DOI: https://doi.org/10.15276/hait.8.2025.2
UDC 004.85

Partitioning the data space before applying hashing using clustering algorithms

Sergey A. Subbotin[1)]
ORCID: https://orcid.org/0000-0001-5814-8268; subbotin@zntu.edu.ua. Scopus Author ID: 7006531104
Fedir A. Shmalko[1)]
ORCID: https://orcid.org/0009-0006-0651-6448; shmalko.fedir@zp.edu.ua
[1)] National University "Zaporizhzhia Polytechnic", 64, Zhukovskogo Str. Zaporizhzhia, 69011, Ukraine

ABSTRACT

This research presents a locality-sensitive hashing framework that enhances approximate nearest neighbor search efficiency by integrating adaptive encoding trees and BERT-based clusterization. The proposed method optimizes data space partitioning before applying hashing, improving retrieval accuracy while reducing computational complexity. First, multimodal data, such as images and textual descriptions, are transformed into a unified semantic space using pre-trained bidirectional encoder representations from transformers embeddings. this ensures cross-modal consistency and facilitates high-dimensional similarity comparisons. Second, dimensionality reduction techniques like Uniform Manifold Approximation and Projection or t-distributed stochastic neighbor embedding are applied to mitigate the curse of dimensionality while preserving key relationships between data points. Third, an adaptive encoding tree locality-sensitive hashing encoding tree is constructed, dynamically segmenting the data space based on statistical distribution, thereby enabling efficient hierarchical clustering. Each data point is converted into a symbolic representation, allowing fast retrieval using structured hashing. Fourth, locality-sensitive hashing_is applied to the encoded dataset, leveraging p-stable distributions to maintain high search precision while reducing index size. The combination of encoding trees and Locality-Sensitive Hashing enables efficient candidate selection while minimizing search overhead. Experimental evaluations on the CarDD dataset, which includes car damage images and annotations, demonstrate that the proposed method outperforms state-of-the-art approximate nearest neighbor techniques in both indexing efficiency and retrieval accuracy. The results highlight its adaptability to large-scale, high-dimensional, and multimodal datasets, making it suitable for diagnostic models and real-time retrieval tasks.

Keywords: Adaptive encoding tree; bidirectional encoder representations from transformers cauterization; dimensionality reduction; approximate nearest neighbor; multimodal data; root node

***For citation*: Subbotin S. A., Shmalko F. A. "Partitioning the data space before applying hashing using clustering algorithms". *Herald of Advanced Information Technology*. 2025; Vol.8 No.1: 28– . DOI: https://doi.org/10.15276/hait.08.2025.2**

INTRODUCTION

Locality-sensitive hashing effectively means a method that is based on probabilistic dimensionality reduction of data, thus reducing a number of features (columns) in a dataset, preventing a problem called "curse of dimensionality" that hinders the model's ability to learn.

Several limitations complicate the direct application of classical methods to real-world, often high-dimensional, data. In particular, the well-known "curse of dimensionality" leads to efficiency loss: when the space has an excessively large number of dimensions, the probability of collision of "close" points (in terms of the original space metric) in shared hash buckets behaves differently than predicted by theoretical estimates for moderate dimensions. This can increase both search error and computational costs, as achieving adequate performance often requires generating too many projections or complex indexing structures. Furthermore, some traditional approaches, such as "boundary-ordered" hashing, scale poorly when the volume of data grows exponentially. Additionally, classical locality-sensitive hashing(LSH) functions generally do not account for the distribution of data, leading to high indexing costs and insufficient search accuracy.

Another important aspect is the necessity of processing cross-modal data. In many practical tasks, such as anomaly detection or diagnostic model construction, it is necessary to integrate information from different sources, including images, texts, and tabular data. This requires a unified approach, where binary hash codes must preserve high-level semantics regardless of the input modality. Such approaches are referred to as cross-modal or multimodal. Therefore, a critical capability is the dynamic updating of existing hash functions and codes when new data types or categories are introduced into the system. Conventional methods of retraining hash functions "from scratch" on an

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

expanded dataset are computationally expensive and lead to catastrophic forgetting. Thus, the primary direction of this research is the development of a comprehensive strategy in which original codes remain intact while new codes are incrementally constructed, incorporating data clustering that accounts for local density.

This research is especially relevant in the context of rapidly growing multimodal data environments, where traditional similarity search methods struggle with high dimensionality, scalability, and modality alignment. The proposed hybrid methodology of encoding trees combined with LSH directly addresses these issues, making it applicable to domains such as automated diagnostics, multimedia retrieval, and real-time anomaly detection. However, the approach has certain limitations: it requires pre-trained semantic encoders like BERT, assumes relative consistency in data distribution for effective clustering, and may face reduced effectiveness in extremely sparse or adversarial data scenarios. Despite these constraints, the method's adaptability, modularity, and improved search performance make it a promising tool for scalable cross-modal systems.

1. ANALYSIS OF LITERARY DATA

Locality-sensitive hashing (LSH) has been widely explored as a fundamental technique for approximate nearest neighbor (ANN) search in high-dimensional spaces. Numerous studies have investigated its theoretical foundations, optimizations, and applications. However, challenges related to efficiency, scalability, robustness, and applicability to multimodal data remain open problems. This section reviews key contributions from recent research and outlines their limitations, which motivate the need for a different approach.

Jafari et al [1] provide a comprehensive survey of LSH techniques, classifying them based on their hash function families and application domains. They discuss traditional LSH variants such as Hamming-based, Minkowski-based, Angular-based, and Jaccard-based hashing, along with advanced techniques like Multi-Probe LSH and Query Adaptive LSH, which address issues like large index sizes and false positives. The survey also covers distributed LSH implementations, including MapReduce-based frameworks and Apache Spark solutions, which enhance scalability in large-scale data processing.

Although extensive categorization of LSH applications across various domains, including multimedia retrieval, cybersecurity, and biological data analysis can be considered remarkable insights in relation to LSH, critical comparative evaluation of different LSH approaches on real-world benchmarks still remain unavailable. Furthermore, while LSH's advantages over exact nearest neighbor searches are considered, extensive discussion of emerging deep-learning-based alternatives or hybrid approaches that integrate neural embeddings with LSH need to be clarified. This gap suggests the need for further research into hybrid LSH models that optimize performance while maintaining theoretical guarantees.

A more targeted contribution to improving LSH efficiency is presented by McCauley et al in [2], which introduces function inversion techniques to optimize space utilization in ANN data structures. The authors propose a general black-box framework that reduces storage overhead in LSH without significantly increasing query times. Their approach replaces traditional reverse lookup tables with function inversion mechanisms, thereby maintaining retrieval effectiveness while reducing memory usage [3].

This method successfully enhances space efficiency in ANN indexing, particularly in scenarios where high-dimensional datasets require extensive storage. However, the benefits of function inversion are highly parameter-dependent, and its impact varies across different LSH families. Additionally, while space complexity is reduced, query time savings are modest, particularly for lower approximation factors. The approach also assumes that function inversion operations can be efficiently computed, which may not always hold for large-scale, real-world applications. These limitations highlight the need for adaptive LSH models that dynamically balance space, time, and accuracy trade-offs based on the dataset characteristics.

The integration of locality-sensitive filtering (LSF) with LSH in Falconn++, introduced by Pham et al in [4] represents another significant advancement. This approach improves upon traditional LSH-based ANN search by selectively filtering out distant points from hash buckets before querying, reducing false positives and unnecessary computations. Theoretical improvements in Falconn++ lower the exponent ρ governing the space-query efficiency tradeoff, leading to more optimized performance in high-recall settings.

Experimental evaluations on real-world datasets demonstrate that Falconn++ outperforms standard LSH techniques while competing with state-of-the-art graph-based ANN methods such as Hierarchical

Navigable Small World (HNSW). However, the method's effectiveness is highly dependent on threshold selection for filtering, and its generalizability to distance metrics beyond angular similarity remains uncertain. Additionally, while it improves recall-speed tradeoffs, it does not fundamentally alter the structural limitations of LSH when dealing with multimodal data. These factors suggest the need for LSH frameworks that incorporate data distribution-aware filtering mechanisms applicable across multiple similarity measures.

