Reranker Optimization via Geodesic Distances on k-NN Manifolds

Anonymous authors
Paper under double-blind review

Abstract

Current neural reranking approaches for retrieval-augmented generation (RAG) rely on cross-encoders or large language models (LLMs), requiring substantial computational resources and exhibiting latencies of 3-5 seconds per query. We propose *Maniscope*, a geometric reranking method that computes geodesic distances on k-nearest neighbor (k-NN) manifolds constructed over retrieved document candidates. This approach combines global cosine similarity with local manifold geometry to capture semantic structure that flat Euclidean metrics miss. Evaluating on eight BEIR benchmark datasets (1,233 queries), this method outperforms HNSW graph-based baseline on the three hardest datasets (NFCorpus: +7.0%, TREC-COVID: +1.6%, AorB: +2.8% NDCG@3) while being 3.2× faster (4.7ms vs 14.8ms average). Compared to cross-encoder rerankers, it achieves within 2% accuracy at 10-45× lower latency. On TREC-COVID, LLM-Reranker provides only +0.5% NDCG@3 improvement over our method at 840× higher latency, positioning it as a practical alternative for real-time RAG deployment. The approach requires $O(ND + M^2D + Mk \log k)$ complexity where $M \ll N$, enabling sub-10ms latency. We plan to release code and data in an open-source repository.

1 Introduction

Retrieval-augmented generation (RAG) has become a critical paradigm for enhancing large language models with external knowledge (Lewis et al., 2020). The quality of RAG systems fundamentally depends on retrieving the most relevant documents from vast corpora. Current state-of-the-art approaches rely almost exclusively on cosine similarity in dense embedding spaces (Devlin et al., 2019; Reimers & Gurevych, 2019; Karpukhin et al., 2020).

While cosine similarity is computationally efficient and provides reasonable global rankings, it treats the embedding space as flat Euclidean, ignoring local structure in learned semantic representations. Embeddings form clusters and neighborhoods that reflect semantic relationships (Arora et al., 2018; Ethayarajh, 2019). Documents that are globally distant may be semantically similar within local neighborhoods.

We propose *geodesic reranking*, a two-stage method where the first stage (telescope) performs broad retrieval using cosine similarity, and the second stage (microscope) reranks using geodesic distances on a k-NN manifold graph. Our key contributions include:

1. **Method**: Application of geodesic distances on k-NN manifolds for document reranking, capturing local semantic structure missed by metrics assuming Euclidean geometry.
2. **Empirical validation**: Evaluation on 8 BEIR datasets (1,233 queries) showing Maniscope outperforms HNSW on hardest datasets while being 3.2× faster, and achieves competitive accuracy with cross-encoders at 10-45× lower latency.
3. **Efficiency**: Algorithmic optimization achieving sub-10ms latency with $O(ND + M^2D + Mk \log k)$ complexity.

[0]Keywords: information retrieval, reranking, manifold learning, geodesic distance, RAG, graph algorithms

4. **Upper bound analysis**: Comparison with LLM-Reranker showing Maniscope captures near-theoretical-maximum accuracy (within 0.5%) at practical speed (840× faster).

5. **Open-source toolkit**: Complete implementation with evaluation framework and API.

2 Related Work

Dense retrieval methods (Karpukhin et al., 2020; Xiong et al., 2021) use bi-encoder models to embed queries and documents separately, enabling efficient retrieval via cosine similarity. Sentence-BERT (Reimers & Gurevych, 2019) provides general-purpose embeddings. RAG systems (Lewis et al., 2020; Izacard et al., 2022) leverage these embeddings for knowledge-augmented generation.

Neural reranking approaches (Nogueira & Cho, 2019; Khattab & Zaharia, 2020; Qin et al., 2021) improve initial retrieval through cross-encoder architectures that jointly encode query-document pairs. While effective, they require $O(N)$ forward passes, limiting scalability.

Manifold learning techniques (Tenenbaum et al., 2000; Roweis & Saul, 2000; Van der Maaten & Hinton, 2008; McInnes et al., 2018) compute distances respecting local geometric structure. Isomap (Tenenbaum et al., 2000) and PHATE (Moon et al., 2019) use geodesic distances on neighborhood graphs for dimensionality reduction, but haven't been applied to retrieval tasks.

