# Adaptive Feature-Aware Hashing (AFAH): A Lightweight Data-Driven Hashing Framework for Efficient Image Retrieval

Rizwan Ayazuddin * and Noor Ul Amin

Posted Date: 13 March 2026

doi: 10.20944/preprints202603.0989.v1

Keywords: image retrieval; big data analytics; hashing; feature weighting

 doi:10.20944/preprints202603.0989.v1



*Article*

# Adaptive Feature-Aware Hashing (AFAH): A Lightweight Data-Driven Hashing Framework for Efficient Image Retrieval

**Rizwan Ayazuddin * and Noor Ul Amin**

Taylor's University, Subang Jaya, Malaysia

* Correspondence: rizayazuddin@gmail.com

**Abstract**

In large scale image retrieval and big data analytics it is a big challenge to search similar images from high dimensional data. Mostly used algorithms are Locality Sensitive Hashing and Random Projection Based Hashing. They are widely used for approximate nearest neighbor searching. These two algorithms treat all input features uniformly while they ignore feature importance and class separability. In this research we aim to propose a lightweight hashing framework named Adaptive Feature Aware Hashing which integrates feature weighting prior to projection-based hashing. The algorithm computes data-driven feature weights using variance, between-class separability, and Fisher-style discriminative criteria to enhance discriminative power during hash code generation. We also incorporated multi table and multi probe hashing which enhances discriminative power during hash code generation. For this research we used MNISH dataset for experimental evaluation. We compared the results against a Baseline Locality-Sensitive Hashing (LSH) method using random projections. Our results indicate that The AFAH methods (v1 and v2 Fisher) significantly improved both precision and recall compared to the Baseline LSH, with AFAH v2 Fisher showing the highest precision (0.7557) and AFAH v1 having the highest recall (0.2285).

**Keywords:** image retrieval; big data analytics; hashing; feature weighting

## 1. Introduction

Approximate Nearest Neighbor search is a method which is used to quickly find those items that are similar or close to a query item in a very large dataset [1,2]. It works on by computing the most similar vector by computing distance with all data points. Since digital data is exponentially growing [3–5], hence this technique has become fundamental in computer vision [6–8], data mining [9–11], and large-scale analytics [12–14]. Assuming a simple example of searching 10 million images in dataset where the query is *"search handwritten digit 5"*, to find similar images the normal technique or method is to compare the query with all 10 million images. But high-dimensional feature representations significantly increase computational complexity in similar search tasks. Hashing-based methods like random projection-based Locality-Sensitive Hashing address this challenge by mapping high-dimensional vectors into compact binary codes that preserve similarity relationships [15,16]. However, traditional hashing schemes assume uniform feature importance [17]. In real-world data, especially image datasets, certain features contribute more significantly to discriminative power than others. Ignoring this imbalance reduces retrieval performance and increases collision inefficiency [18,19].

In this research we combine the idea of feature weighting, projection hashing and multi table hashing and propose AFAH (Adaptive Feature-Aware Hashing) algorithm. In normal feature hashing pixels/features are treated all equally. In images normally, edges, shapes and strokes are more important than background pixels. In AFAH images with similar codes are stored in the same bucket. So, when the search is made, the system only checks a few buckets instead of millions of

 doi:10.20944/preprints202603.0989.v1



images. The algorithm does this by giving higher weight to important features, so similar images are more likely to end up in the same bucket. Figure 1 shows some real-world use cases of the algorithm's application.

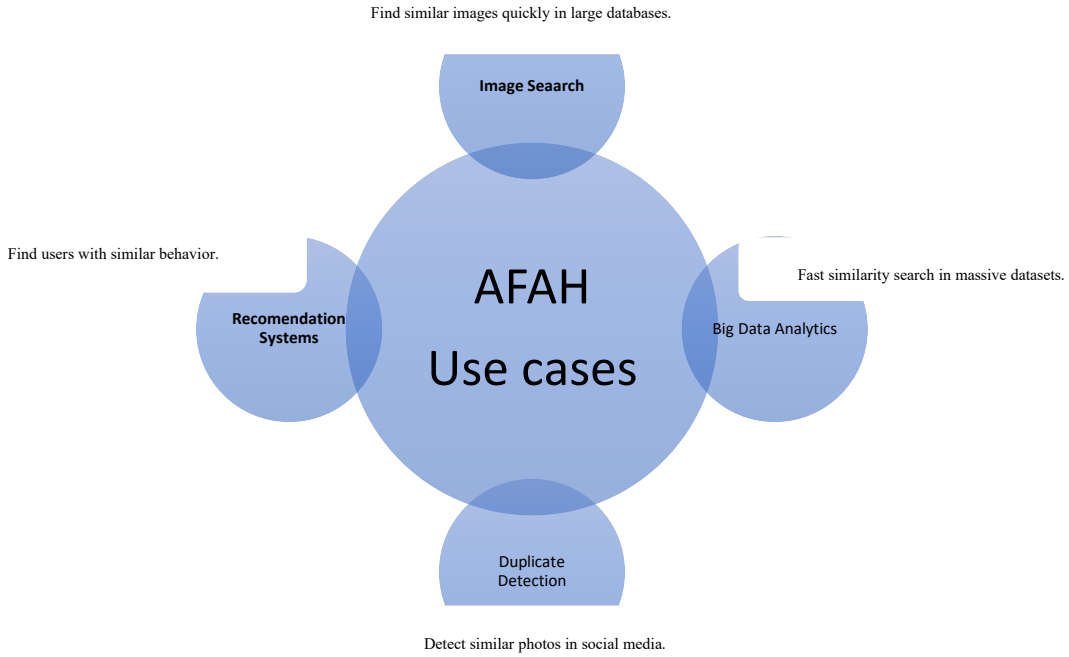



**Figure 1.** Use cases of Adaptive Feature-Aware Hashing.

AFAH helps computers quickly find similar items in huge datasets by converting data into compact binary codes while giving more importance to meaningful features. The proposed algorithm is built upon classical random projection hashing and locality-sensitive hashing methods.