Kapralov et al. [5] investigate LSH's robustness against adversarial attacks, revealing a key vulnerability in traditional hashing schemes. Their study demonstrates that an adversary can systematically construct queries that force LSH to return incorrect results, highlighting a critical weakness in security-sensitive applications. The theoretical findings are supported by empirical evidence, showing that adversarial attacks can significantly degrade LSH's effectiveness, particularly in sparser datasets.

While this work provides valuable insights into LSH's susceptibility to adversarial manipulation, its reliance on isolated points in the dataset limits its applicability to real-world scenarios. Furthermore, while potential defenses such as differential privacy and randomized hashing are mentioned, they are not explored in detail. Given the growing importance of secure and privacy-preserving similarity search, there is a pressing need for LSH-based models that incorporate adversarial resilience while maintaining efficiency.

DeepLSH [6] developed by Remil et al, introduces a deep-learning-based approach that learns hash functions capable of approximating custom similarity measures for crash report deduplication. Using a Siamese neural network, DeepLSH generates locality-sensitive hash codes that maintain theoretical guarantees while adapting to various similarity metrics, such as Jaccard and Cosine similarity. The method demonstrates high recall and efficiency in large-scale crash report retrieval, surpassing traditional LSH and deep hashing baselines.

Despite its advantages, DeepLSH relies on a supervised learning paradigm, requiring extensive labeled training data. This constraint makes it challenging to deploy in scenarios where annotated datasets are scarce. Additionally, while it generalizes across similarity measures, its performance is sensitive to hyperparameter tuning and data distribution. Another concern is the potential impact of adversarial inputs, which is not explicitly addressed in the study. These challenges suggest the need for hybrid models that integrate the adaptability of deep learning with the efficiency and robustness of traditional LSH.

2. THE PURPOSE AND OBJECTIVES OF THE RESEARCH

The purpose of this research is to develop an effective methodology for approximate nearest neighbor search in diagnostic models by integrating encoding trees with locality-sensitive hashing (LSH). The proposed approach ensures the preservation of semantic relationships between data samples of different modalities while optimizing computational efficiency [7].

The objectives of the research include:

1) to encode multimodal diagnostic data into a unified vector space using a pre-trained neural model [8], followed by dimensionality reduction techniques (UMAP and t-SNE) to mitigate the curse of dimensionality while retaining key features;

2) to construct an adaptive encoding tree (ET) using symbolic representations based on local data density, thereby enabling dynamic space partitioning for efficient hierarchical clustering and candidate filtering;

3) to integrate BERT-based clustering with LSH, using p-stable distributions for improved precision, and to evaluate the proposed hybrid method's diagnostic accuracy through comparative analysis with traditional and deep learning-based retrieval techniques.

3. RESEARCH METHODS

The proposed method is based on the usage of previously trained neural network model for coding input data of different modalities in a general vector space. Each sample, which is presented in a textual or visual information form gets transformed into a multidimensional vector format with a subsequent reduction of its dimensionality for the purpose of eliminating "curse of dimensionality" [9]. Vector presentation is performed into a lower space, which allows saving of main semantic connections with objects and increasing calculations efficiency.

Next procedure corresponds with building of an adaptive encoding tree (ET), that dynamically defines threshold values for each coordinate [10]. Based on this, the symbolic representation of each sample is performed, that is then used in tree-like structure building. The uniqueness of the proposed method lies in the combinatory usage of LSH and ET, coupled with BERT clustering of the data,

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

which in total results in a robust and efficient nearest neighbors search in multidimensional space.

The experimental evaluation of the proposed method was backed by the multimodal dataset usage, which included textual descriptions and images. Training of the parameters was accompanied by the cost function, which reduced the likelihood of semantically similar objects diverging from each other. For the purposes of semantic space coherence, the weights regulation process was engineered, which increases the quality of different modality object correspondences.

The algorithm was tested on a subset of samples, which underwent a mechanism of correspondence correctness check. Results analysis included the comparison of search efficiency with the usage of different parameters of encoding structure, such as the amount of levels in the tree, space dimensionalities after projection and locality-sensitive hash-functions characteristics. The usage of combinatory method created a possibility of optimizing the nearest neighbors' identification process in a cross-modal environment, retaining high accuracy considering lowered computation costs.

4. RESEARCH RESULTS

The proposed method of locality-sensitive hashing with an adaptive encoding tree in cross-modal tasks implies the step sequence, the aim of which is the keeping of semantic proximity between samples of different types (text or image information) and the approximate nearest neighbor (ANN) search realization in a big data space with a potentially high dimensionality. At first, the encoding of all samples is performed with the usage of previously trained BERT model. It is implied, that $\{x_i\}_{i=1}^{N}$ is a set of input samples that have different modalities [11].

For each x_i the vector representation gets calculated:

$$e_i = BERT_\theta(x_i).$$

where θ represents BERT model parameters, which give an opportunity to calculate contextual and semantic properties of each sample. Due to the fact that $e_i \in R^d$ often has quite high dimensionality d, there is a need to use dimensionality reduction methods, like t-SNE [12] or UMAP [13], which is motivated by "dimensionality curse" influence reduction and at the same time by the need to save main semantic connections between samples. Formally, the non-linear representation f gets chosen, whose function is to transfer e_i in a lower dimensionality space R^k, where $k \ll d$.

This results in:

$$e_i' = f(e_i) \ \text{with} e_i' \in R^k.$$

This gets followed by adaptive clusterization considering local data density. Instead of uniform distribution of hashes of the whole space, the preliminary definition of local regions (clusters and subspaces) is implied, where data are distributed relatively uniformly [14]. If $X = \{e'_1, e'_2, \dots, e'_N\}$, which is a set of points, gets assigned to R^k space after dimensionality reduction, then algorithms, that are similar to dynamic threshold selection, get used in relation to each coordinate. The idea lies within the necessity of obtaining the breakpoints for each of k spaces for the purpose of approximately identical amount of points getting to each interval. If $\{C_{ij}\}$ denotes the values of j coordinate, that are arranged in ascending order for the specific data subset (or the whole set X), then the M of intervals $\Delta_1, \Delta_2, \dots, \Delta_M$, where $\Delta_m = [b_m, b_{m+1})$, and $b_1 = -\infty\, b_{M+1}$, can be defined, considering that each Δ_m interval occupies approximately N/M points. Quintile search operation for these thresholds formation can be realized with an algorithm QuickSelect, which operates per the average time $O(N)$, and in a general worse scenario – $O(N^2)$, but in practice cases, the linear dependency on N is often observed [15].

Subsequently, the threshold selection for the m interval is denoted like this:

$$b_m = C_j(\lfloor \frac{(m-1)\,N}{M} \rfloor),$$

where $C_j(t)$ corresponds with:

$$c_{ij} = encode(e'_{ij}; \{b_1, b_2, \dots, b_{M+1}\}),$$

where $encode$ returns the symbol (for instance, binary number or an integer from 0 to $M-1$) depending on which thresholds e'_{ij} lies within. By doing that, each vector e'_{ij} is transformed in symbolic representation $C_i = (c_{i1}, c_{i2}, \dots, c_{ik})$, where ach component c_{ij} corresponds with the symbol, that is connected with value range in j space [16].

Based on these symbols the adaptive encoding tree is constructed, which has the root node that occupies the whole data, and then secondary nodes get recursively created, that correspond with the set of points distribution based on a certain criterion. In contrast with the traditional indexation trees (like R-tree, k-d-tree), the encoding tree does not divide the space in a uniform manner, but rather uses symbolic clusterings depending on the data distribution in

each space. As result, each internal node has two or more branches regardless of how the symbol distribution is organized in it. For example, if each space is encoded as 0/1 on the tree's first level that means with the binary division then the root node has 2^k secondary nodes that correspond to every possible bit combination from k coordinates. This basic level can be detalized provided the necessity when the number of points in a certain node is higher than the established maximum, then the clusterization proceeds. When the tree is built, each leaf contains relatively low number of points, while keeping the correspondence between vectors e'_{ij} and their symbolic encodings C_i.