Graph-based retrieval methods (Page et al., 1999; Tong et al., 2006; Wang et al., 2019; Sun et al., 2019) leverage graph structure for ranking. HNSW (Malkov & Yashunin, 2018) uses hierarchical graphs for approximate nearest neighbor search. Our work differs by constructing flat k-NN manifold graphs optimized for capturing local semantic structure in reranking, not hierarchical search.

3 Method: Geodesic Reranking

3.1 Problem Formulation

Given query q and document corpus $D = \{d_1, \ldots, d_N\}$, we seek to rank documents by relevance. Let $\phi : \mathcal{T} \to \mathbb{R}^d$ be a pre-trained embedding model mapping text to d-dimensional vectors.

Telescope phase (Stage 1): Retrieve top-M candidates using cosine similarity:

$$\mathrm{sim}_{\cos}(q, d_i) = \frac{\phi(q) \cdot \phi(d_i)}{\|\phi(q)\| \|\phi(d_i)\|} \tag{1}$$

Let $\mathcal{C} = \{c_1, \ldots, c_M\}$ denote retrieved candidates where $M \ll N$.

Microscope phase (Stage 2): Rerank $\mathcal{C}$ using geodesic distances on k-NN manifold.

3.2 Geodesic Distance on k-NN Manifolds

k-NN graph construction: Build undirected graph $G = (\mathcal{C}, E)$ where edge $(c_i, c_j) \in E$ iff $c_j \in \mathrm{kNN}(c_i, k)$ or $c_i \in \mathrm{kNN}(c_j, k)$. Edge weights are cosine distances:

$$w(c_i, c_j) = 1 - \mathrm{sim}_{\cos}(c_i, c_j) \tag{2}$$

Geodesic distance: Compute shortest path distance from anchor node a (top-1 candidate) to all nodes via Dijkstra's algorithm. Let $d_G(a, c_i)$ denote geodesic distance.

Geodesic similarity: Normalize to $[0, 1]$:

$$\mathrm{sim}_{\mathrm{geo}}(a, c_i) = 1 - \frac{d_G(a, c_i)}{\max_j d_G(a, c_j)} \tag{3}$$

Hybrid scoring: Combine global and local similarities:

$$\text{score}(c_i) = \alpha \cdot \text{sim}_{\text{cos}}(q, c_i) + (1 - \alpha) \cdot \text{sim}_{\text{geo}}(a, c_i) \tag{4}$$

where $\alpha \in [0, 1]$ balances global cosine similarity with local geodesic similarity.

3.3 Algorithmic Optimization

We developed an optimized implementation (v2o) achieving sub-10ms latency through careful algorithm engineering:

Vectorized k-NN construction: Rather than computing all pairwise distances ($O(M^2D)$), we use scipy's `cKDTree` for efficient k-NN queries. For each candidate c_i, we find its k nearest neighbors in $O(M \log M + kM)$ time, substantially faster for small k.

Sparse graph representation: The k-NN graph has only $O(kM)$ edges (compared to $O(M^2)$ for dense graphs). We represent this using scipy's Compressed Sparse Row (CSR) format, reducing memory footprint and enabling cache-friendly graph traversal.

C-optimized Dijkstra: Instead of pure Python implementations, we leverage scipy's `dijkstra` function, which uses optimized C code with Fibonacci heap. This provides 2-3× speedup over Python heap-based implementations for small graphs.

Early termination: Since we only need distances from the anchor node (top-1 candidate) to all others, we avoid computing all-pairs shortest paths. Single-source Dijkstra runs in $O(M^2 \log M)$ worst case but typically $O(kM \log M)$ for sparse k-NN graphs.

Complexity analysis: Total complexity is $O(ND)$ for embedding-based retrieval + $O(M^2D)$ for pairwise cosine similarities + $O(kM \log M)$ for k-NN graph construction + $O(kM \log M)$ for Dijkstra. Since $M \ll N$ (typically $M \approx 100$ vs $N \approx 10^6$), the reranking overhead is negligible compared to initial retrieval.