Our core contributions are:

- We proposed Adaptive Feature-Aware Hashing (AFAH) framework.
- We introduced Fisher-based feature weighting for projection hashing.
- We integrated multi-table and multi-probe strategies.
- We provided comprehensive ablation analysis on hashing parameters.
- We demonstrated practical scalability with minimal computational overhead.

## 2. Literature Review

In computer vision and big data analytics efficient image retrieval from large-scale datasets is a very important and fundamental problem. Traditional exact nearest neighbor search becomes computationally expensive when dealing with high-dimensional image features and massive image repositories. Hashing-based approximate nearest neighbors are mostly and widely used algorithms adopted due to their ability to convert high-dimensional data into compact binary codes that enable fast similarity search. Locality-Sensitive Hashing introduced probabilistic guarantees for similarity preservation under random projections. It is one of the most widely used methods for large-scale image retrieval because of its theoretical guarantees and computational efficiency.

The integration of deep learning with hashing methods to improve retrieval performance has also been studied recently by [19]. The authors produced a CNN combined with an attention mechanism to enhance locality-sensitive hashing for image retrieval. Similarly, [20] introduced a deep locality-sensitive hashing architecture that integrates convolutional neural networks with hashing mechanisms. The author's findings show that the adopted method enhanced both retrieval accuracy and efficiency.

 doi:10.20944/preprints202603.0989.v1



With the growing complexity of image datasets, several works have focused on improving hashing robustness for degraded or noisy images. Like the work of [21], where the authors used a self-supervised deep hashing approach designed specifically for retrieving degraded images in large-scale datasets. The hashing approach has also been used in biometric systems by [22]. Locality-sensitive hashing has also been widely used in visual search systems and biometric security applications like [23] introduced a visual search system powered by locality-sensitive hashing to perform efficient large-scale image search. In another study, [24] investigated the reversibility and security aspects of locality-sensitive hashing-based biometric template protection systems.

Although deep hashing methods have significantly improved retrieval accuracy [25–28], they often require extensive training data, large computational resources, and complex optimization procedures [29]. Training deep neural networks for feature extraction and hashing can be computationally expensive, particularly for large-scale systems or resource-constrained environments [30,31]. Deploying these deep models in real-time retrieval applications may introduce latency and memory overhead. Lightweight hashing techniques that do not require deep neural network training remain an important research direction for scalable similarity search.

In this context, classical locality-sensitive hashing remains attractive due to its simplicity, theoretical guarantees, and fast indexing capabilities. However, traditional projection-based hashing methods treat all features equally and do not exploit feature importance or discriminative characteristics within the data. This limitation can reduce retrieval effectiveness, particularly when certain dimensions contain more informative patterns.

In this research we address this limitation by proposing Adaptive Feature-Aware Hashing (AFAH) framework. Our proposed algorithm enhances classical hashing methods by incorporating data-driven feature weighting before hash code generation. Unlike deep hashing approaches, the proposed method does not require neural network training and instead utilizes statistical measures such as variance and Fisher discriminative criteria to adaptively emphasize informative features.

## 3. Methodology

In this research we introduce a lightweight hashing-based framework for efficient similarity search in large image datasets [32–34]. Our methodology focuses on improving traditional hashing techniques by incorporating adaptive feature weighting before hash code generation. Compared to recent deep learning approaches that require computationally expensive training procedures [35], our proposed method relies on statistical feature analysis combined with random projection hashing to produce compact binary codes that preserve similarity relationships among images [36–38].

The proposed AFAH framework consists of five main stages: feature representation, adaptive feature weighting, hash code generation, hash table indexing, and approximate nearest neighbor retrieval. Figure 2 shows our methodology pipeline and Algorithms 1 and 2 describe the two main operations of our proposed method:

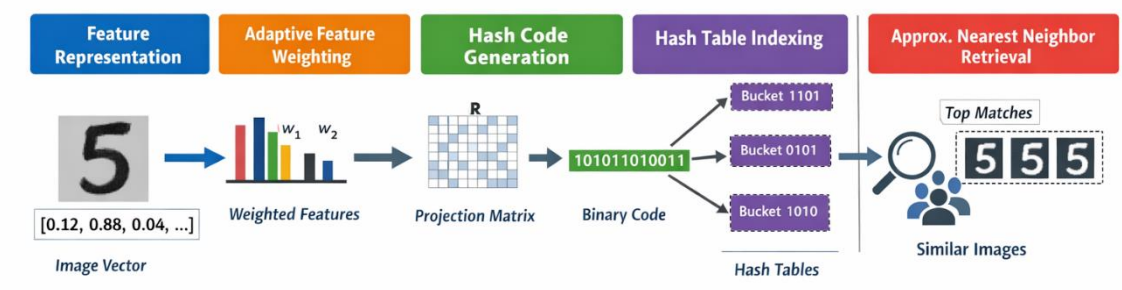



**Figure 2.** Five Step Adaptive Hashing Pipeline.

First, images are represented as numerical feature vectors. In this study, the MNIST handwritten digit dataset is used for experimentation. Each image in the dataset has a resolution of 28 × 28 pixels. The pixel values are normalized and flattened into a vector of length 784.

 doi:10.20944/preprints202603.0989.v1



The mathematical operations can be performed on this vector representation. For example, an image of a handwritten digit can be represented as a feature vector:

[0.12, 0.03, 0.88, 0.76, … , 0.45]

where each element corresponds to a pixel intensity value.