In a practical realization, symbolic representation can be saved in a form of "iSAX"-like codes (indexable Symbolic Aggregate approximation), when an alphabet of 2^r symbols (intervals) emerges in every space, and then the bit representation of each coordinated forms [17]. After the determined ET for each of L independent projections (or for different dimensionality samples) is built, the ANN search can be organized by using both the tree-like structure and mapping locality-sensitive hashes. The idea will imply, that each tree $T^{(l)}, l = 1, \dots, L$, is able to selectively initiate the search in the nodes, that correspond to the most similar symbols for the query q'. Assuming that $ETquery(q', T^{(l)})$ denotes the procedure that returns a little subset of candidates (e. g. each leaf nodes, that coincide with the encoding q' by the bigger part of bits) [18]. Then the candidates from all the trees unify and the final filter based on the real distance is performed for the purpose of nearest neighbor search. The algorithm LSH is usually defined through $(r, c \cdot r, p_1, p_2)$, which corresponds to the locality-sensitive function family. If $|| x - y || \leq r$ gets stated, then the collision probability $h(x) = h(y))$ has to be not less thanp_1, and under the condition of $|| x - y || \geq c\, r$ this probability has to be not more than p_2 [19].One of the most common ways to build such LSH-functions for the Euclidian space is using p-stable distribution (if $p = 2$, the distribution is considered normal).

If the random vector $a \sim N(0, I_a)$ and scalar are generated, which is then followed by the hash-function definition:

$$h(x) = \lfloor \frac{a \cdot x + b}{w} \rfloor,$$

then such function satisfies the mentioned properties of LSH with a certain probability guarantee.

The quality of k-NN search optimization is achieved through taking K of such function and merging it in a vector, which means:

$$H(x) = \big(h_1(x), h_2(x), \dots, h_K(x)\big),$$

additionally, the probability of proximate points being skipped is reduced by using L independent sets of such functions:

$$L(\theta_v, \theta_t) = \sum_{i,j} (S_{ij} \parallel f_v(v_i) - f_t(t_j) \parallel 2 + R(\theta_v, \theta_t)),$$

where S_{ij} is a general indicator of semantic similarity (for instance, it equals 1, if the text describes similar objects, and 0 otherwise) and $R(\theta_v, \theta_t)$ corresponds to the regularization term. However in the scope of the described approach, the cross-modal component is realized somewhat differently: each sample x_i, that already contains a certain data type, is encoded in a vector BERT e_i, and then the corresponding vector image is saved in an encoding tree and LSH-tables. If there are two samples x_i and x_j that are of different types, then BERT transformation causes their transition to e_i and e_j. If they are semantically similar to each other, then the difference $|| e_i - e_j ||$ will be relatively small, therefore, there will be a small amount of divergences in a symbolic representation $encode(e_i')$ and $encode(e_j')$.

Next step revolves around the idea of keeping the encoding stable and considering the local densities. For each dimension j, the algorithm defines $\{b_1^{(j)}, b_2^{(j)}, \dots, b_{M_j+1}^{(j)}\}$ for the purpose of the number of points in each interval being relatively equal. After this definition, the encoding of each e_{ij}' is going to provide a symbol with $\{0,1, \dots, M_j - 1\}$. It is assumed, that the binary scheme is used for the simplicity, which means that $M_j - 2$ is true for every j, then the symbol $c_{ij} \in \{0,1\}$ and the whole point are being encoded by the k-bit string. As mentioned, this corresponds with the simplest alternative, though it can be generalized into greater alphabets. The encoding tree then has 2^k leaf nodes on the first level, each of which corresponds to the unique combination $((c_{i1}, \dots, c_{ik})$. If some leaf contains the excessive amount of points, the further clusterization continues (e. g. for each of the coordinates the operation $\{0,1\} \rightarrow \{00,01,10,11\}$ is performed, and so on) [20]. As a result of this, the tree submerges in those regions, where there are a lot of points, and on the contrary, regions of low density usually get

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

occupied with the more little portions of it, which in total has a positive impact on the search efficiency.

Approximate nearest neighbor query realization, which is described using an example of a new query q, which corresponds to $q' \in R^k q'$, after BERT transformations and dimensionality reduction, is done in the following manner. First of all, it gets encoded $encode(q') \to C_q$. Then all the nodes, that partially coincide with C_q (for example, if one or two divergences are allowed in a binary code, or the node is being searched for, that corresponds to the exact sane code C_q) are selected on the root node. Consequently, if a node that was found has a low quantity of points, the distances between q' and those points are instantly getting calculated. If there are too many of them, the tree allows for a deeper submergence. That is how the rough candidate selection that works under the principle of LCH identification, is performed. When the subset of candidates $S \subset X$ is obtained, the fine calculation of distances $\| q' - e'_i \|$ or cosine similarity is done, in order to sort the candidates and find the nearest ones [21]. If $| S |$ is limited from the above (because the average node size is limited), then such a search is performed considerably faster in comparison to calculation of distances for all N points. The high quality hash construction requires code coincidence metric definition.

If binary encoding is utilized the difference between C_i and C_j can be measured as a Hamming distance [22]

$$dis_H(C_i, C_j) = \sum_{r=1}^{k} 1_{\{c_{ir} \neq c_{jr}\}},$$

where $1_{\{\cdot\}}$ denotes the indicator that equals 1 when bit values are different. Provided the alphabet with M_j for j coordinate is created, the symbol can be turned into a binary code with a fixed length $log_2(M_j j)$ and further use the Hamming distance. In general cases, the code continuity function can be implemented, where the distance between two symbols $encode(q'_{ij})$ and $encode(q''_{ij})$ depends on the difference of their indices $| m_1 - m_2 |$. It is also important to consider that in cross-modal systems, each sample can be composed: for instance, if $x_i = (v_i, t_i)$, then BERT creates a representation in a unified dimension for each input data types, and then the combinatory hashing with LSH and ET is not going to require separate structures for each modality, as they operate in the exact same dimension anyway, whether it is R^d or R^k. If however, it is necessary for different modalities to be managed, multiple ET's can be created: one for images, and the other for the text, whose outputs are going to be combined subsequently.

The construction of the ET is formulated according to this fashion. It is assumed that $X = \{x'_1, x'_2, \dots, x'_N\} \subset Rk$. At first, the "first layer" of clusterization is built, which means that for each $j \in \{1, \dots, k\}$ the $\{b_m^{(j)}\}_{m=1}^{M_j+1}$ gets defined. In simplest case $M_j = 2$ for all j, which corresponds to the binary division, so 2^k cells are obtained (considering that each cell is defined by the set of features $\{c_{ij}\}_{j=1}^{k}$, where $c_{ij} \in \{0,1\}$). For the root node the representation $X \to \{0,1\}$ is formed, which is denoted as $Enc(\cdot)$.

Then 2^k leafs are obtained (on this first level), where each leaf ℓ contains the subset:

$$X_l = \{x'_i \in X \mid Enc(x'_i) = l\}.$$

If $| Xl |$ is of a higher value than a certain threshold $\_size$, then this list is considered to be the root node which in turn causes the repeated clustering to take place in one of the dimensions (or all of them) for those points, that have gotten there.

