RESEARCH ARTICLE

MDH-Pose: Multimodal Deep Hashing with Adaptive Graph Fusion for Efficient Retrieval of Similar Protein Structures

Abid Hussain[1,*], **Haider Abbas**[2] and **Ijlal Hussain**[3]

[1] School of Software Engineering, Northwestern Polytechnical University, Xi'an 710072, China
[2] School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu 611756, China
[3] School of Science, China University of Geosciences (Beijing), Beijing 100083, China

Abstract

The Protein Data Bank (PDB) and, even more, predicted-structure databases have grown to a size at which pairwise structural alignment is no longer a practical way to search them. Alignment tools are accurate, but their cost grows superlinearly with protein length and linearly with database size, which rules out interactive search over hundreds of millions of entries. We present MDH-Pose, a multimodal deep hashing framework that maps each protein structure to a compact binary code. Separate encoders process the amino acid sequence, the Cα geometry and residue-level physicochemical properties, and a cross-modal attention module fuses the three streams. In the structural stream, the edge weights of the residue graph are learned from spatial proximity, BLOSUM62 substitution similarity and physicochemical similarity, so that the graph attention network can also use residue pairs that are functionally related but not in direct contact. Codes are trained with a triplet objective whose margin grows with the TM-score gap between the positive and negative pair and with training progress. On SCOPe, CATH and PDB40, MDH-Pose reaches a mean average precision (mAP) of 94.2% $\pm$ 0.8%, 91.7% $\pm$ 0.9% and 89.5% $\pm$ 1.0%, respectively, with an average query time of 0.54 ms and 98.7% less storage than Cα coordinates (128-bit codes, proteins of about 103 residues), which makes large-scale structure retrieval feasible on standard hardware. The margin over TM-align widens from 6.9 percentage points on SCOPe to 7.9 on the harder PDB40 set, and ablation studies show that removing any of the three modalities lowers accuracy on all datasets.

Keywords: protein structure retrieval, deep hashing, multimodal learning, graph neural networks, bioinformatics, similarity search.

1 Introduction

Finding structurally similar proteins quickly is a routine step in functional annotation of new structures,

Submitted: 12 June 2026
Revised: 19 September 2026
Accepted: 12 October 2026
Published: 28 November 2026

Vol. 3, **No.** 1, 2027.
10.62762/JAIB.2026.100792

***Corresponding author:**
✉ Abid Hussain
abidhussain@nwpu.edu.cn

Citation
Hussain, A., Abbas, H., & Hussain, I. (2026). MDH-Pose: Multimodal Deep Hashing with Adaptive Graph Fusion for Efficient Retrieval of Similar Protein Structures. *Journal of Artificial Intelligence in Bioinformatics*, 3(1), 13–26.

© 2026 by the Authors. Published by Institute of Central Computation and Knowledge. This is an open access article under the CC BY license (https://creativecommons.org/licenses/by/4.0/).

drug target identification, evolutionary analysis and structure-based drug design. The standard approach is pairwise structural alignment: the atomic coordinates of two proteins are superposed and a similarity score is computed. DALI [1], TM-align [2] and CE [3] are the best-known implementations and are still the reference against which new methods are tested. They are accurate, but three properties make them impractical for real-time retrieval from large databases.

First, pairwise alignment scales superlinearly with the number of residues, between $O(n^2)$ and $O(n^3)$ depending on the algorithm. Comparing one query exhaustively against a full structural database is therefore expensive, and an all-against-all comparison over the PDB can take days to weeks on a single CPU. Second, the methods operate on floating-point coordinates, which are a poor basis for a search index: storage and memory bandwidth grow in direct proportion to database size, so full coordinate sets are too large to store and stream on lightweight web or mobile platforms. Third, rigid superposition handles multi-domain proteins, flexible linkers and conformational variability poorly, because a single global transformation cannot follow domains that move relative to one another. The scale problem has become much sharper with deep-learning structure prediction, which now produces models faster than they can be searched [4, 5]. The AlphaFold Protein Structure Database holds predicted models for more than 10^8 proteins [6], and metagenomic prediction has added hundreds of millions more [7]. The bottleneck is no longer structure determination but search [8, 9].

A rough calculation shows the size of the gap. Even if one pairwise alignment took only 1 ms, scanning 10^8 entries for a single query would take 10^5 s, i.e., more than a day of CPU time, and an all-against-all comparison would be out of reach. What is needed instead is a representation that is computed once per structure, stored in a few bytes, and compared in constant time per pair, so that the cost of a query is dominated by memory bandwidth rather than by geometry.

