

# HQANN: Efficient and Robust Similarity Search for Hybrid Queries with Structured and Unstructured Constraints

Wei Wu\*

Heterogeneous Computing Center  
Kuaishou Technology, China  
wuwei10@kuaishou.com

Junlin He\*

Multimedia Understanding  
Kuaishou Technology, China  
hejunlin03@kuaishou.com

Yu Qiao

Multimedia Understanding  
Kuaishou Technology, China  
qiaoyu@kuaishou.com

Guoheng Fu

Multimedia Understanding  
Kuaishou Technology, China  
fuguoheng@kuaishou.com

Li Liu

Heterogeneous Computing Center  
Kuaishou Technology, China  
liliu@kwai.com

Jin Yu<sup>†</sup>

Multimedia Understanding  
Kuaishou Technology, China  
yujin07@kuaishou.com

## ABSTRACT

The in-memory approximate nearest neighbor search (ANNS) algorithms have achieved great success for fast high-recall query processing, but are extremely inefficient when handling hybrid queries with unstructured (*i.e.*, feature vectors) and structured (*i.e.*, related attributes) constraints. In this paper, we present HQANN, a simple yet highly efficient hybrid query processing framework which can be easily embedded into existing proximity graph-based ANNS algorithms. We guarantee both low latency and high recall by leveraging navigation sense among attributes and fusing vector similarity search with attribute filtering. Experimental results on both public and in-house datasets demonstrate that HQANN is 10x faster than the state-of-the-art hybrid ANNS solutions to reach the same recall quality and its performance is hardly affected by the complexity of attributes. It can reach 99% recall@10 in just around 50 microseconds On GLOVE-1.2M with thousands of attribute constraints.

## CCS CONCEPTS

• **Information systems** → **Query intent; Document filtering; Recommender systems; Retrieval efficiency.**

## KEYWORDS

hybrid query processing, proximity graph, navigable attribute search

## 1 INTRODUCTION

Nearest neighbor search for unstructured data such as texts, images, and videos has played an important role in large scale information retrieval systems. [16, 19–21] Typically, Unstructured data is first converted into high dimensional feature vectors, and subsequent similarity search is conducted on these vectors. As the cost of exact solutions for K-nearest neighbor search [27] is extremely high in big data scenarios, researchers have proposed many kinds of approximate nearest neighbor search (ANNS) algorithms [4, 7, 9–13]. However, most of the algorithms can only treat hybrid queries with structured and unstructured constraints as disjoint tasks, leading to high query latency and low recall rate, which contradicts the growing demand for hybrid query processing in modern recommendation systems [15, 18, 25]. Hybrid queries not only supplements

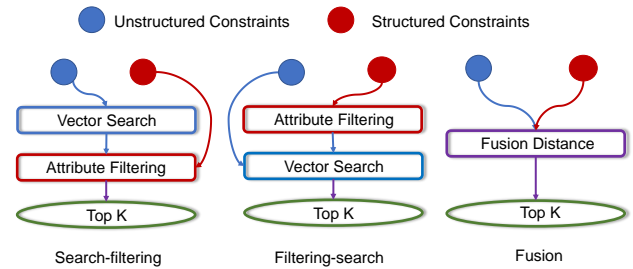


Figure 1: Three strategies for hybrid query processing.

the necessary condition information for retrieval, but also effectively improves the accuracy of ANNS algorithms for narrow down the vector search space.