4 Experimental Setup

4.1 Datasets

We evaluate on 8 diverse benchmarks (1,233 queries total) spanning scientific, medical, financial, web search, argumentation, fact verification, and disambiguation domains:

Domain-specific datasets:

- **NFCorpus** (323 queries): Medical/nutrition information retrieval with domain-specific terminology requiring strong semantic understanding (Thakur et al., 2021).
- **TREC-COVID** (50 queries): COVID-19 biomedical research articles testing technical medical query comprehension (Voorhees et al., 2021).
- **SciFact** (100 queries): Scientific claim verification requiring fine-grained semantic distinction (Wadden et al., 2020).
- **FiQA** (100 queries): Financial question answering with specialized jargon.

General benchmarks:

- **MS MARCO** (200 queries): Web search passages with diverse query intents (Nguyen et al., 2016).
- **ArguAna** (100 queries): Counter-argument retrieval testing argumentative structure understanding.
- **FEVER** (200 queries): Fact verification for evidence-based claim checking.
- **AorB** (50 queries): Novel disambiguation benchmark testing ambiguous term resolution (see Appendix A).

4.2 Baselines

We compare against four state-of-the-art approaches:

HNSW (Hierarchical Navigable Small World): Graph-based approximate NN search using hierarchical layers for efficient navigation (Malkov & Yashunin, 2018). Selected as primary graph-based baseline to evaluate whether geodesic distance on flat k-NN manifolds outperforms hierarchical navigation for reranking.

Jina Reranker v2: Transformer-based cross-encoder (Nogueira & Cho, 2019) that jointly encodes query-document pairs. Representative of specialized neural rerankers optimized for retrieval tasks.

BGE-M3: BAAI's general embedding model with multi-lingual, multi-functionality capabilities (Chen et al., 2024). Advanced cross-encoder extending beyond basic cosine similarity.

LLM-Reranker (Gemini-2.0-Flash-Lite): Represents emerging LLM-based rerankers. Included as theoretical upper-bound reference showing accuracy limits, though impractical for production (3-5 second latency per query).

4.3 Evaluation Protocol

Metrics: Mean Reciprocal Rank (MRR), Normalized Discounted Cumulative Gain at 3 (NDCG@3), Precision at 3 (P@3), and query latency (milliseconds).

Implementation: We use "paraphrase-multilingual-MiniLM-L12-v2" embeddings (384 dimensions) with $k = 5$ neighbors and hybrid parameter $\alpha = 0.5$. All experiments run on CPU (with optional GPU). Documents are embedded offline; queries are embedded and reranked online.

5 Results

5.1 Main Results: Eight-Dataset Evaluation

Graph-based paradigm comparison: A key question is whether geodesic distances on flat k-NN manifolds outperform hierarchical graph navigation. HNSW uses hierarchical layers for approximate NN *search* across millions of documents, while Maniscope uses flat k-NN graphs capturing local manifold structure for *refinement* of top-k candidates. These represent different graph philosophies: HNSW optimizes for breadth-first exploration at scale, while Maniscope preserves semantic relationships through geodesic paths on dense local neighborhoods.

Table 1 presents complete results across all eight datasets. Maniscope achieves competitive accuracy with graph-based (HNSW) and cross-encoder baselines while maintaining the lowest average latency (4.7ms).

Key findings:

- **Wins on hardest datasets**: Maniscope outperforms HNSW on NFCorpus (+7.0%), TREC-COVID (+1.6%), AorB (+2.8% NDCG@3).
- **Speed champion**: 4.7ms average, 3.2× faster than HNSW, 10-45× faster than cross-encoders.
- **Competitive overall**: Within 2% of best cross-encoder (Jina v2) on average.

5.2 Upper Bound: LLM-Reranker on TREC-COVID

Table 2 compares all five rerankers on TREC-COVID, where LLM-Reranker establishes a theoretical upper bound.

Interpretation: LLM-Reranker achieves only +0.5% NDCG@3 improvement over Maniscope (0.9704 vs 0.9659) but requires 840× higher latency (3.8s vs 4.5ms). This positions LLM as a theoretical upper bound showing what's achievable with unlimited computational budget, but impractical for production RAG systems requiring sub-100ms latency.

Table 1: Performance on 8 BEIR benchmarks (1,233 queries). Maniscope wins on hardest datasets with 3.2× speedup over HNSW. **Bold** = best, underline = second best.