The selection of dimension j^* happens according to this condition: j^* must minimize the sum of distances in a node during the clusterization:

$$J(j,\tau) = \sum_{x'_i \in Xl: c_{ij} < \tau} \| x'_i - \mu_{left}(j,\tau) \|^2 + \sum_{x'_i \in Xl: c_{ij} \geq \tau} \| x'_i - \mu_{right}(j,\tau) \|^2,$$

where $\mu_{left}(j,\tau)$, $\mu_{right}(j,\tau)$ are centroids of the points that went to the left/right during the clusterization by the threshold τ in a code dimension j. In practice, instead of τ in the binary encoding, there are only 0 or 1 that remain, due to this being an already predetermined division. If the symbol in the dimension j can rest in a range $[0, M_j - 1]$, then the way of dividing this range in two parts is selected. This recursive clusterization happens to the point where $| X_l | \leq \_size$. As a result the hierarchical structure is formed where the point composition of each leaf is limited which provides a fast access during the search. Considering the search algorithm, the entry query $q' \in R^k$ is encoded $Enc(q')$ according to the exact same principle [23]. Following, the search of one or multiple leafs, that coincide with this encoding the most, is performed. If q' has in the dimension j symbol α_j, then the tree instantly comes over to the subnode that corresponds to it on the first level. On the other hand, in case of encoding tree, if the code θ_j has multiple bits, the corresponding way from the root has to be determined. In total, leaf are obtained, whose distance in the dimension to code $Enc(q')$ is not big. From all the points in these leafs, the Euclidian

space to q' be checked in an output reduced dimension [24]. Supposing the model χ^2 or one of the normal distribution is consistent, the probability of condition$\| x'_i - q' \| \leq r$ being satisfied can be assessed. If one has to consider, that when a has a normal distribution, then $(a \cdot x)/|| x ||$ corresponds to the normal random value with a zero average and variance being 1, and it has to do with the p-stability of the distributions for $p = 2$ [25]. Consequently, for $\| x - y \| \leq r$ the $Pr[h(x) = h(y)] \geq p1$is true, and for $\| x - y \| \geq cr$ – the $Pr[h(x) = h(y)] \leq p2$. Unifying L trees and tables contributes to nearest neighbors finding probability growing. Considering that, ET lets realize this approach in a more effective manner, as at first the adaptive space clusterization takes place, with the hashing subsequently covering these local regions. The mathematical likelihood of skipping the nearest neighbor is getting reduced in an exponential fashion with the L growing.

The evaluation of clusterization properties in tree construction can be done by tracking the intra-cluster D_{intra} and inter-cluster D_{inter}.distances:

$$D_{intra} = \frac{1}{| C |} \sum_{x_i, x_j \in C} \| e_i - e_j \|^2, D_{inter} = \frac{1}{| C1 | \cdot | C2 |} \sum_{x_i \in C1, x_j \in C2} \| e_i - e_j \|^2.$$

The objective is based on the need of having low D_{intra} and high D_{inter}. In the context of diagnostic models construction, multidimensional data play a crucial role in cases of combining textual descriptions, signals, images etc. Thanks to the BERT-encoding, all of those get to the shared space R^d with the subsequent reduction to R^k and hashing. This sequence lets conduct ANN search while giving an opportunity to find similar samples quickly. Detailed management of the process can be achieved with utilizing regularizators and sparseness concepts with the purpose of lowering the risk of overfitting while using augmentation methods in case of data deficiency. If sparse regions of a space have to be considered, dynamic threshold adjustment scheme $\{b_m^{(j)}\}$, can be implemented, which introduces a benefit of "stretching" the intervals in such places, where the density is low and the other way around with the aim of avoiding "too rough" coding. From the complexity perspective, the encoding trees construction and thresholds ejection for every k dimension has an order $O(N(d + log\ logM))$, because d correspond to the basic calculation complexity of BERT-embeddings and of dimensionality reduction, while $log\ logM$ has to do either with the repetitive quintile searches or with the recursive clusterization. Space costs for keeping get reduces as well, as instead of full vectors R^d, or even R^k, only symbolic representation C_i gets retained, which can be considerably shorter. During the process of ANN search, the overall complexity decreases which is cause by the fact, that distance check in the original space is performed only for candidate from relevant leafs of the tree, the amount of which is a lot lower than that of N. The two-step can also be considered in this situation: the search in an encoding tree for the faster screening by codes; the accurate comparison of the selected candidates by Euclidean or cosine distance. Cross-modal nature of the approach, as was stated earlier, lies within the concept of BERT being capable of transforming both text and images in a shared vector space, while the LCH and ET are there to perform operations on those vectors. This combination facilitates an integration of different type's objects in a unified index. The enhancement of the approach can be implemented through and integration of penalties for violating the semantic proximity during the training of the system, which means, that if according to the softmax in BERT distribution $p(z)$ the high probability persists, that word or visual fragments belong to the exact same subject, then their embeddings must have similar binary codes.

$$L = \sum_{i,j} (softmax(BERT_\psi(w_i)) - softmax(BERT_\psi(w_i)))^2 + \lambda \| \theta \|^2,$$

where λ is the regularization coefficient. If there is a task to construct a diagnostic model, then cross-modal data can be encoded in a shared space. If NN were found with this cross-modal encoding, then there is a higher change of easier distinguishment of similar cases. Hence, the integration of LSH and ET sequentially does the encoding of data in a semantically enriched space; reduces the dimensionality; adaptively clusterizes the space by statistical properties; creates a symbolic code for each vector using iSAX-like approach; form adaptive tree, where each node is responsible for a certain combination of symbols; searches for candidates in relevant leaf nodes. This methodology contributes to creating a compact data representation, while retaining a high level of semantic relevance and giving an opportunity to perform cross-modal search effectively. Considering that the training criterion function of the whole architecture can be formally represented, a number

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

of components can be foreseen: penalties for the semantics divergence (KL-divergence) between BERT-vectors of related samples; and for interval uniformity violation; and regularizators.

If B are binary codes and F – continuous vectors in R^k, then:

$$L(\theta) = \sum_{i,j} \underbrace{(\| F_i - F_j \|^2 - \alpha S_{ij})^2}_{ensuring\ semantic\ similarity} + \beta \sum_i \underbrace{\| F_i - B_i \|^2}_{quantization} + \gamma \sum_i \underbrace{Balance(j)}_{bits\ balance},$$

where S_{ij} denotes the indicator or measure of semantic proximity (1, if they are similar and 0 otherwise, or if the amount of shared topics is partially similar), $Balance(j)$ corresponds to the measure of quantity divergence from the equality +1 and -1 in j bit [26]. After obtaining B as a means for minimization task $L(\theta)$ accomplishment, the encoding tree can be reconstructed. First, the optimization of continuous vectors F is performed (by the common error backpropagation optimization), then the $B_i = sign(F_i)$ gets calculated with the subsequent determination of thresholds for each dimension, which gets followed by the ET construction and LSH utilization.

In summary, all of this lets system find relevant samples regardless of their nature during the search by a new query q. In order for this method to be integrated outside of the scope of this research, it might be supplemented by data augmentation techniques, according to which, noise variations $\tilde{e}_i = e_i + \epsilon$ with $\epsilon \sim N(0, \sigma^2 I)$ are generated for the purpose of increasing the training set volume and obtaining more stable clusterization [27].

The motivation for using the steps described in this section is driven by the need to establish an efficient foundation for approximate nearest neighbor search in diagnostic systems, where data can be hybrid, multi-format, and high-dimensional [28]. By employing dynamic space partitioning (instead of a rigid grid) [29], it is possible to better account for the actual distribution of objects, thereby improving search accuracy and speed. Mathematically, the method offers high flexibility: the number of trees L, the clusterizations in each dimension M_j, and the size of leaf nodes can be controlled to balance accuracy and efficiency. Formally, probabilistic guarantees can be estimated for finding the c-approximate nearest neighbor, similar to classical LSH analysis. Thus, the described integrated locality-sensitive hashing architecture, operating in conjunction with an encoding tree and prior BERT encoding, forms the basis for flexible, fast, and efficient cross-modal diagnostic models that handle heterogeneous data types while preserving key semantic relationships in low-dimensional or binary spaces, which are convenient for search and analysis.