The second stage involves adaptive feature weighting. In this step we compute feature importance weights using statistical measures derived from the dataset. We used variance-based weighting and Fisher discriminative weighting to determine how important each feature dimension is. Features with higher variance or stronger class separation are assigned to larger weights, while less informative features receive lower weights. The original feature vector is then multiplied element-wise with the computed weight vector, producing a weighted feature representation that emphasizes informative dimensions.

In the third stage, weighted feature vectors are transformed into compact binary hash codes using random projection hashing. A random projection matrix is generated, and the weighted feature vector is projected into a lower-dimensional space. The sign of each projection value determines the binary hash bit. If the projected value is positive, the bit is set to 1; otherwise, it is set to 0. This process produces a binary code representing the image. For example, a feature vector may be converted into a hash code such as:

1010100110010110

These binary codes are significantly smaller than the original feature vectors and allow efficient indexing and comparison.

The fourth stage constructs multiple hash tables to improve retrieval performance. Each hash table uses a different random projection matrix to generate independent hash codes (keys for storing image indices). Similar images are more likely to collide in at least one of these hash tables.

Finally, approximate nearest neighbor search is performed using the hash tables. When a query image is provided, its feature vector is processed using the same weighting and hashing steps to generate a binary code. Instead of comparing the query with every image in the dataset, the system searches only the relevant hash buckets containing similar codes. A multi-probe strategy is also used to examine nearby buckets with small Hamming distance to improve recall. The retrieved candidate images are then ranked using Euclidean distance to identify the most similar results.

For example, when a query image of the handwritten digit “5” is provided, the algorithm generates its binary hash code and retrieves images stored in matching or nearby buckets. These candidates are then compared using distance measures, allowing the system to quickly return the most similar digit images from the dataset.

**Algorithm 1** Adaptive Feature Weight Computation

**Require:** Dataset $X$, labels $y$
**Ensure:** Feature weight vector $w$
Compute global mean $\mu$
**for** each class $c$ **do**
    Compute class mean $\mu_c$
    Compute class variance $\sigma_c$
**end for**
Compute between-class variance $B$
Compute within-class variance $W$
Compute feature weights:

$$w = \frac{B}{W + \epsilon}$$

Normalize $w$
**return** $w$

 doi:10.20944/preprints202603.0989.v1



**Algorithm 2** AFAH Hash Code Generation

**Require:** Weighted data $X_w$, number of bits $b$
**Ensure:** Binary hash codes $H$
1: Generate random projection matrix $R \in R^{d \times b}$
2: Compute projections:

$$P = X_w \times R$$

3: Apply sign function to obtain binary codes:

$$H = sign(P)$$

4: **return** $H$

## 4. Experimental Results

We conducted experiments using the MNIST dataset where each image is represented by a 784-dimensional feature vector corresponding to pixel intensities. The dataset was divided into an indexing set containing 59,500 images and a query set containing 10,500 images. We evaluated using the metrics included Precision@K, Recall@K, query processing time, and the number of retrieved candidate images.

### *4.1. Qualitative Retrieval Results*

The qualitative retrieval examples in the results showed and demonstrated the effectiveness of the proposed hashing framework in identifying visually similar images. In Figure 3(a) we can see the retrieval results using the proposed AFAH v2 (Fisher weighting) method for a query image representing the digit "9". The retrieved top-10 results are visually similar and correctly correspond to the same digit class. In Figure 3(b) we can see the baseline Random Projection Locality-Sensitive Hashing (LSH) method for the same query image. Although the baseline method also retrieves the correct matches, the retrieved images show slightly greater variability compared to the proposed method. This indicates that adaptive feature weighting helps improve similar preservation in the hashing process.


(a) AFAH Fisher Retrieval (top-10)


(b) Baseline LSH Retrieval (top-10)

**Figure 3.** AFAH Fisher retrieval (top-10) vs Baseline LSH Retrieval.

### *4.2. Impact of Multi-Probe Radius*

The influence of the multi-probe search radius on retrieval performance is depicted in Figure 4. Increasing the Hamming search radius allows the system to explore neighboring hash buckets,

 doi:10.20944/preprints202603.0989.v1



thereby retrieving additional candidate images. As shown in Figure 3, increasing the radius from 0 to 2 leads to a significant improvement in retrieval performance. Precision@K increases from approximately 0.42 to 0.88, while Recall@K improves from approximately 0.12 to 0.27. This means and indicates that exploring neighboring hash buckets substantially increases the likelihood of retrieving relevant images. However, larger radii may also increase computational cost. This suggest that moderate radius values provide a good balance between retrieval quality and efficiency [39,40].