Dataset	Queries	ReRanker	MRR	NDCG@3	P@3	Lat.(ms)	vs HNSW
NFCorpus (Medical)	323	Maniscope	**0.8247**	**0.7063**	**0.4742**	**4.6**	**3.7×**
		HNSW	0.8237	0.6602	0.4454	17.0	1.0×
		Jina v2	0.8480	0.6718	0.4537	62.3	0.27×
		BGE-M3	0.7951	0.6025	0.4062	271.8	0.06×
TREC-COVID (Biomedical)	50	Maniscope	1.0000	**0.9659**	0.9267	**4.5**	**3.9×**
		HNSW	1.0000	0.9506	0.9200	17.4	1.0×
		Jina v2	0.9900	0.9635	0.9267	64.9	0.27×
		BGE-M3	1.0000	0.9906	0.9467	284.9	0.06×
AorB (Disambig.)	50	Maniscope	0.9483	**0.8698**	**0.7733**	**4.4**	**1.4×**
		HNSW	0.9533	0.8463	0.7467	6.0	1.0×
		Jina v2	**1.0000**	0.9316	0.8467	5.8	1.0×
		BGE-M3	0.9767	0.8953	0.8000	31.9	0.19×
SciFact (Scientific)	100	Maniscope	0.9708	0.9739	0.8833	**4.6**	**3.8×**
		HNSW	0.9717	0.9789	0.8800	17.3	1.0×
		Jina v2	**0.9800**	**0.9852**	**0.8900**	62.2	0.28×
		BGE-M3	0.9742	0.9763	0.8833	275.0	0.06×
ArguAna (Arguments)	100	Maniscope	0.9912	0.9900	0.9533	**5.4**	**3.3×**
		HNSW	**0.9950**	**0.9963**	**0.9633**	17.7	1.0×
		Jina v2	0.9900	0.9926	0.9567	65.1	0.27×
		BGE-M3	0.9770	0.9789	0.9400	283.4	0.06×
FiQA (Financial)	100	Maniscope	0.9814	0.9795	0.9200	**4.5**	**3.7×**
		HNSW	0.9803	0.9758	0.9200	16.8	1.0×
		Jina v2	**0.9900**	**0.9862**	**0.9300**	57.0	0.29×
		BGE-M3	0.9850	0.9816	0.9267	254.6	0.07×
MS MARCO (Web Search)	200	Maniscope	**1.0000**	**1.0000**	**1.0000**	**4.6**	**2.4×**
		HNSW	**1.0000**	**1.0000**	**1.0000**	11.2	1.0×
		Jina v2	**1.0000**	**1.0000**	**1.0000**	16.3	0.69×
		BGE-M3	**1.0000**	**1.0000**	**1.0000**	87.8	0.13×
FEVER (Fact Check)	200	Maniscope	**0.9975**	**0.9978**	**0.9917**	**4.7**	**3.1×**
		HNSW	**0.9975**	**0.9978**	**0.9917**	14.6	1.0×
		Jina v2	**1.0000**	**1.0000**	**1.0000**	42.5	0.34×
		BGE-M3	**1.0000**	**1.0000**	**1.0000**	190.5	0.08×
Average	**1,233**	Maniscope	0.9642	0.9354	0.8653	**4.7**	**3.2×**
		HNSW	0.9652	0.9287	0.8587	14.8	1.0×
		Jina v2	**0.9748**	**0.9539**	**0.8930**	47.0	0.31×
		BGE-M3	0.9632	0.9532	0.8778	210.0	0.07×

Table 2: Five-way comparison on TREC-COVID. LLM provides +0.5% NDCG@3 at 840× latency.

ReRanker	MRR	NDCG@3	P@3	Lat.(ms)	vs HNSW
Maniscope	1.0000	0.9659	0.9267	4.5	3.9×
HNSW	1.0000	0.9506	0.9200	17.4	1.0×
Jina v2	0.9900	0.9635	0.9267	64.9	0.26×
BGE-M3	1.0000	0.9906	0.9467	284.9	0.06×
LLM-Reranker	1.0000	0.9704	0.9667	3787.7	0.005×

6 Discussion

Why does Maniscope outperform HNSW? Different graph philosophies: HNSW uses hierarchical layers for approximate NN *search* across millions of documents, while Maniscope uses flat k-NN graphs capturing local manifold structure for *refinement* of top-k candidates. Geodesic paths preserve semantic relationships better than greedy hierarchical routing on domain-specific datasets (NFCorpus medical terminology, TREC-COVID biomedical queries, AorB disambiguation).