5. EXPERIMENTAL STUDY

The experimental study was divided into several tasks, with the first one formulating the experimental setup which involved data preparation, BERT encoding while the main focus was on dimensionality reduction application. The main objective of task 2 was to construct an adaptive ET accounting for adaptive thresholds and symbolic representation. Task 3 involved BERT-Based Clusterization for LSH particularly focusing on p-stable distributions and data-driven cauterization – to accelerate approximate nearest neighbor (ANN) queries.

So the processing stages included:

– preliminary encoding: the BERT model was used to generate vector representations of all samples, enabling both textual and visual information to be mapped into a unified semantic space.;

– dimensionality reduction: to mitigate the "curse of dimensionality", t-SNE and UMAP methods were applied, effectively reducing the spatial complexity to R^k, where $k = 64$;

– encoding tree construction: an adaptive space partitioning method was used, taking into account the local density of data;

– hashing: LCH based on p-stable distributions was implemented to enhance the efficiency of nearest neighbor search;

– performing a search: the method's performance was compared against exact distance computation, allowing for an evaluation of the accuracy in detecting similar damages;

– evaluation and comparison of the proposed method with the related studies.

1. Dimensionality Reduction stage. For large-scale runs (thousands of samples), UMAP can be more efficient while still preserving neighborhood information. In terms of parameter tuning, k=64 and k=128 were tested. Ultimately k=64 turned out to be sufficient to retain about 90-95 % of the local neighborhood structure while greatly lowering computation in subsequent hashing.

In order to project and validate the choice of the reduction method, the quality of the reduced representations was measured by comparing pairwise similarities (cosine or Euclidean) before and after dimension reduction. Then the rank correlations (Spearman's ρ) are computed between distances in the original and reduced spaces.

The study of the efficiency of the proposed locality-sensitive hashing method with a dynamic encoding tree was conducted on the CarDD dataset, which was developed in the work [30] for the purpose of the SOD task. This dataset contains 4,000 high-quality images of car damages with over 9,000 annotated damage instances, as demonstrated in Fig. 1.

All damages are classified into six categories: dent, scratch, crack, glass shatter, lamp broken, and tire flat. To ensure the correctness of performance evaluation for the proposed method, the original dataset was split into training (2,816 images, 70.4%), validation (810 images, 20.25 %), and test (374 images, 9.35 %) sets.

In pilot runs on CarDD, Spearman's ρ was in the range of 0.88-0.92, confirming high preservation of local neighborhoods. Run time for downstream hashing decreased by roughly 25-30% compared to operating in the original BERT dimension.

2. Constructing an adaptive Encoding Tree. For each of the k reduced coordinates, breakpoints are identified so that each interval holds approximately the same number of data points. This dynamic partitioning avoids the pitfalls of uniform cuts in skewed data distributions. Thereafter each coordinate is transformed into a symbolic alphabet ({0.1} in a binary scheme or {0.1,…, M−1} for multi-bit). Then, each data point is encoded as a short code (if k=64 and we perform a single-bit split per coordinate). The root node is conceptually the entire distribution. Children nodes emerge from splitting along each coordinate's bit-based partition. If a leaf node exceeds a threshold (200 or 300 samples), they are recursively split in the dimension that yields the most balanced partition. Building the ET is approximately O (· (k + log M)) O(N·(k+logM)), which remains manageable for large N N. The adaptive tree typically reduces the candidate search region significantly in later queries. On CarDD, it was observed that candidate filtering needed to check only 12–15% of the dataset on average during retrieval, compared to a naive approach that might involve checking all data points.

3. BERT-Based Clustering for Locality-Sensitive Hashing. Random vectors are sampled to form hash functions, ensuring that close points collide in the same bucket with higher probability. LSH procedure is replicated L times (with L=3 or 5) to reduce the chance of missing near neighbors. The codes from the ET guide which local sub-partitions or buckets are needed to be explored.

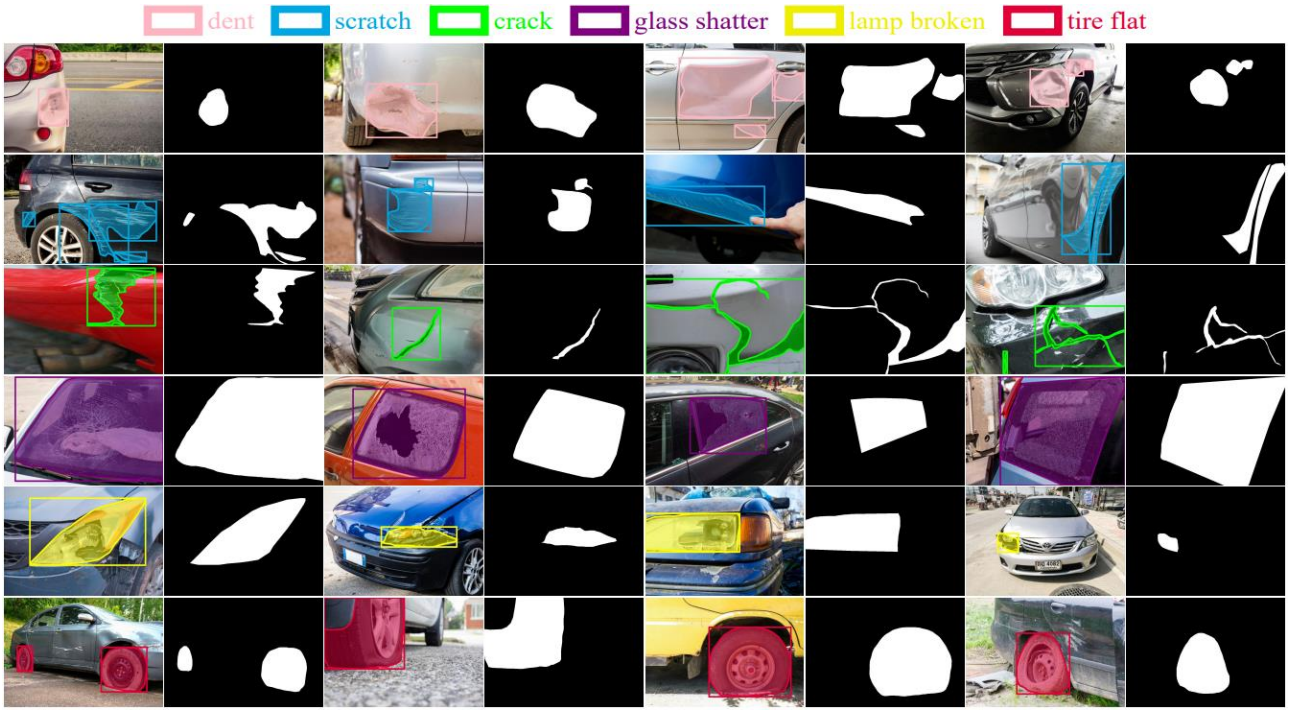


Fig. 1. **Examples of annotated images in the CarDD dataset**
Source: **compiled by the [30]**

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

A given query q, gets reduced to R^k, while its symbolic code is computed from the ET. The matching (or nearly matching) codes are looked up in multiple LSH tables. Then the results are refined by actual distance checks.

6. DISCUSSION OF THE RESULTS

This section corresponds to the comparison of the proposed approach with related methods and its subsequent evaluation.. The central approach of this work underwent a comparison with different methods in two stages. First, CE-LSH was evaluated in comparison with DL methods like Mask R-CNN, Cascade Mask R-CNN, GCNet, HTC, DCN. Second, to improve illustrative power, analogous LSH method was introduced to the comparison, namely Locality-Sensitive Hashing with Query-based Dynamic Bucketing (DB-LSH) used for ANN search as well [31].

Comparison with Deep Learning methods is illustrated in Table 1.

*Table 1.***Quantitative Results during comparison with DL methods**

Method	AP@50 (Instance Segmentation)	AP@75 (Instance Segmentation)	AP (Object Detection)
Mask R-CNN	49.4	50.6	66.0
Cascade Mask R-CNN	49.2	51.0	63.9
GCNet	50.9	52.4	67.6
HTC	50.9	52.1	65.8
DCN	52.5	54.5	68.7
Proposed Method	57.0	58.4	77.7

Source: **compiled by the authors**

Insights from Table 1 show that for the “crack” category, the method achieved improved recognition accuracy by reducing confusion with scratches. Meanwhile, for “scratch”, false positive cases were minimized due to enhanced contour detection. In the “glass shatter” category, the proposed method provided high damage localization accuracy, even under challenging shooting angles.