There are only a few approaches working on hybrid query processing, including Vearch[15], Alibaba AnalyticDB-V (ADB-V) [25] and Milvus[23]. Vearch first conducts the original vector similarity search over the feature vectors and then filters these results with attributes, which can be easily extended to any ANNS algorithms. ADBV introduces a product quantization (PQ) [11] based hybrid query framework and enables vector similarity search with a pre-defined bitmap from attribute filtering. Milvus adds an additional partition stage based on frequently used attributes and split the dataset into several subsets, leading to a search space shrinking with attribute constraints. Both of them treat vector similarity search and attribute filtering as separate problems, as shown in Figure 1, which is not optimal in terms of efficiency or accuracy. The search-filtering strategy has to expand the list of candidates to compensate for the merging loss from two separate subquery systems, while the filtering-search strategy could suffer high latency for exhaustive scanning with a bitmap. Furthermore, the current state-of-the-art PQ-based hybrid query processing systems fail to take advantages of proximity graph-based ANNS, which have been proved far superior to PQ in speed and precision without memory concern.

Recently, [24] proposed a native hybrid query(NHQ) framework based on proximity graph. In NHQ, attribute values processed with *xor* functions are used as fine-tuning factors for the original distance metric, providing a specialized composite index and joint pruning modules for hybrid queries. However, NHQ regards the

\*These authors contributed equally to this work.

<sup>†</sup>Corresponding author.

original vector distance metric as dominant and fails to emphasize the importance of attributes, its performance drops dramatically as the number of attributes increases.

In this paper, we argue that attributes should be in the leading position for proximity graph-based hybrid query processing. We propose HQANN, a simple yet efficient hybrid query processing framework which can be easily embedded into any existing proximity graph-based ANNS algorithms. Following the fusion strategy in Figure 1, we merge the search and filtering stage by going through the composite graph, leading to high recall hybrid ANNS algorithms with negligible extra computation overhead.

The main contributions of this paper are as follows:

- We introduce a novel hybrid query processing framework, HQANN, which efficiently manages hybrid queries in a fused way. HQANN is generally applicable to many proximity graph-based ANNS algorithms. We demonstrate the flow of construction and search for proximity graphs can be efficiently adapted to the proposed framework.
- We propose to leverage the navigation sense among attributes to traverse the proximity graphs. We present the effectiveness of our method through analysis and experiments.
- We show that HQANN leads to large performance gains over previous hybrid query processing techniques. Our method achieves state-of-the-art performance on standard large-scale benchmarks such as Glove-1.2M. In addition to recall gains, HQANN greatly improves retrieval efficiency of original proximity graphs, with attribute constraints limiting the search space.

## 2 RELATED WORKS

### 2.1 Content-based Retrieval Systems

Content-based Retrieval systems have made great achievements in facilitating analytics on unstructured data, such as words, images and videos. In these systems, the learned embedding functions are used to map items in a common vector space, and subsequent retrievals are conducted. [6, 26] Although content-based retrieval system effectively support unstructured data, there are many scenarios where both unstructured and structured data should be jointly retrieved (we call them hybrid queries), a typical application is search under a specific limit (*i.e.* male or female). Besides, imposing structured constraints can narrow down the original vector search space and effectively improve the accuracy.

### 2.2 Methods for optimizing hybrid query processing

Based on the previously proposed high quality ANNS algorithms, Vearch [15], Alibaba AnalyticDB-V (ADB-V) [25] and Milvus [23] deployed two-stage hybrid processing systems, which treat queries over feature vectors and attributes as disjoint tasks. These systems are only friendly to space partitioning [3, 4, 12] or quantization [2, 5, 14] based ANNS solutions, for they have to conduct a detailed scan of the entire dataset with the blacklist from attribute filtering. To take advantage of proximity graph-based ANNS solutions [17, 22], NHQ [24] is proposed to provide a specialized composite index and joint pruning modules for hybrid queries.

Our work differs from the above methods as they fail to insert filtering stage naturally into search stage, while we develop an approach in the following section where we conduct a joint search strategy in a proximity graph.

## 3 HQANN

The performance of hybrid query processing is largely determined by the efficiency of settling attributes, as they determine the accurate search space for feature vectors. From this motivation, we propose HQANN. In HQANN, all attributes is treated as a native part of the fusion distance metric, which can make full use of the inherent Navigation feature of proximity graph-based ANNS solutions.