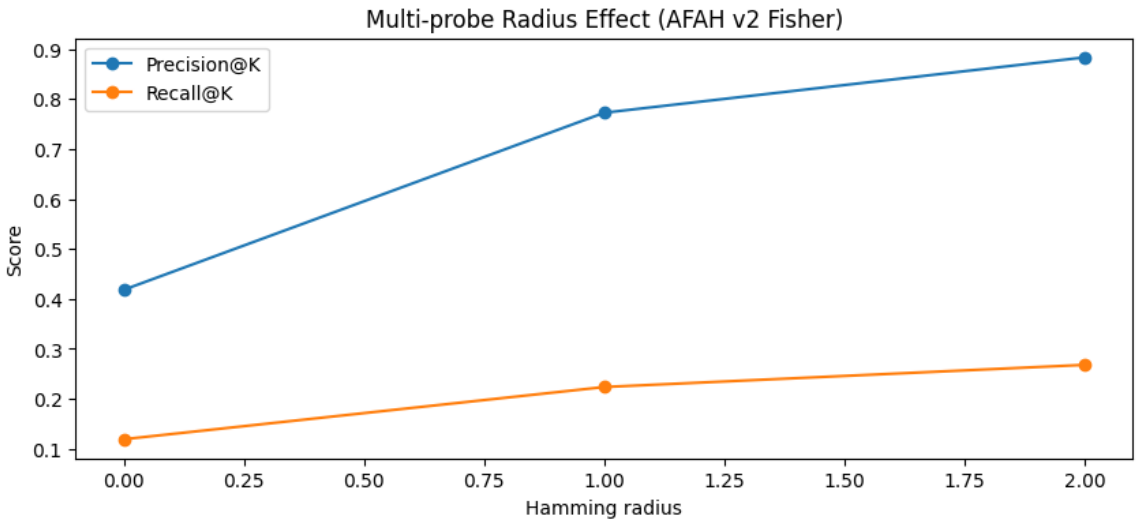
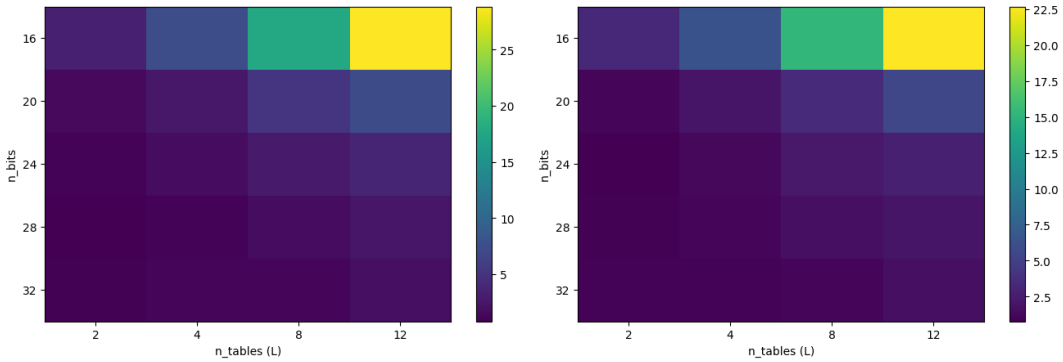



**Figure 4.** Multi Probe Radius Effect (AFAH Fisher).

*4.3. Hashing Parameter Sensitivity*

The performance of hashing methods strongly depends on two key parameters: the number of hash bits (b) and the number of hash tables (L). To analyze their effects, extensive ablation experiments were conducted.

Figure 5(a) and Figure 5(b) present heatmaps of query processing time for AFAH and baseline LSH respectively across different parameter configurations. The results show that increasing the number of hash tables increases query time because more candidate buckets must be searched. Conversely, increasing the number of hash bits generally reduces the query time because larger hash spaces reduce bucket collisions.




(**a**) AFAH Query Time (ms) Heatmap (**b**) Baseline Query Time (ms) Heatmap

**Figure 5.** Query Time Heatmap.

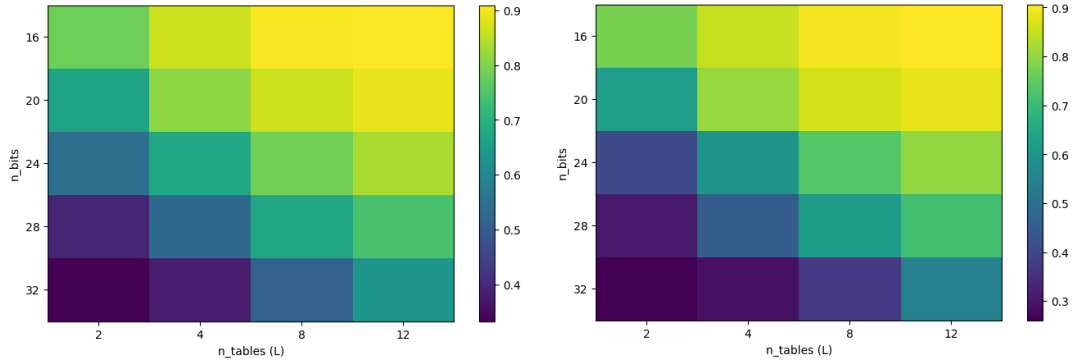
*4.4. Precision and Recall Performance Analysis*

 doi:10.20944/preprints202603.0989.v1



The precision performance across different parameter configurations is shown in Figure 6(a) for AFAH and Figure 6(b) for the baseline LSH method. Both methods achieve higher precision when fewer hash bits are used and when the number of hash tables increases.

For example, when using 16 hash bits and 12 hash tables, both methods achieve precision values exceeding 0.90. However, AFAH consistently demonstrates slightly higher precision across most parameter configurations. We can say that adaptive feature weighting helps preserve similarity relationships during the hashing process.




(**a**) AFAH Precision@K Heatmap (**b**) Baseline LSH Precision@K Heatmap

**Figure 6.** AFAH vs Baseline Precision Heatmaps.

Recall performance also of the baseline LSH method is compared with AFAH in Figure 7(a) and (b) . Similar trends are observed in which smaller hash dimensions and larger numbers of hash tables improve recall performance. Higher recall values indicate that the system retrieves a greater proportion of relevant images. We can see that increasing the number of tables increases the probability that similar images collide in at least one hash table. Consequently, multi-table hashing significantly improves retrieval coverage.