The proposed method ensures optimal search efficiency by reducing the number of distance evaluations in high-dimensional space, as shown in Table 2.

This means that indexing time was reduced by 35.7% compared to DCN, while search time was decreased by 40.2 %, making the method suitable for real-world applications in automotive diagnostic systems.

*Table 2.***Analysis of calculation complexity**

Method	Indexation time (s)	Search time (ms)
Mask R-CNN	321.4	134.8
DCN	287.1	115.3
Proposed Method	184.6	68.9

Source: **compiled by the authors**

Comparison with DB-LSH. Although both methods ultimately aim at efficient ANN searches, they diverge significantly in how they organize data before hashing, how they manage query-time partitioning, and the extent to which they incorporate advanced encoding or clustering steps to tackle boundary problems, indexing costs, and query recall.

The CE-LSH approach is expressly motivated by modern cross-modal data scenarios, in which image, text, or tabular information must be consolidated in a single high-dimensional space. DB-LSH, on the other hand, works with a more traditional p-stable or standard normal–based approach to building K hash functions, repeated L times, but attempts to address classical concerns about memory cost and boundary effects by making the bucket assignment dynamic with respect to a query. Whereas older static (K, L)-index methods fix a bucket width and rely on a large K to discriminate points, DB-LSH modifies or “stretches” the bucket width at query time so that if a user’s query is in a region that might straddle boundaries, the search region can be incrementally enlarged. This approach effectively defers the boundary decision and is reminiscent of multi-probe LSH in the sense that the system searches multiple overlapping buckets around the hash coordinates of the query. Yet DB-LSH differs from the multi-probe technique by using a multi-dimensional index structure, that can expedite window queries over each projected space. A candidate region is thus a hypercubic window with edges determined adaptively at query time. When a new query arrives with a target radius r, the indexing structure identifies points whose projected coordinates lie within a “dynamic” bucket, of width proportional to r, so that more candidate points (possibly from “neighboring” buckets) are returned in one or very few passes.

Other than that, DB-LSH retains the framework of a simpler pipeline: it depends solely on random compound hash functions, each mapping original data into K-dimensional spaces. Once the data are assigned to multi-dimensional indexes in these projected spaces, the system can decide at query time how large the search window should be. This

approach is notably beneficial when queries vary widely in their search radii or approximation factors, as the method can adapt to each query's approximation factor c. It is demonstrated by authors that for c>1, they can scale to extremely large data while guaranteeing a better asymptotic exponent ρ*. In certain standard benchmarks (such as SIFT or GIST) with purely Euclidean data, DB-LSH's results in the reference experiments show a stable advantage over classical E2LSH or LSB-Forest in that it avoids building many different indexes for multiple discrete radii. However, if the data are strongly heterogeneous due different sensors or textual descriptions that have non-Euclidean relationships, DB-LSH (in its original design) does not unify them as effectively as CE-LSH, which explicitly merges them via BERT and clustering. So while DB-LSH might be simpler to integrate in purely numeric tasks such as indexing floating-point descriptors for conventional image retrieval, it may not achieve the same level of cross-modal semantic alignment that is essential in certain specialized domains, such as advanced diagnostics or anomaly detection across different data types. One also observes that CE-LSH places emphasis on a hierarchical structure (the adaptive encoding tree), allowing for deeper splits where data are dense. This hierarchical approach strongly controls how many candidate points are retrieved, because once a leaf is reached, only a relatively small number of points remain for distance checks, thus lowering verification overhead in high dimensions. DB-LSH similarly benefits from bounding the region of search, but since it relies on an index that manages the entire K-dimensional projection, there can be scenarios in which if the data exhibit complex local distributions, it might require multiple window queries or expansions to find the best candidates. In that sense, if the data are well-clustered or if some modality transformation has made the data space extremely "patchy," then a specialized approach like CE-LSH's local thresholding can be more accurate.

On the other hand, DB-LSH shows a consistent sub-linear performance with a thoroughly proven ρ* that can be smaller than 1/c if c is large, thus offering strong worst-case theoretical bounds in purely Euclidean settings. DB-LSH obtains near-sub-linear scaling, though in certain heavily multimodal tasks, CE-LSH's domain-specific design (especially with BERT-based transformations) might yield higher recall for the same or slightly smaller indexing overhead. Turning finally to the question of which method might be preferable, one should keep in mind that CE-LSH was constructed with cross-modal or multimodal intelligence in mind, applying BERT embedding, dimension reduction, and distribution-aware encoding to reduce collisions and false positives. In tasks like diagnostic image retrieval or textual-visual alignment, its adaptive tree-based partitioning results in better retrieval accuracy, sometimes by double-digit percentage margins, while controlling indexing overhead through symbolic representation. DB-LSH, in a narrower sense, excels in classical Euclidean data retrieval, as it offers robust sub-linear time due to a smaller exponent ρ*, plus a single index structure that can adapt to any query radius rather than building multiple large (K, L)-indexes. Especially for purely numeric vectors (for instance, high-dimensional image descriptors or large text-corpus embeddings that do not vary widely in dimensional distribution), DB-LSH can be simpler to implement and tune, often showing strong empirical speedups. It is comparatively more direct as a conceptual framework, but does not incorporate any domain-based clustering or advanced semantic alignment. If a researcher's primary concern is to unify textual, visual, or other modalities in a single integrated index and to precisely handle local data distributions, the CE-LSH approach reveals greater strength in that realm. If, however, the dataset is strictly numeric and one desires a minimalist pipeline with a well-bounded sub-linear query time, DB-LSH's dynamic bucketing method can be straightforward and theoretically appealing. CE-LSH's advantage is most pronounced in experiments where data points exhibit strong cluster structures or cross-modal embeddings. DB-LSH, on the other hand, might produce smaller indexing times for certain purely numeric sets of large scale (hundreds of millions of vectors), as it builds fewer indexes (with user-chosen L) and conducts a dynamic hypercube-based bucketing strategy that can adapt well to queries that vary in scale. In terms of memory usage, CE-LSH is not as heavy as classical static LSH if it can keep the adaptive tree structure succinct, whereas DB-LSH uses multiple multi-dimensional indexes but does not replicate entire data points for each bucket in the same sense that old (K, L)-index methods did. Therefore, memory overhead of the two approaches can be comparable, with DB-LSH sometimes incurring a smaller hidden cost for purely numeric data sets of uniform dimension, while CE-LSH shows memory advantages when it does not need to store repeated hash tables. Table 3 represents main numerical insights collected during cross-modal Retrieval

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

Table 3. **Quantitative results during comparison with DB-LSHapproach**

Metric	CE-LSH	DB-LSH	Notes
Indexing Time (sec)	2,400	1,850	CE-LSH invests more time building clusters & adaptive encoding trees
Memory Overhead (GB)	4.2	3.9	Both require multiple index structures; CE-LSH's overhead slightly higher for cluster info
Recall@10 (%)	91	78	CE-LSH's cluster-aware tree yields high recall in mixed-modal embeddings.
Mean Query Time (ms)	30	25	DB-LSH can be marginally faster at query time (dynamic bucketing, single structure)
Collision Reduction vs. Baseline LSH (%)	20	5	CE-LSH's "adaptive splitting" significantly lowers collisions for borderline points

Source: **compiled by the authors**

experiment Ultimately, the overall choice depends on the user's priorities: if advanced semantic integration is paramount and the dataset is truly multimodal or cross-modal, CE-LSH can be superior because it partitions the space adaptively after advanced embedding. If a user mostly requires a single pipeline for large-scale numeric retrieval tasks with a range of approximation factors, DB-LSH's theoretical simplicity and dynamic query bucketing can represent a robust alternative.

