algorithm}` tags. A machine learning-based approach could detect additional patterns and expand the dataset. Additionally, search functionality could be improved by leveraging transformer based models for other fields beyond reference snippets, particularly the LaTeX pseudocode field, exploring retrieval across a larger domain gap from NL to LaTeX. Moreover, incorporating ranking algorithms that dynamically adapt to query type could enhance precision based on query context.

Our evaluation uses captions as query proxies to avoid manual annotation. While captions resemble the short keyword-style queries common in search engines (Jansen et al., 2000), they could differ from real user queries in some ways. Captions share authorship with reference snippets, which may create lexical affinity that benefits term-matching approaches. However, caption quality varies by author: some captions remain highly generic even after length filtering, others use domain-specific shorthand or informal algorithmic notation, and not all captions are written with the expectation of being rendered in the PDF, introducing noise that can make retrieval harder than a well-formed user query would. Future work could supplement this evaluation with human-generated queries to further validate the benchmark.

References

Lukas Blecher, Guillem Cucurull, Thomas Scialom, and Robert Stojnic. 2023. Nougat: Neural optical understanding for academic documents. *Preprint*, arXiv:2308.13418.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. BGE M3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *Preprint*, arXiv:2402.03216.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? An analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286. Association for Computational Linguistics.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld. 2020. SPECTER: Document-level representation learning using citation-informed transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2270–2282, Online. Association for Computational Linguistics.

Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, page 758–759, New York, NY, USA. Association for Computing Machinery.

Bernard J Jansen, Amanda Spink, and Tefko Saracevic. 2000. Real life, real users, and real needs: A study and analysis of user queries on the web. *Information Processing & Management*, 36(2):207–227.

Michael Kohlhase, Bogdan A. Matican, and Corneliu C. Prodescu. 2012. Mathwebsearch 0.5 – scaling an open formula search engine. In *Intelligent Computer Mathematics*, LNAI, pages 342–357. Springer.

Michael Kohlhase and Ioan Şucan. 2006. A search engine for mathematical formulae. In *Proceedings of Artificial Intelligence and Symbolic Computation, AISC'2006*, LNAI, pages 241–253. Springer.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374. Association for Computational Linguistics.

Sumith Kulal, Panupong Pasupat, Kartik Chandra, Mina Lee, Oded Padon, Alex Aiken, and Percy Liang. 2019. Spoc: Search-based pseudocode to code. *CoRR*, abs/1906.04908.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Patrice Lopez. 2008–2023. GROBID. https://github.com/kermitt2/grobid.

Ioana Normann and Michael Kohlhase. 2007. Extended formula normalization for ϵ-retrieval and sharing of mathematical knowledge. In *MKM/Calculemus - Towards Mechanized Mathematical Assistants*, LNAI, pages 266–279. Springer.

Yusuke Oda, Hiroyuki Fudaba, Graham Neubig, Hideaki Hata, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2015. Learning to generate pseudo-code from source code using statistical machine translation (t). *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 574–584.

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

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and permuted pre-training for language understanding. *Preprint*, arXiv:2004.09297.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*.

Levent Toksoz, Gang Tan, and C. Lee Giles. 2024. Automatic pseudocode extraction at scale. In *2024 IEEE International Conference on Information Reuse and Integration for Data Science (IRI)*, pages 264–269.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808. Association for Computational Linguistics.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Preprint*, arXiv:2002.10957.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2024. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations (ICLR)*.

Haoyi Xiong, Jiang Bian, Yuchen Li, Xuhong Li, Mengnan Du, Shuaiqiang Wang, Dawei Yin, and Sumi Helal. 2024. When search engine services meet large language models: Visions and challenges. *Preprint*, arXiv:2407.00128.

Zhichao Xu, Fengran Mo, Zhiqi Huang, Crystina Zhang, Puxuan Yu, Bei Wang, Jimmy Lin, and Vivek Srikumar. 2025. A survey of model architectures in information retrieval. *Preprint*, arXiv:2502.14822.

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. *Preprint*, arXiv:2407.10671.

Puxuan Yu, Luke Merrick, Gaurav Nuti, and Daniel Campos. 2024. Arctic-embed 2.0: Multilingual retrieval without compromise. *Preprint*, arXiv:2412.04506.

Maksym Zavershynskyi, Alex Skidanov, and Illia Polosukhin. 2018. Naps: Natural program synthesis dataset. *Preprint*, arXiv:1807.03168.

Dun Zhang, Jiacheng Li, Ziyang Zeng, and Fulong Wang. 2025. Jasper and stella: distillation of sota embedding models. *Preprint*, arXiv:2412.19048.

Xin Zhang, Yanzhao Zhang, Dingkun Long, Wen Xie, Ziqi Dai, Jialong Tang, Huan Lin, Baosong Yang, Pengjun Xie, Fei Huang, et al. 2024. mGTE: Generalized long-context text representation and reranking models for multilingual text retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1393–1412.