### 3.1 Fusion Distance Metric

The key point for hybrid query processing is to define an effective distance metric to facilitate analytics on attributes. Give the fused distance function as the basic premise, a composite proximity graph-based index can be constructed and retrieved without invading its original execution flow.

Generally, performance of a proximity graph is largely determined by the quality of the graph node neighborhoods. Since the feature vector search space of hybrid queries is determined by the attributes, we treat attributes as the dominant part of the fusion distance metric, which means a graph node tends to establish links with datapoints containing the same or vary similar attributes. In such circumstances, a hybrid query can quickly locate those graph nodes with exactly the same attributes and conduct detailed vector similarity search among their neighborhoods.

Given an hybrid dataset  $S$ , with  $m$ -dimensional feature vector data space  $X$  and  $n$ -dimensional attribute data space  $V$  (we assume that the attributes have been pre-processed into vectors), The distance between two hybrid datapoints  $s_i, s_j \in S$  can be measured as:

$$Dist(s_i, s_j) = Dist(x_i, x_j) + Dist(v_i, v_j). \quad (1)$$

Here  $Dist(x_i, x_j)$  can represent any frequently-used vector distance metric (*i.e.* euclidean distance). Let  $g(x, y)$  and  $f(x, y)$  denote distance functions for feature vectors and attributes separately, and the fusion distance metric can be computed with:

$$Dist(s_i, s_j) = w \cdot g(x_i, x_j) + f(v_i, v_j), \quad (2)$$

where  $f(v_i, v_j)$  satisfies

$$f(v_i, v_j) = \begin{cases} 0, & v_i = v_j \\ bias - \frac{1}{\lg(e^{(v_i, v_j)} + 1)}, & v_i \neq v_j \end{cases}, \quad (3)$$

with  $e(v_i, v_j)$  and  $bias$  defined as follows:

$$e(v_i, v_j) = \sum_{k=1}^n |v_i[k] - v_j[k]| \quad (4)$$

$$bias \gg \max(w \cdot g(x_i, x_j)) + \frac{1}{\lg 2}$$

Note that attribute vectors should only contain integers in production environments, thus  $\min(e(v_i, v_j)) = 1$ . We see that when two hybrid datapoints share the same attributes, their attribute distance is defined as zero. Otherwise,  $bias$  is applied first to keep

attribute distance much larger than feature vector distance, which guarantees the closet neighbors of a hybrid datapoint should have exactly matched attributes. Then  $\frac{1}{\lg(e^{(v_i, v_j)} + 1)}$  is attached as a fine-tuning effect for attribute distance values, leading to smaller distances among attributes with higher similarity. After we obtain the distances among attribute parts, they are fused with feature vector distances with a scale factor  $w$ , we tune  $w$  to ensure the feature vector distances are much smaller than attribute distances, while avoid them being too small to be ignored.

The mapping function  $e(x)$  we used here represents the Manhattan Distance among two attribute vectors, as we only focus on the representation space of attribute distances instead of their accurate values. For example, if  $e(v_i, v_j)$  performs a *xor* operation for all dimension of two attribute vectors and sum them up, then return value of  $f(v_i, v_j)$  would be the same for a mass of combinations of attributes, this deprives the navigation sense among attributes and the composite proximity graph will degenerate into a normal one with feature vector similarity information only. In fact, any mapping function that is capable of preserving attribute representation space can be used as  $e(x)$  (*i.e.* Euclidean distance), we select Manhattan Distance here for computational efficiency. The  $\lg$  function is applied to prevent attribute values from being too large and erases the gap among different attributes.

Eq.2-4 provide a simple and efficient solution to fuse two disjoint distance metrics. In our experimental study,  $w = 0.25$  and  $bias = \max(g(x_i, x_j)) + \frac{1}{\lg 2}$  work well for most datasets. We want to point out that in most production scenarios these feature vectors  $x \in X$  are pre-normalized with inner product(IP) as the distance metric, which means  $\max(g(x_i, x_j)) = 1$ , thus we do not need to tune these hyper parameters specifically for most cases (to satisfy minimum priority in fusion distance metric, let  $g(x_i, x_j) = 1 - x_i \cdot x_j$  for IP).