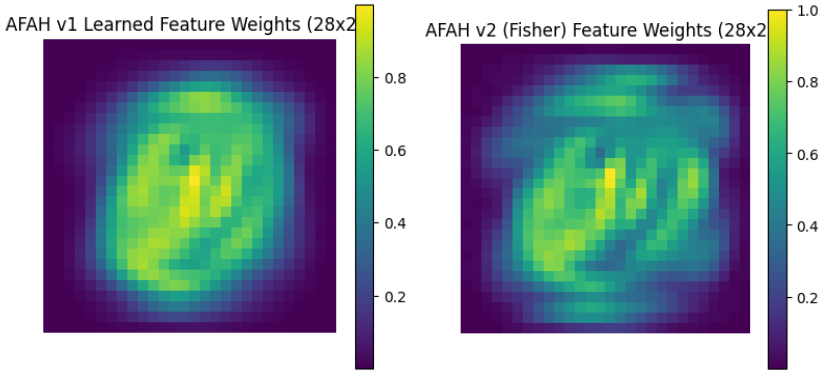



**Figure 7.** Feature Weight Visualization.

*4.5. Feature Weight Visualization*

The feature importance weights learned by the Fisher-based AFAH method is visualized in Figure 7. The heatmap shows that the algorithm assigns larger weights to central image regions corresponding to the digit strokes, while background pixels receive lower weights. This confirms that the adaptive weighting strategy successfully identifies informative features within the image representation.

The distribution of candidate images retrieved per query is shown in Figure 7. The AFAH variants retrieve a larger number of candidate images compared to the baseline LSH method. While this slightly increases query processing time, it significantly improves retrieval accuracy because more relevant images are included in the candidate set.

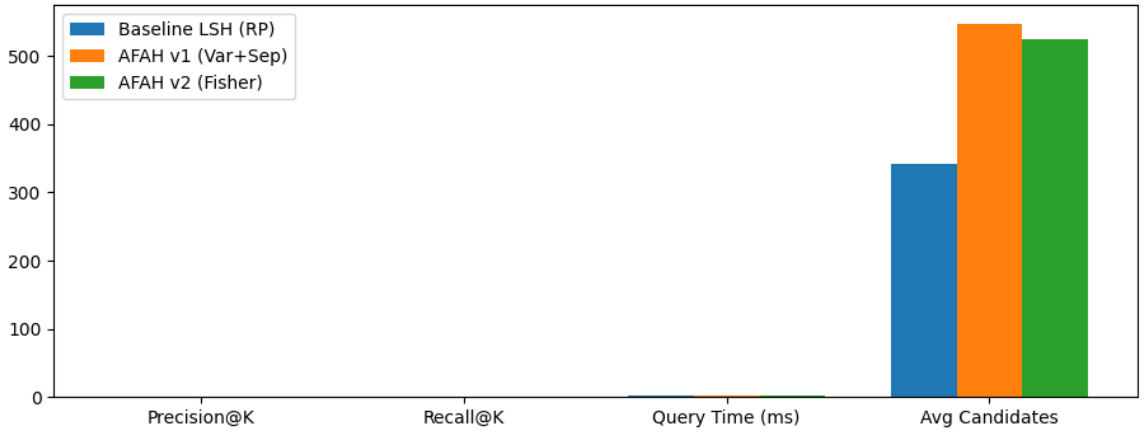
 doi:10.20944/preprints202603.0989.v1



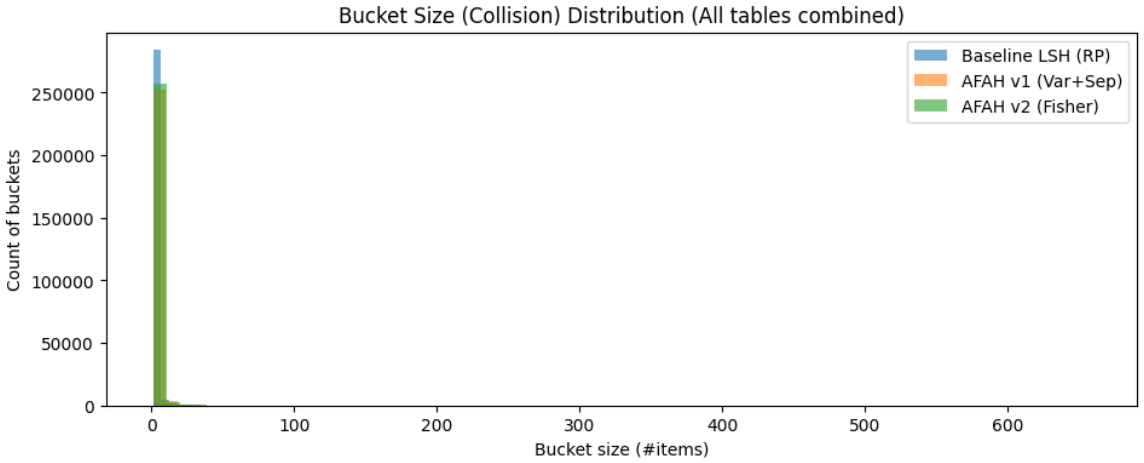
The Fisher-based AFAH method retrieves approximately 525 candidate images per query on average, compared to approximately 342 candidates for baseline LSH. This broader candidate pool contributes to improved precision and recall.

*4.6. Overall Performance Comparison*

A direct comparison of retrieval performance is presented in Figure 8(a, b and c). The baseline LSH method achieves a Precision@50 of 0.7127 and Recall@50 of 0.2168 with an average query time of 2.05 ms. In contrast, the AFAH v1 method achieves a Precision@50 of 0.7553 and Recall@50 of 0.2285 with a query time of 2.83 ms. Similarly, the Fisher-weighted AFAH v2 method achieves Precision@50 of 0.7557 and Recall@50 of 0.2265 with a query time of 2.71 ms.