Learned representations offer another route: once a protein is embedded, the embedding can be reused for every query instead of recomputing an alignment. Current learned protein representations, however, have two limitations for retrieval. Most use only one modality, either sequence [10, 11] or structure [12, 13], although the two are complementary, and physicochemical properties of residues carry functional information that neither encodes explicitly; multimodal protein models have appeared only recently [14, 15]. Most also output high-dimensional continuous embeddings. These are useful for downstream prediction, but both storage and distance computation grow with embedding dimension, which makes large-scale nearest-neighbor search costly. Learning to hash addresses the second limitation. A hash function maps each item to a short binary code that preserves similarity in Hamming space, where a distance is a population count over a machine word. Hashing originated in similarity search for large databases [16] and is well established in computer vision, with a large literature developed over the past two decades [17–20]. It has only recently been applied to protein structures [21], where similarity is defined by structural alignment rather than by semantic labels.

We propose MDH-Pose, which combines multimodal representation learning with supervised hashing for protein structure retrieval. Our contributions are as follows.

- **A unified multimodal architecture.** Rather than concatenating independently trained representations, we encode sequential, structural and physicochemical information with separate encoders and merge them with a cross-modal attention module.
- **An adaptive graph neural network for the structural stream.** Instead of a fixed distance threshold, edge weights in the residue interaction graph are computed from spatial proximity, amino acid substitution similarity and physicochemical similarity with learnable coefficients. The network can therefore attend to residue pairs that are functionally related even when they are not closely packed.
- **A triplet objective with adaptive margin scheduling.** The margin scales with the structural similarity gap between the positive and negative pair and with training progress. This acts as a curriculum and improves the separation of proteins that are structurally close but not identical.
- **An end-to-end pipeline** that produces hash codes of 16–256 bits. On our benchmarks, MDH-Pose reduces storage by 98.7% relative to Cα coordinates (128-bit codes, proteins of about 103 residues) while matching or exceeding the

accuracy of alignment-based methods.

The rest of the paper is organized as follows. Section 2 reviews protein structure comparison, learned protein representations and hashing-based retrieval. Section 3 describes MDH-Pose: the multimodal encoders, the adaptive graph neural network (GNN) and the training objective. Section 4 gives the datasets, evaluation measures, baselines and implementation details. Section 5 reports the main results and ablation studies. Section 6 discusses the findings and future directions, and Section 7 concludes the paper.

2 Related Work

2.1 Structural Alignment and Structure Search

DALI compares proteins through their intramolecular distance matrices using Monte Carlo optimization [1], and was used for an all-against-all comparison of the PDB to map the protein fold universe [22]. TM-align combines dynamic programming with TM-score optimization and yields a length-normalized score that is largely independent of protein size [2, 23]. CE builds alignments by combinatorial extension of aligned fragment pairs [3]. All three are accurate, and all scale quadratically or worse with the number of residues. A different line of work reduces structure comparison to sequence comparison over a structural alphabet. Foldseek encodes tertiary residue interactions as a 3D-interaction alphabet and reports a speed-up of four to five orders of magnitude over alignment tools while retaining most of their sensitivity [8]. Earlier, a kappa-alpha structural alphabet with a BLOSUM-like substitution matrix was used for rapid search of structure databases with precomputed encodings [24].

2.2 Learned Protein Representations

Protein language models are trained on sequences alone at scale. ProtTrans/ProtBERT [10] and ESM [11] produce residue- and protein-level embeddings that transfer to function prediction and remote homology detection, and the ESM-2 language model also predicts atomic-level structure directly from sequence [7]. Dense sequence embeddings have already been used for fast homolog retrieval [9]. Structure-based encoders operate on the residue graph: DeepFRI trains a graph convolutional network on contact maps for function prediction [12], GearNet pre-trains a multi-relational residue graph encoder with contrastive and self-prediction tasks [13], and geometric vector perceptron networks learn directly from local geometric features [25]. Contrastive objectives, popularized for images by SimCLR [26], have also been used to pre-train 3D structure encoders [13]. Multimodal models that combine sequence and structure have recently been proposed for function description and representation learning [14, 15].

These methods produce continuous embeddings of 256 to 1024 dimensions. They are good feature extractors but were not designed as search indices; using them for retrieval requires either exhaustive floating-point comparison or a separate approximate nearest-neighbor layer.