### 3.2 Composite Proximity Graph

The basic idea of HQANN is that we obtain a fusion distance metric by fine-tuning the attribute distances via feature vector distances. More specifically, the distance between two hybrid datapoints is mainly defined by their attribute distance, and their feature vector distance serves as a adjustment factor with limited impact on the fusion distance. During the construction stage of the proximity graphs, datapoints with the same or very similar attributes will be linked first as they tend to have much smaller distance compared with our candidates. Then the remaining neighborhood vacancies would be filled up with datapoints that are relatively distant in attributes, which strongly maintains the connectivity of the graph. In the search stage, a hybrid query traverses the graph until it reaches the first node with the same attribute, by fully scanning the node’s neighborhoods, a high quality return candidate list can be efficiently accessed.

The overall configuration of the composite proximity graph is shown in Figure 2. With attribute distances play a major role in the fusion distance metric, we actually accomplish the filtering-search strategy within a single traversing of the graph. The numerical differences among attribute distances and feature vector distances skillfully exploiting the proximity graphs’ essence of voracious

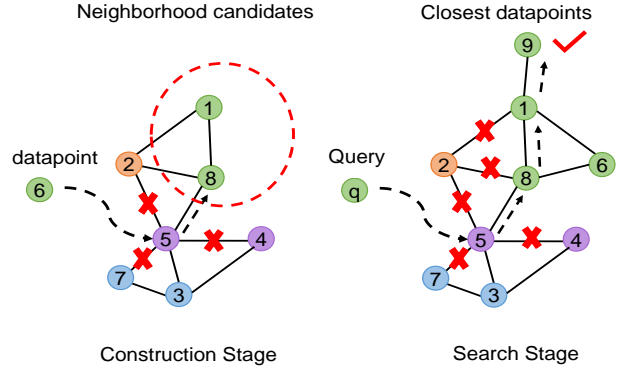


Figure 2: Work flow of HQANN framework.

searching for graph nodes closest to the query. Hence, HQANN is robust against scale of attributes and remains highly efficient.

## 4 EXPERIMENTS

In this section, we demonstrate that HQANN is more efficient and robust compared with previous state-of-the-art hybrid processing methods, and show that under attribute constraints, the composite graph surpasses the original proximity graph in both latency and recall rate.

### 4.1 Experiment Setup

We conduct all the experiments on an Intel Xeon E5-2680(2400MHz frequency) with 16 CPU threads. The datasets we used are four public datasets GLOVE-1.2M, SIFT-1M, GIST-1M, DEEP-1B from the ANNS benchmark website [1], and an extra billion-scale in-house merchandise dataset with millions of merchant IDs as attribute constraints. As public datasets do not originally contain structured constraints, we generate attributes randomly for all datapoints following the same method in [23]. All datapoints are pre-normalized and we set  $w = 0.25$ ,  $bias = 4.32$  by default unless otherwise specified.

### 4.2 Recall-Speed Benchmark

To evaluate effectiveness of hybrid query processing algorithms in a realistic setting, it is important to performance end-to-end retrieval experiments and compare speed-recall curves. We choose three state-of-the-art two-stage hybrid query processing frameworks(Vearch [15], ADBV [25], milvus [23]) and the recently proposed proximity graph-based algorithm (NHQ [24]) as our comparison targets. Our implementations of two-stage frameworks build on product quantization with SIMD based ADC [8] for distance computation, which offers a strong baseline to confirm superiority of HQANN. We fix the bit-rate at  $dimension \times 4bits$  and carefully tune the search space to observe reasonable recall-speed results. Both HQANN and NHQ are implemented on HNSW [17]], with the same hyper parameters for all datasets. ( $M = 64$ ,  $efConstruction = 512$ ). The number of possible attribute constraints we used here is 100.