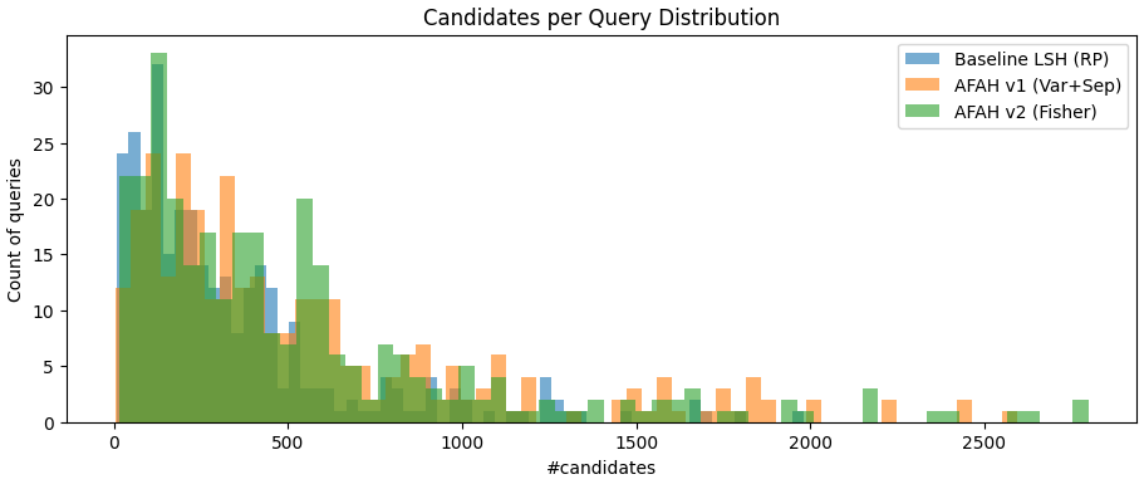

(a) AFAH vs Baseline (K-50)




(b) Bucket Size (Collision) Distribution

 doi:10.20944/preprints202603.0989.v1






(c) Candidates per Query Distribution

**Figure 8.** Overall Performance Comparison Bar charts.

### *4.7. Alpha Parameter Sensitivity*

We can see in Figure 9 the impact of the weighting parameter $\alpha$ used in AFAH v1, which controls the balance between variance-based and separability-based feature weighting. The results show that $\alpha = 0.0$ provides the best performance, indicating that separability-based weighting contributes more effectively to discriminative hashing than variance-based weighting alone.

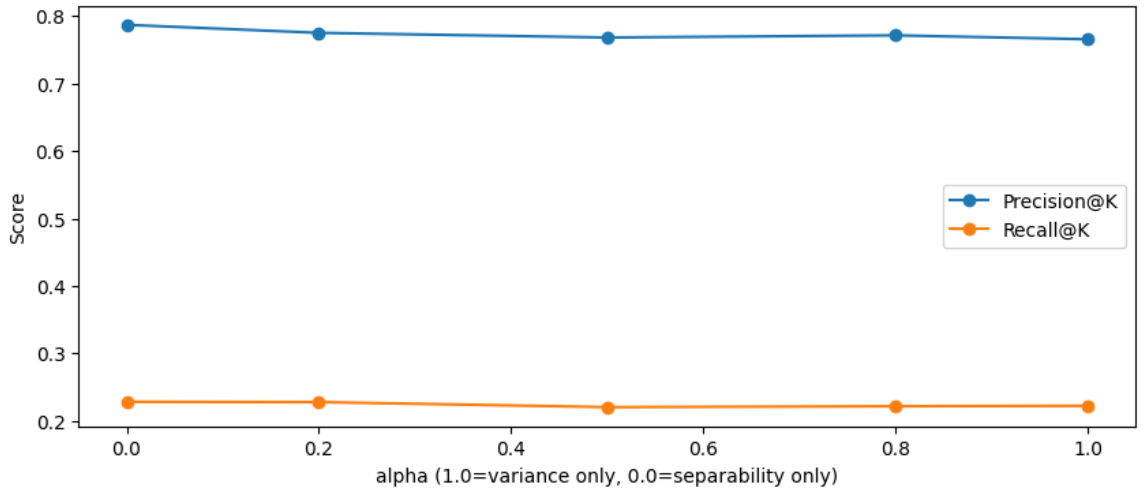



**Figure 9.** AFAH Alpha Sweep (Var vs Seprability Mix).

## 6. Discussion

From the results we can see that lightweight statistical feature adaptation can enhance classical hashing frameworks without deep learning overhead. This makes AFAH suitable for big data systems, embedded devices, and memory-constrained retrieval applications. We summarize our results as below:

(a) Adaptive feature weighting improves similarity preservation in hashing-based retrieval systems.
(b) Multi-table hashing significantly increases recall by increasing collision opportunities among similar images.
(c) Multi-probe search improves retrieval accuracy by exploring neighboring hash buckets.

 doi:10.20944/preprints202603.0989.v1



(d) The proposed AFAH method achieves higher precision and recall compared to traditional LSH while maintaining comparable query efficiency.

To analyze the impact of the feature weighting parameter $\alpha$ in the proposed AFAH v1 framework, an ablation experiment was conducted by varying the $\alpha$ value between 0.0 and 1.0. The parameter $\alpha$ controls the balance between variance-based weighting and class separability-based weighting in the feature importance computation stage. Specifically, $\alpha = 1$ emphasizes variance-based weighting, whereas $\alpha = 0$ relies entirely on separability-based weighting. The results of this experiment are summarized in Table 1, which reports Precision@50, Recall@50, average query time, and the average number of candidate images retrieved per query.