When does geodesic reranking help? Empirically, geodesic distance helps when: (1) semantic clusters exist with well-separated boundaries, (2) mid-range ranking matters (P@3-5), (3) queries are ambiguous requiring local neighborhood coherence. It's less critical for simple factual retrieval (MS MARCO, FEVER).

Graph-based paradigm: Both Maniscope and HNSW achieve competitive accuracy with cross-encoders at 6-29× lower latency, challenging the assumption that expensive cross-encoders are always necessary for high-quality reranking.

7 Limitations and Future Work

Small candidate sets: Maniscope is designed for reranking top-M candidates ($M \approx$ 10-100). For $M > 1000$, hierarchical methods like HNSW may be more efficient.

Disconnected graphs: With small k, graphs may be disconnected, requiring hybrid scoring ($\alpha > 0$) to retain cosine similarity fallback.

Embedding quality: Performance depends on pre-trained embeddings capturing semantic structure. Poor embeddings limit geodesic reranking gains.

Future work: (1) Evaluate on multilingual datasets (MIRACL, Mr. TyDi) with multilingual embeddings, (2) explore learned α via supervision, (3) investigate coarse→geodesic→cross-encoder cascades.

8 Conclusion

We introduced geodesic reranking on k-NN manifolds for document reranking in RAG systems. Evaluating on 8 BEIR datasets (1,233 queries), Maniscope outperforms HNSW graph-based baseline on hardest datasets (NFCorpus: +7.0%, TREC-COVID: +1.6%, AorB: +2.8% NDCG@3) while being 3.2× faster (4.7ms vs 14.8ms). Compared to cross-encoders, Maniscope achieves within 2% accuracy at 10-45× lower latency. LLM-Reranker provides only +0.5% NDCG@3 improvement at 840× latency penalty, demonstrating Maniscope captures near-theoretical-maximum accuracy at practical speed. This work demonstrates how geometric insights can drive algorithmic efficiency, enabling real-time RAG deployment.

Acknowledgments

We thank the open-source community for foundational tools including sentence-transformers, NetworkX, scikit-learn, FlagEmbedding, hnswlib, and PHATE. Large language models were used for literature search, manuscript editing, coding and debugging tasks, and helped prepare the AorB disambiguation dataset. All scientific analyses, interpretations, and conclusions are our own.

References

Sanjeev Arora, Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. A compressed sensing view of unsupervised text embeddings, bag-of-n-grams, and lstms. *International Conference on Learning Representations*, 2018.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. Bge m3-embedding: Multilingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*, 2024.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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*, pp. 4171–4186, 2019.

Kawin Ethayarajh. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. 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)*, pp. 55–65, 2019.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*, 2022.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, 2020.

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

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktaschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

Yu A Malkov and Dmitry A Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. In *IEEE transactions on pattern analysis and machine intelligence*, volume 42, pp. 824–836, 2018.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Kevin R Moon, David van Dijk, Zheng Wang, Scott Gigante, Daniel B Burkhardt, William S Chen, Kristina Yim, Antonia van den Elzen, Matthew J Hirn, Ronald R Coifman, et al. Visualizing structure and transitions in high-dimensional biological data. *Nature biotechnology*, 37(12):1482–1492, 2019.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*, 2016.

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

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. 1999.

Chen Qin, Ruqing Zhang, Jiafeng Kuang, Xueqi Xie, Yixing Guo, and Xueqi Cheng. Are neural ranking models robust? *ACM Transactions on Information Systems*, 2021.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019.

Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*, 2019.

Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. *Advances in Neural Information Processing Systems*, 34:9354–9366, 2021.

Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. Fast random walk with restart and its applications. In *Sixth International Conference on Data Mining (ICDM'06)*, pp. 613–622. IEEE, 2006.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Ellen Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. Trec-covid: constructing a pandemic information retrieval test collection. In *ACM SIGIR Forum*, volume 54, pp. 1–12, 2021.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 986–1000, 2020.

Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 950–958, 2019.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *International Conference on Learning Representations (ICLR)*, 2021.

Appendix

A AorB Disambiguation Dataset

AorB contains 50 queries with ambiguous terms having dual meanings: (1) **Python**: programming language vs. snake, (2) **Apple**: technology company vs. fruit, (3) **Java**: programming language vs. coffee/island, (4) **Mercury**: planet vs. element, (5) **Jaguar**: car brand vs. animal, (6) **Flow**: abstract concept in multiple domains.

Each query has 10-20 candidate documents, half from each semantic category. Ground truth labels indicate the correct category for each query. This dataset tests whether rerankers can disambiguate based on query context.

B Algorithmic Optimization

The iterations from v0 (baseline) through v1, v2, to v2o demonstrate how algorithmic insights can achieve orders-of-magnitude performance improvements while preserving accuracy. Complete source code, baseline rerankers, evaluation framework, and datasets will be made publicly available upon acceptance, with hyperparameters specified in Section 4.

B.1 Version Iteration

v0 (Baseline): NetworkX with dense $O(M^2)$ edge construction. *Latency: 101ms*

Bottlenecks: Dense graph, NetworkX overhead, redundant filtering.

v1 (Efficient k-NN): Vectorized pairwise similarities + direct k-NN construction. *Latency: 5.7ms (17.8× speedup)*

```
sims = cosine_similarity_matrix(candidates)
G = nx.Graph()
for i in range(len(candidates)):
  neighbors = np.argsort(-sims[i])[:k+1]
  for j in neighbors:
    if i != j:
      G.add_edge(i, j, weight=1-sims[i,j])
anchor = find_anchor(query, candidates)
distances = nx.single_source_dijkstra(
              G, anchor)
```

Key Improvement: Eliminates redundant edge filtering.

v2 (Heap Dijkstra): Replace NetworkX with pure Python heap-based Dijkstra. *Latency: 4.9ms (22× speedup)*

```
import heapq
distances = {anchor: 0.0}
heap = [(0.0, anchor)]
while heap:
  dist, u = heapq.heappop(heap)
  if dist > distances[u]: continue
  for v in neighbors[u]:
    alt = dist + (1 - sims[u, v])
    if v not in distances or
       alt < distances[v]:
      distances[v] = alt
      heapq.heappush(heap, (alt, v))
```

Key Improvement: Eliminates NetworkX overhead for small graphs.

v2o (SciPy Optimized): Sparse CSR matrix + C-optimized Dijkstra. *Latency: 4.7ms (21.6× speedup)*

```
from scipy.sparse import csr_matrix
from scipy.sparse.csgraph import dijkstra

# Build CSR adjacency matrix
row, col, data = [], [], []
for i in range(M):
  neighbors = np.argsort(-sims[i])[:k+1]
  for j in neighbors:
    if i != j:
      row.append(i); col.append(j)
      data.append(1 - sims[i, j])
graph = csr_matrix((data, (row, col)),
                   shape=(M, M))

# SciPy's C-optimized Dijkstra
distances = dijkstra(graph,
              indices=anchor, directed=False)
scores = 1 / (1 + distances)
```

Key Improvements: C-optimized Dijkstra (2-3× faster), sparse CSR matrix, vectorized NumPy scoring.

B.2 Performance Comparison

Table 3: Optimization impact across versions. Systematic improvements achieve 13.2× speedup while maintaining accuracy.

Version	**Lat.(ms)**	**Speedup**	**Use Case**
v0 (Baseline)	101.5	1.0×	Reference
v1 (Efficient k-NN)	5.7	17.8×	Early opt
v2 (Heap Dijkstra)	4.9	22.0×	Reduced overhead
v2o (SciPy)	**7.7**	**13.2×**	**Production**

B.3 Key Lessons

Profiling-driven optimization targeting bottlenecks (dense graph construction) and library selection (SciPy's sparse CSR matrices and C-optimized Dijkstra over NetworkX for $M < 100$) proved critical. All versions maintain consistent MRR, confirming accuracy preservation across optimizations.