The performance results on four public datasets are shown in Figure 3. It demonstrate that HQANN significantly outperforms

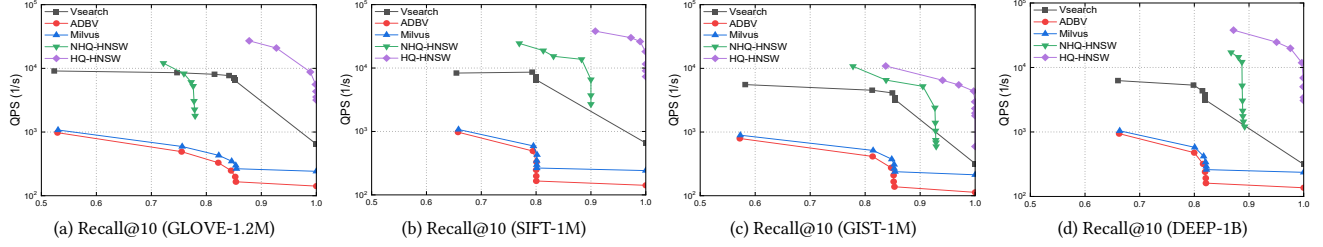


Figure 3: Speed-recall trade-off on different datasets

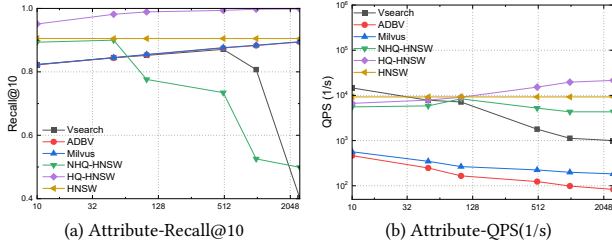


Figure 4: Recall and Speed performance on GLOVE-1.2M with different scale of attribute constraints

competing methods in terms of speed and recall rate (up and to the right is better) on all four datasets and it can obtain a recall of 99% for all datasets. Specially, HQANN is more than ten times faster than other methods to reach the good recall quality (95%).

### 4.3 Robustness Verification

Then we conduct an experiment on GLOVE-1.2M to compare the robustness with existing hybrid ANNS solutions. For Vearch, we return 100 times more candidates (1000 for Recall@10) for further attribute filtering. For ADBV and milvus, we traverse all data-points outside the blacklist with product quantization based ADC. For NHQ and HQANN, we fix the same hyper parameters for HNSW ( $M = 64$ ,  $efConstruction = 512$ ,  $efSearch = 80$ ). In addition, to evaluate the hypothesis that using attributes to limit search space can improve search performance, we attach the search performance of HNSW on GLOVE-1.2M without attribute constraints as a baseline with exactly the same settings. We gradually increase the number of attribute constraints from 10 to 2500, detailed results are shown in Figure 4. We see that Vearch and NHQ suffer from dramatic accuracy drop, and all other methods become several times slower with the increasing categories of attributes. On the contrary, HQANN can maintain very high recall quality (over 95% Recall@10) and its efficiency increases as attribute constraints become more complex. It also demonstrates that vector similarity search can benefit from attribute constraints in both latency and recall for its filtering feature. For example, with 2500 attribute constraints, the composite graph is two times faster with 10% recall gains.

Table 1: Recall@10 comparison with different settings of  $w$  on GLOVE-1.2M and merchandise datasets, values in parentheses represent the number of attribute constraints.