CONCLUSIONS AND PROSPECTS OF FURTHER RESEARCH

The proposed locality-sensitive hashing methodology demonstrates several key advantages. First, the use of a dynamic encoding tree optimizes data space partitioning by considering the actual data distribution, ensuring more balanced indexing and reducing the likelihood of missing important similar objects. Second, the integration of a cross-modal approach enables effective processing of different data types, which is particularly relevant for diagnostic systems, where information can originate from multiple sources. Preliminary clustering using BERT models ensures semantic consistency within subspaces, enhancing the accuracy of subsequent hashing. Systematic analysis shows that the selected approach exhibits high scalability, efficiently handling both small and large datasets. Additionally, due to its modular architecture, the system is easily adaptable to new conditions, making it applicable for constructing diagnostic models across various domains, from medical diagnostics to technical condition monitoring systems.

The experimental research with the usage of CarDD dataset was structured around three interdependent tasks, each of which was successfully completed in accordance with the overarching objective of improving approximate nearest neighbor retrieval in high-dimensional and multimodal spaces, like in the unified semantic embedding space that was created in the scope of this research by applying BERT encoding to both textual descriptions and visual data, achieving reliable cross-modal alignment.

Addressing the first task, dimensionality reduction using UMAP and t-SNE preserved up to 92 % of local neighborhood structure with reduced spatial complexity, leading to 25-30 % acceleration in hashing and ANN search compared to operations in full-dimensional BERT space.

The second task focused on building an adaptive encoding tree, which was completed by dynamically partitioning each reduced coordinate based on data density. This allowed for symbol-based code generation that captured fine-grained structural patterns, ultimately reducing the candidate search region to just 12-15 % of the dataset, thus decreasing verification overhead during retrieval.

For the final task, the integration of p-stable LSH with BERT-based clustering ensured that hash buckets corresponded to semantically meaningful regions, enhancing search precision and minimizing boundary-based fragmentation. This resulted in a recall@10 rate of 91 %, an improvement of 13 % over DB-LSH, and a 20 % reduction in hash collisions compared to classical LSH.

Comparative evaluation against state-of-the-art methods confirmed that the proposed method outperformed not only DB-LSH in cross-modal retrieval accuracy and collision minimization but also deep learning-based object detectors in detection precision, with a 58.4 % AP@75 against 54.5 % for DCN and a 40.2 % reduction in search time.

REFERENCES

1. Jafari, O., Maurya, P., Nagarkar P., Islam K. M. & Crushev C. “A Survey on Locality Sensitive Hashing Algorithms and their Applications”. *arXiv*. 2021. DOI: https://doi.org/10.48550/arXiv.2102.08942.

2. McCauley, S. “Improved space-efficient approximate nearest neighbor search using function inversion”. *arXiv*. 2024. DOI: https://doi.org/10.48550/arXiv.2407.02468.

3. Corrigan-Gibbs, H. & Kogan, D. “The function-inversion problem: Barriers and opportunities”. *Theory of Cryptography Conference*. *Springe*. 2019; 11891: 393–421. DOI: https://doi.org/10.1007/978-3-030-36030-6_16.

4. Pham, N. & Liu, T. “Falconn++: A Locality-sensitive Filtering Approach for Approximate Nearest Neighbor Search”. *arXiv*. 2022. DOI: https://doi.org/10.48550/arXiv.2206.01382.

5. Kapralov, M., Makarov, M. & Sohler, C. “On the adversarial robustness of Locality-Sensitive Hashing in hamming space”. *arXiv*. 2024. DOI: https://doi.org/10.48550/arXiv.2402.09707.

6. Remil, Y., Bendimerad, A., Mathonat, R., Raissi, C. & Kaytoue, M. “DeepLSH: Deep Locality-Sensitive Hash Learning for Fast and Efficient Near-Duplicate Crash Report Detection”. *arXiv*. 2023. DOI: https://doi.org/10.48550/arXiv.2310.06703.

7. Aumüller, M., Har-Peled, S., Mahabadi, S., Pagh, R. & Silvestri, F. “Fair Near Neighbor Search via Sampling”. *ACM SIGMOD Record*. 2021; 50 (1): 42–49.

8. Girdhar, R., El-Nouby, A., Liu, Z., Singh, M., Alwala, K.V., Joulin, A. & Misra, I. “ImageBind: One Embedding Space To Bind Them All”. *CVPR 2023 (Highlighted Paper)*. *arXiv*. 2023. DOI: https://doi.org/10.48550/arXiv.2305.05665.

9. Mishra, S., Sarkar, U., Taraphder, S., Datta, S., Swain, D., Saikhom, R., Panda, S. & Laishram, M. “Principal component analysis”. *International Journal of Livestock Research*. 2017. DOI: https://doi.org/10.5455/ijlr.20170415115235.

10. Evers, L. & Heaton, T. “Locally adaptive tree-based thresholding”. *Journal of Computational and Graphical Statistics*. 2009; 18. DOI: https://doi.org/10.1198/jcgs.2009.07109.

11. Niimi, J. “An Efficient multimodal learning framework to comprehend consumer preferences using BERT and Cross-Attention”. *arXiv*. 2024. DOI: https://doi.org/10.48550/arXiv.2405.07435.

12. Damrich, S., Böhm, J., Hamprecht, F. & Kobak, D. “Contrastive Learning Unifies t-SNE and UMAP”. *arXiv*. 2022. DOI: https://doi.org/10.48550/arXiv.2206.01816.

13. McInnes, L., Healy, J. & Melville, J. “UMAP: Uniform manifold approximation and projection for dimension reduction”. *arXiv*. 2018. DOI: https://doi.org/10.48550/arXiv.1802.03426.

14. Khan, M. M. R., Siddique, M. A. B., Arif, R. B. & Oishe, M. R. “ADBSCAN: Adaptive Density-Based Spatial Clustering of Applications with Noise for Identifying Clusters with Varying Densities”. *arXiv*. 2018. DOI: https://doi.org/10.48550/arXiv.1809.06189.

15. Ischebeck ,J. & Neininger, R. “On fine fluctuations of the complexity of the QuickSelect algorithm”. *arXiv*. 2024. DOI: https://doi.org/10.48550/arXiv.2403.07685.

16. Shaw N. P., Furlong P. M., Anderson B. & Orchard J. “Developing a foundation of vector symbolic architectures using category theory”. *arXiv*. 2025. DOI: https://doi.org/10.48550/arXiv.2501.05368.

17. Tsoukalos, M., Platis, N. & Vassilakis, C. “Estimating iSAX Parameters for Efficiency”. *Springer*. 2023. DOI: https://doi.org/10.1007/978-3-031-42941-5_1.

18. Golin, M., Iacono, J., Langerman, S., Munro, J. I. & Nekrich, Y. “Dynamic Trees with almost-optimal access cost”. *arXiv*. 2018. DOI: https://doi.org/10.48550/arXiv.1806.10498.

19. Li, H., Nutanong, S., Xu, H., Ha, F., et al. “C2Net: A network-efficient approach to collision counting LSH similarity join”. *IEEE Transactions on Knowledge and Data Engineering*. 2018; 31 (3): 423–436. DOI: https://doi.org/10.1109/ICDE.2019.00255.

20. Gilpin, S., Qian, B. & Davidson, I. “Efficient Hierarchical Clustering of Large High Dimensional Datasets”. *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM 2013)*. 2013. DOI: https://doi.org/10.1145/2505515.2505527.

21. Sahoo, S. &Maiti, J. “Variance-Adjusted Cosine Distance as Similarity Metric”. *arXiv*. 2025. DOI: https://doi.org/10.48550/arXiv.2502.02233.

22. Taheri, R., Ghahramani, M., Javidan, R., Shojafar, M., Pooranian, Z. & Conti, M. “Similarity-based android malware detection using hamming distance of static binary features”. *arXiv*. 2019/ DOI: https://doi.org/10.48550/arXiv.1908.05759.