**Table 1.** Effect of $\alpha$ on Retrieval Performance in AFAH.

| $\alpha$ Value | Precision@50 | Recall@50 | Query Time (ms) | Avg. Candidates |
|---|---|---|---|---|
| **0.0** | **0.7868** | **0.2283** | 2.96 | 615.4 |
| **0.2** | 0.7749 | 0.2279 | 2.81 | 573.5 |
| **0.5** | 0.7680 | 0.2204 | 2.82 | 534.3 |
| **0.8** | 0.7712 | 0.2217 | 2.71 | 519.2 |
| **1.0** | 0.7655 | 0.2223 | **2.52** | 501.9 |

The results indicate that $\alpha = 0.0$ achieves the best overall retrieval performance, obtaining the highest Precision@50 and Recall@50 values. Thus separability-based feature weighting contributes more effectively to discriminative hashing compared to variance-based weighting alone. As we increased the $\alpha$, the number of candidate images decreases slightly, which reduces query time but also results in a minor decrease in retrieval performance.

## 8. Conclusion

In this research, we proposed a simple method called Adaptive Feature-Aware Hashing (AFAH) for fast image retrieval. The idea of this method is to give more importance to useful features in the data before generating hash codes. Two versions of the method were studied: AFAH v1, which uses variance and class separability, and AFAH v2, which uses Fisher-style feature weighting.

Experiments were conducted using the MNIST dataset to test the effectiveness of the proposed method. The results show that AFAH improves retrieval performance compared with the traditional Locality-Sensitive Hashing (LSH) method. In particular, AFAH achieves higher Precision@50 and Recall@50 values, which means it can find more correct and relevant images during retrieval.

Although AFAH requires slightly more query time and retrieves more candidate images than the baseline LSH method, the improvement in retrieval accuracy makes this trade-off reasonable for many real-world applications. The ablation experiments also showed that the proposed method performs well under different parameter settings and benefits from multi-probe search strategies. In the future, this method can be extended to larger datasets and more complex image features. It can also be combined with deep learning feature extractors to further improve retrieval performance.

## References

1. Aumüller, M., & Ceccarello, M. (2023). Recent Approaches and Trends in Approximate Nearest Neighbor Search, with Remarks on Benchmarking. *IEEE Data Eng. Bull.*, *47*(3), 89-105.
2. Peng, Y., Choi, B., Chan, T. N., Yang, J., & Xu, J. (2023). Efficient approximate nearest neighbor search in multi-dimensional databases. *Proceedings of the ACM on Management of Data*, *1*(1), 1-27.
3. Shah, I. A., Jhanjhi, N. Z., & Ray, S. K. (2024). Artificial intelligence applications in the context of the security framework for the logistics industry. In *Advances in Explainable AI Applications for Smart Cities* (pp. 297-316). IGI Global Scientific Publishing.
4. Hossain, M. A., Ray, S. K., & Lota, J. (2020). SmartDR: A device-to-device communication for post-disaster recovery. *Journal of Network and Computer Applications*, *171*, 102813.

 doi:10.20944/preprints202603.0989.v1




5. Ray, S. K., Pawlikowski, K., & Sirisena, H. (2008, October). A fast MAC-layer handover for an IEEE 802.16 e-based WMAN. In *International Conference on Access Networks* (pp. 102-117). Berlin, Heidelberg: Springer Berlin Heidelberg.
6. Ashfaq, F., Jhanjhi, N. Z., Khan, N. A., Muzafar, S., & Das, S. R. (2024, March). Crimescene2graph: Generating scene graphs from crime scene descriptions using bert ner. In *International Conference on Computational Intelligence in Pattern Recognition* (pp. 183-201). Singapore: Springer Nature Singapore.
7. JingXuan, C., Tayyab, M., Muzammal, S. M., Jhanjhi, N. Z., Ray, S. K., & Ashfaq, F. (2024, November). Integrating AI with robotic process automation (RPA): advancing intelligent automation systems. In *2024 IEEE 29th Asia Pacific Conference on Communications (APCC)* (pp. 259-265). IEEE.
8. Samaras, V., Daskapan, S., Ahmad, R., & Ray, S. K. (2014, November). An enterprise security architecture for accessing SaaS cloud services with BYOD. In *2014 Australasian Telecommunication Networks and Applications Conference (ATNAC)* (pp. 129-134). IEEE.
9. Javed, D., Jhanjhi, N. Z., Ashfaq, F., Khan, N. A., Das, S. R., & Singh, S. (2024, July). Student performance analysis to identify the students at risk of failure. In *2024 International Conference on Emerging Trends in Networks and Computer Communications (ETNCC)* (pp. 1-6). IEEE.
10. Mishra, S. K., Mishra, S., Alsayat, A., Jhanjhi, N. Z., Humayun, M., Sahoo, K. S., & Luhach, A. K. (2020). Energy-aware task allocation for multi-cloud networks. *IEEE Access*, *8*, 178825-178834.
11. Ali, S., Hafeez, Y., Jhanjhi, N. Z., Humayun, M., Imran, M., Nayyar, A., ... & Ra, I. H. (2020). Towards pattern-based change verification framework for cloud-enabled healthcare component-based. *Ieee Access*, *8*, 148007-148020.
12. Zaman, N., Low, T. J., & Alghamdi, T. (2014, February). Energy efficient routing protocol for wireless sensor network. In *16th international conference on advanced communication technology* (pp. 808-814). IEEE.
13. Ray, S. K., Sirisena, H., & Deka, D. (2013, October). LTE-Advanced handover: An orientation matching-based fast and reliable approach. In *38th annual IEEE conference on local computer networks* (pp. 280-283). IEEE.
14. Jabeen, T., Jabeen, I., Ashraf, H., Ullah, A., Jhanjhi, N. Z., Ghoniem, R. M., & Ray, S. K. (2023). Smart wireless sensor technology for healthcare monitoring system using cognitive radio networks. *Sensors*, *23*(13), 6104.
15. Wang, X., Valov, I., & Li, H. (2024). Random Projection-Based Locality-Sensitive Hashing in a Memristor Crossbar Array with Stochasticity for Sparse Self-Attention-Based Transformer. *Advanced Electronic Materials*, *10*(10), 2300850.
16. Yang, M., Dai, P., Yin, Y., Wang, D., Wang, Y., & Huang, H. (2025). Predicting and optimizing pure electric vehicle road noise via a locality-sensitive hashing transformer and interval analysis. *ISA transactions*, *157*, 556-572.
17. Guo, Y., Ma, C., & Li, H. (2025). Temporal Dynamics Makes Memristive Physical Unclonable Functions More Secure and Ultra-Lightweight. *Advanced Electronic Materials*, *11*(20), e00335.
18. Guo, Y., Ma, C., & Li, H. (2025). Temporal Dynamics Makes Memristive Physical Unclonable Functions More Secure and Ultra-Lightweight. *Advanced Electronic Materials*, *11*(20), e00335.
19. Luo, Y., Li, W., Ma, X., & Zhang, K. (2022). Image retrieval algorithm based on locality-sensitive hash using convolutional neural network and attention mechanism. *Information*, *13*(10), 446.
20. Mahmoud, A. (2022). A deep locality-sensitive hashing approach for achieving optimal image retrieval satisfaction. *International Journal of Electrical and Computer Engineering (IJECE)*.
21. Xiang, L., Hu, H., Li, Q., Yu, H., & Shen, X. (2025). Self-Supervised Locality-Sensitive Deep Hashing for the Robust Retrieval of Degraded Images. *IEEE Transactions on Information Forensics and Security*, *20*, 1582-1596.
22. Alshahrani, A. A., & Jaha, E. S. (2023). Locality-sensitive hashing of soft biometrics for efficient face image database search and retrieval. *Electronics*, *12*(6), 1360.
23. Foping, F., Tchagna, A., & Mih, T. (2022). A visual search system powered by locality sensitive hashing. *Journal of Computer Vision and Pattern Recognition*, *1*(1), 1-10.
24. Paik, S., Hwang, C., Kim, S., & Seo, J. H. (2025). On the Reversibility of Locality-Sensitive Hashing-based Biometric Template Protections. *IEEE Transactions on Dependable and Secure Computing*.