Dataset	Scale Factor $w$			
	$w=1.00$	$w=0.50$	$w=0.25$	$w=0.10$
GLOVE-1.2M (10)	0.986	0.987	<b>0.989</b>	0.988
GLOVE-1.2M (100)	0.947	0.949	<b>0.950</b>	0.950
merchandise-0.2B (0.8M)	0.727	0.955	<b>0.999</b>	0.999

### 4.4 Parameter Sensitivity

Finally, We select some typical scenarios to prove that we do not need to tune these hyper parameters of HQANN for specific datasets. Since  $bias$  just need to satisfy  $bias \gg w + 3.32$ , we fix  $bias = 4.32$  and verify the proper ratio of attribute distances to feature vector distances by simply shrinking  $w$ . Table 1 shows that for simple scenarios such as GLOVE-1.2M with hundreds of attributes,  $w = 1$  can reach a reasonable recall quality, while for big data scenarios with millions of attribute constrains, shrinking  $w$  from 1 to 0.25 brings huge recall benefits. It also proves that further shrinking  $w$  cause no recall gains, which means  $w = 0.25$  is a suitable critical point for aligning the ratio of two distance parts. We stop reduce  $w$  any further in case feature vector part is too small to be completely ignored.

## 5 CONCLUSIONS

In this paper, we propose HQANN as a new hybrid query processing framework. The new framework leverages navigation sense among attribute constraints, leading to a fusion distance metric which can be flexibly embedded into proximity graph-based ANNS algorithms. Our experiments show superior performance on retrieval recall and efficiency compared with existing hybrid query solutions.