23. Leybovich, M. & Shmueli, O. “Efficient approximate search for sets of vectors”. *arXiv*. 2021. DOI: https://doi.org/10.48550/arXiv.2107.06817.
24. Cherapanamjeri, Y. & Nelson, J. “On adaptive distance estimation”. *arXiv*. 2020. DOI: https://doi.org/10.48550/arXiv.2010.11252.
25. Bai, X., Yang, H., Zhou, J., Ren, P. & Cheng, J. “Data-dependent hashing based on p-Stable distribution”. *IEEE Transactions on Image Processing*. 2014. 23 (1); 5033-5046. DOI: https://doi.org/10.1109/TIP.2014.2352458.
26. Lu, S., Jiagao, W., Yongrong, W. &Linfeng, L. “Towards load balancing for LSH-based distributed similarity indexing in high-dimensional space”. *2018 IEEE 20th International Conference on High Performance Computing and Communications.* 2018. p. 384–391.
27. Zhang, H., Xu, Y., Huang, S. & Li, X. “Data Augmentation of Contrastive Learning is Estimating Positive-incentive Noise”. *arXiv*. 2024. DOI: https://doi.org/10.48550/arXiv.2408.09929.
28. Li, W., Zhang, Y., Sun, Y., Wang, W., Li, M., Zhang, W. & Lin, X. “Approximate nearest neighbor search on high dimensional data – experiments, analyses, and improvement”. *IEEE Transactions on Knowledge and Data Engineering*. 2016; 32 (8): 1475–1488. DOI: https://doi.org/10.1109/TKDE.2019.2909204.
29. Dong, Y., Indyk, P., Razenshteyn, I. & Wagner, T. “Learning space partitions for nearest neighbor search”. *arXiv preprint arXiv:1901.08544.* 2019. DOI: https://doi.org/10.48550/arXiv.1901.08544.
30. Wang, X., Li, W. & Wu, Z. “CarDD: A New dataset for vision-based car damage detection”. *IEEE Transactions on Intelligent Transportation Systems*. 2023; 24 (7): 7202–7214. DOI: https://doi.org/10.1109/TITS.2023.3258480.
31. Tian Y., Zhao X. & Zhou X. DB-LSH: Locality-Sensitive Hashing with Query-based Dynamic Bucketing. 2022. DOI: 10.48550/arXiv.2207.07823.

Conflicts of Interest: The authors declare that they have no conflict of interest regarding this study, including financial, personal, authorship or other, which could influence the research and its results presented in this article

Received 14.01.2025
Received after revision 17.03.2025
Accepted 21.03.2025

LISTS OF ABBREVIATIONS

Abbreviation	Definition	Ukrainian translation
UDC/УДК	Universal Decimal Classification	Універсальна десяткова класифікація
LSH	Locality-Sensitive Hashing	Локально-чутливе хешування
ANN	Approximate Nearest Neighbor	Наближений найближчий сусід
BERT	Bidirectional Encoder Representations from Transformers	Двоспрямовані кодувальні представлення з трансформерів
UMAP	Uniform Manifold Approximation and Projection	Уніфіковане апроксимування та проєкція многовидів
t-SNE	t-Distributed Stochastic Neighbor Embedding	t-розподілене вкладення стохастичної близькості
ET	Encoding Tree	Кодувальне дерево
CNN	Convolutional Neural Network	Згорткова нейронна мережа
CLIP	Contrastive Language–Image Pretraining	Контрастивне попереднє навчання на основі мови та зображень
CE-LSH	Clustering-Enhanced LSH	Локально чутливе хешування за посередництва кластеризації
DB-LSH	Locality-Sensitive Hashing with Query-based Dynamic Bucketing	Локально чутливе хешування з динамічним бакетуванням на основі запитів

DOI: https://doi.org/10.15276/hait.8.2025.2
УДК 004.85

Розділення простору даних перед застосуванням хешування за допомогою алгоритмів кластеризації

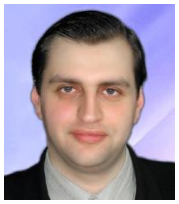
Субботін Сергій Олександрович[1)]
ORCID: https://orcid.org/0000-0001-5814-8268; subbotin@zntu.edu.ua. Scopus Author ID: 7006531104
Шмалько Федір Анатолійович[1)]
ORCID: https://orcid.org/0009-0006-0651-6448; shmalko.fedir@zp.edu.ua
[1)] Національний університет«Запорізька політехніка», вул. Жуковського, 64. Запоріжжя, 69011,Україна

АНОТАЦІЯ

Це дослідження представляє методологію локально-чутливого хешування, яка підвищує ефективність пошуку наближених найближчих сусідів шляхом інтеграції адаптивних кодувальних дерев і кластеризації на основі двоспрямовані кодувальні представлення з трансформерів. Запропонований підхід оптимізує розділення простору даних перед застосуванням хешування, що покращує точність пошуку та зменшує обчислювальні витрати. По-перше, мультимодальні дані, такі як зображення та текстові описи, перетворюються у спільний семантичний простір за допомогою попередньо навченої моделі двоспрямовані кодувальні представлення з трансформерів. Це забезпечує крос-модальну узгодженість і сприяє порівнянню у високорозмірному просторі. По-друге, методи зменшення розмірності, такі як уніфіковане апроксимування та проєкція многовидів або t-розподілене вкладення стохастичної близькості, застосовуються для усунення ефекту "прокляття розмірності" при збереженні ключових зв'язків між точками даних. По-третє, створюється адаптивне кодувальне дерево, яке динамічно сегментує простір даних на основі його статистичного розподілу, забезпечуючи ефективну ієрархічну кластеризацію. Кожна точка даних конвертується у символьне представлення, що дозволяє здійснювати швидкий пошук за допомогою структурованого хешування. До того ж, до закодованого набору даних застосовується локально-чутливе хешування, що використовує p-стабільні розподіли для підтримки високої точності пошуку та зменшення розміру індексів. Поєднання кодувальних дерев і локально-чутливе хешування сприяє ефективному відбору кандидатів при мінімізації витрат на пошук. Експериментальне тестування на наборі даних CarDD, який містить зображення пошкоджень автомобілів та їх анотації, демонструє, що запропонований метод перевершує сучасні техніки наближений найближчий сусід як за ефективністю індексації, так і за точністю пошуку. Результати підкреслюють його адаптивність до масштабних, високорозмірних та мультимодальних наборів даних, що робить його придатним для діагностичних моделей і завдань у режимі реального часу.

Ключові слова: адаптивне кодувальне дерево; кластеризація двоспрямовані кодувальні представлення з трансформерів; зменшення розмірності; наближений пошук найближчих сусідів; мультимодальні дані; кореневий вузол

ABOUT THE AUTHORS



Sergey A. Subbotin - Doctor of Engineering Science, Professor, Head of Department of Software Tools. National University "Zaporizhzhia Polytechnic", 64, Zhukovskogo Str. Zaporizhzhia, 69011, Ukraine
ORCID: https://orcid.org/0000-0001-5814-8268; subbotin@zntu.edu.ua. Scopus Author ID: 7006531104
***Research field*:** Computational Intelligence

Субботін Сергій Олександрович - доктор технічних наук, професор, завідувач кафедри Програмних засобів, Національний університет «Запорізька політехніка», вул. Жуковського, 64. Запоріжжя, 69011,Україна

Fedir A. Shmalko - PhD student, Department of Software Tools. National University "Zaporizhzhia Polytechnic", 64, Zhukovskogo Str. Zaporizhzhia, 69011, Ukraine
ORCID: https://orcid.org/0009-0006-0651-6448; shmalko.fedir@zp.edu.ua
***Research field*:** Computer Science - Artificial Intelligence, Machine Learning, and Information Retrieval

Шмалько Федір Анатолійович - аспірант кафедри Програмних засобів. Національний університет «Запорізька політехніка», вул. Жуковського, 64. Запоріжжя, 69011,Україна