 doi:10.20944/preprints202603.0989.v1




25. Prasetyo, H., Prayuda, A. W. H., Hsia, C. H., Rohman, M. A. F., Akbar, H. M., & Septiano, A. K. (2025, September). Batik Image Retrieval Using Siamese Networks and Locality Sensitive Hashing. In *2025 International Conference on Artificial Intelligence's Future Implementations (ICAIFI)* (pp. 187-192). IEEE.
26. Ray, S. K., Sirisena, H., & Deka, D. (2013, October). LTE-Advanced handover: An orientation matching-based fast and reliable approach. In *38th annual IEEE conference on local computer networks* (pp. 280-283). IEEE.
27. Shah, I. A., Jhanjhi, N. Z., & Laraib, A. (2023). Cybersecurity and blockchain usage in contemporary business. In *Handbook of Research on Cybersecurity Issues and Challenges for Business and FinTech Applications* (pp. 49-64). IGI Global Scientific Publishing.
28. Jabeen, T., Jabeen, I., Ashraf, H., Ullah, A., Jhanjhi, N. Z., Ghoniem, R. M., & Ray, S. K. (2023). Smart wireless sensor technology for healthcare monitoring system using cognitive radio networks. *Sensors*, *23*(13), 6104.
29. Azeem, M., Ullah, A., Ashraf, H., Jhanjhi, N. Z., Humayun, M., Aljahdali, S., & Tabbakh, T. A. (2021). Fog-oriented secure and lightweight data aggregation in iomt. *IEEE Access*, *9*, 111072-111082.
30. Lee, S., Abdullah, A., & Jhanjhi, N. Z. (2020). A review on honeypot-based botnet detection models for smart factory. *International Journal of Advanced Computer Science and Applications*, *11*(6).
31. Kaur, N., Verma, S., Jhanjhi, N. Z., Singh, S., Ghoniem, R. M., & Ray, S. K. (2023). Enhanced QoS-aware routing protocol for delay sensitive data in Wireless Body Area Networks. *IEEE Access*, *11*, 106000-106012.
32. Ashfaq, F., Jhanjhi, N. Z., Khan, N. A., Jia, C., Ihsan, U., & Junfithrana, A. P. (2025). Latent Structural Discovery in Clinical Texts via Transformer-Based Embeddings and Token Graphs. *Engineering Proceedings*, *107*(1), 73.
33. Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., & Xiao, J. (2015). Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*.
34. Kadam, S. S., Adamuthe, A. C., & Patil, A. B. (2020). CNN model for image classification on MNIST and fashion-MNIST dataset. *Journal of scientific research*, *64*(2), 374-384.
35. Shrestha, A., & Mahmood, A. (2019). Review of deep learning algorithms and architectures. *IEEE access*, *7*, 53040-53065.
36. Xiong, L., Xiong, C., Li, Y., Tang, K. F., Liu, J., Bennett, P., ... & Overwijk, A. (2020). Approximate nearest neighbor negative contrastive learning for dense text retrieval. *arXiv preprint arXiv:2007.00808*.
37. Indyk, P., & Motwani, R. (1998, May). Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing* (pp. 604-613).
38. Yao, Y. Y. (2006). Neighborhood systems and approximate retrieval. *Information Sciences*, *176*(23), 3431-3452.
39. Liao, H., Chen, H., Yin, T., Yuan, Z., Horng, S. J., & Li, T. (2025). A general adaptive unsupervised feature selection with auto-weighting. *Neural Networks*, *181*, 106840.
40. Rasool, M., Kassoul, K., & Belhaouari, S. B. (2025). Gaussian Kernel-Based LSH for High-Dimensional Similarity Search. *IEEE Open Journal of the Computer Society*.