## REFERENCES

- [1] Martin Aumüller, Erik Bernhardsson, and Alexander Faithfull. 2020. ANN-Benchmarks: A benchmarking tool for approximate nearest neighbor algorithms. *Information Systems* (2020).
- [2] Artem Babenko and Victor Lempitsky. 2014. Additive quantization for extreme vector compression. In *IEEE Conference on Computer Vision and Pattern Recognition*. 931–938.
- [3] Artem Babenko and Victor Lempitsky. 2014. The inverted multi-index. *IEEE transactions on pattern analysis and machine intelligence* (2014), 1247–1260.
- [4] Dmitry Baranchuk, Artem Babenko, and Yury Malkov. 2018. Revisiting the inverted indices for billion-scale approximate nearest neighbors. In *European Conference on Computer Vision*. 202–216.
- [5] Tiezheng Ge, Kaiming He, Qifa Ke, and Jian Sun. 2013. Optimized product quantization for approximate nearest neighbor search. In *IEEE Conference on Computer Vision and Pattern Recognition*. 2946–2953.
- [6] Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldrige, Eugene Ie, and Diego Garcia-Olano. 2019. Learning Dense Representations for Entity Retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*. Association for Computational Linguistics, Hong Kong, China, 528–537. <https://doi.org/10.18653/v1/K19-1049>
- [7] Yunchao Gong, Svetlana Lazebnik, Albert Gordo, and Florent Perronnin. 2011. Iterative quantization: A procrustean approach to learning binary codes. In *IEEE Conference on Computer Vision and Pattern Recognition*. 2916–2929.
- [8] Ruiqi Guo, Sanjiv Kumar, Krzysztof Choromanski, and David Simcha. 2015. Quantization based Fast Inner Product Search. (09 2015).
- [9] Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. 2020. Accelerating large-scale inference with anisotropic vector quantization. In *International Conference on Machine Learning*. 3887–3896.
- [10] Suhas Jayaram Subramanya, Fnu Devvrit, Harsha Vardhan Simhadri, Ravishankar Krishnawamy, and Rohan Kadekodi. 2019. DiskANN: Fast Accurate Billion-point Nearest Neighbor Search on a Single Node. In *Advances in Neural Information Processing Systems*, Vol. 32.
- [11] Hervé Jegou, Douze Matthijs, and Cordelia Schmid. 2010. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence* (2010), 117–128.
- [12] Hervé Jégou, Romain Tavenard, Matthijs Douze, and Laurent Amsaleg. 2011. Searching in one billion vectors: re-rank with source coding. In *IEEE International Conference on Acoustics, Speech and Signal Processing*. 861–864.
- [13] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data* (2017).
- [14] Yannis Kalantidis and Yannis Avrithis. 2014. Locally optimized product quantization for approximate nearest neighbor search. In *IEEE Conference on Computer Vision and Pattern Recognition*. 2321–2328.
- [15] Jie Li, Haifeng Liu, Chuanghua Gui, Jianyu Chen, Zhenyuan Ni, Ning Wang, and Yuan Chen. 2018. The Design and Implementation of a Real Time Visual Search System on JD E-Commerce Platform. 9–16. <https://doi.org/10.1145/3284028.3284030>
- [16] Kinchen Liu, Wu Liu, Huadong Ma, and Huiyuan Fu. 2016. Large-scale vehicle re-identification in urban surveillance videos. In *2016 IEEE International Conference on Multimedia and Expo (ICME)*. 1–6. <https://doi.org/10.1109/ICME.2016.7553002>
- [17] Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence* (2018), 824–836.
- [18] Jianbin Qin, Wei Wang, Chuan Xiao, Ying Zhang, and Yaoshu Wang. 2021. High-Dimensional Similarity Query Processing for Data Science. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (Virtual Event, Singapore) (KDD '21)*. Association for Computing Machinery, New York, NY, USA, 4062–4063. <https://doi.org/10.1145/3447548.3470811>
- [19] Fred Richardson, Douglas Reynolds, and Najim Dehak. 2015. A Unified Deep Neural Network for Speaker and Language Recognition. (04 2015).
- [20] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. FaceNet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 815–823. <https://doi.org/10.1109/CVPR.2015.7298682>
- [21] Ján Suchal and Pavol Návrat. 2010. Full Text Search Engine as Scalable k-Nearest Neighbor Recommendation System. In *Artificial Intelligence in Theory and Practice III*. Springer Berlin Heidelberg, 165–173.
- [22] Jing Wang, Jingdong Wang, Gang Zeng, Zhuowen Tu, Rui Gan, and Shipeng Li. 2012. Scalable k-nn graph construction for visual descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition*. 1106–1113.
- [23] Jianguo Wang, Xiaomeng Yi, Rentong Guo, Hai Jin, Peng Xu, Shengjun Li, Xianguy Wang, Xiangzhou Guo, Chengming Li, Xiaohai Xu, Kun Yu, Yuxing Yuan, Yinghao Zou, Jiquan Long, Yudong Cai, Zhenxiang Li, Zhifeng Zhang, Yihua Mo, Jun Gu, Ruiyi Jiang, Yi Wei, and Charles Xie. 2021. *Milvus: A Purpose-Built Vector Data Management System*. 2614–2627. <https://doi.org/10.1145/3448016.3457550>
- [24] Mengzhao Wang, Liang Lv, Xiaoliang Xu, Yuxiang Wang, Qiang Yue, and Jionggang Ni. 2022. Navigable Proximity Graph-Driven Native Hybrid Queries with Structured and Unstructured Constraints. arXiv:2203.13601
- [25] Chuangxian Wei, Bin Wu, Sheng Wang, Renjie Lou, Chaoqun Zhan, Feifei Li, and Yuanzhe Cai. 2020. AnalyticDB-V: a hybrid analytical engine towards query fusion for structured and unstructured data. *Proceedings of the VLDB Endowment* 13 (08 2020), 3152–3165. <https://doi.org/10.14778/3415478.3415541>
- [26] Jason Weston, Samy Bengio, and Nicolas Usunier. 2010. Large Scale Image Annotation: Learning to Rank with Joint Word-Image Embeddings. *Mach. Learn.* 81, 1 (oct 2010), 21–35. <https://doi.org/10.1007/s10994-010-5198-3>
- [27] Peter Yianilos. 1993. Data Structures and Algorithms for Nearest Neighbor Search in General Metric Spaces. *Fourth Annual ACM-SIAM Symposium on Discrete Algorithms* 93. <https://doi.org/10.1145/313559.313789>