2.3 Learning to Hash

Data-independent hashing, such as locality-sensitive hashing (LSH), offers asymptotic guarantees but needs long codes in practice [16, 27]. Data-dependent methods learn the projection: iterative quantization minimizes the quantization error under a rotation [17], and minimal loss hashing optimizes a structured-prediction bound [28]. Deep hashing learns features and codes jointly, using pairwise labels [18, 29], ranking supervision [30], continuation to handle the non-differentiable sign function [19], greedy discrete optimization [31], or hash centers as similarity targets [32, 33]. Cross-modal hashing [34] and multi-feature hashing [35] extend these ideas to heterogeneous inputs, which is also the setting of MDH-Pose, where several feature streams are merged into one code. A survey of the area is given in [20].

These ideas have recently been applied to protein structure retrieval. POSH [21] builds a k-nearest-neighbor residue graph with hand-crafted bond- and dihedral-angle node features and distance-based edge features, trains it with an InfoNCE-style contrastive objective with one positive and K sampled negatives per query, and reports results on SCOPe and an independent PDB subset. POSH showed that hashing is viable for this task, but it relies on structure alone, without sequence or physicochemical information. MDH-Pose differs from POSH and from the general hashing literature in three ways: (i) it fuses sequential, structural and physicochemical streams through cross-modal attention instead of using the structural stream only; (ii) the graph over which structure is aggregated has adaptive edge weights, with learnable coefficients for spatial, substitution and physicochemical similarity; and (iii) it is trained with a triplet loss whose margin grows with the structural similarity gap between the positive and negative and with training progress, rather than with the fixed-temperature contrastive

loss over a single positive used by POSH [21].

3 Proposed Method

This section describes MDH-Pose. The framework has three components: a multimodal architecture that integrates sequential, structural and physicochemical information; an adaptive graph neural network that learns which residue interactions are relevant; and a triplet objective with adaptive margin scheduling. We first formalize the hashing problem and then describe each component. The overall architecture is shown in Figure 1.

3.1 Problem Formulation

The protein structure hashing problem is to learn a mapping $h : \mathcal{P} \to \{-1, +1\}^k$ that assigns each protein P_i a k-bit binary code $\mathbf{b}_i = h(P_i)$, such that the Hamming distance between codes preserves the structural similarity between proteins. For two proteins P_i and P_j with structural similarity $s(P_i, P_j) \in [0, 1]$ (TM-score or another normalized similarity measure), the codes should satisfy

$$d_H(\mathbf{b}_i, \mathbf{b}_j) \propto 1 - s(P_i, P_j), \tag{1}$$

where d_H is the normalized Hamming distance

$$d_H(\mathbf{b}_i, \mathbf{b}_j) = \frac{1}{k} \sum_{l=1}^{k} \mathbb{I}\big[b_i^{(l)} \neq b_j^{(l)}\big],$$

$b_i^{(l)}$ is the l-th bit of $\mathbf{b}_i$ and $\mathbb{I}[\cdot]$ is the indicator function. Normalization by k keeps d_H in $[0, 1]$, so codes of different lengths are directly comparable. The structural similarity $s(P_i, P_j)$ is the normalized TM-score [23], where 1 indicates identical structures and values near 0 indicate no structural similarity. Given a query protein P_q and a database $\mathcal{D}$, retrieval returns the K proteins with the smallest Hamming distances:

$$\mathcal{R}(P_q, K) = \operatorname*{arg\,min}^{(K)}_{P_i \in \mathcal{D}} \; d_H\big(h(P_q), h(P_i)\big), \tag{2}$$

where $\arg\min^{(K)}$ denotes the set of K elements with the smallest values.

Computing d_H is cheap. For codes packed into 64-bit words, the Hamming distance between two k-bit codes requires $\lceil k/64 \rceil$ XOR operations followed by population counts, i.e., two word operations for $k = 128$. A linear scan over a database of N proteins therefore costs $O(Nk/64)$ word operations and needs $Nk/8$ bytes of memory. For $N = 10^8$ and $k = 128$ the codes occupy 1.6 GB and fit in the main memory of a single workstation, whereas Cα coordinates for the same number of proteins with about 100 residues each would require more than 100 GB. For larger databases, the codes can additionally be indexed with multi-index hashing or bucketing, but we use exhaustive Hamming ranking in all experiments so that the reported accuracy does not depend on an approximate index.

3.2 Sequential Encoder

The sequential encoder captures local and long-range sequence-order information that complements the 3D structure stream. Each residue type is mapped to a learnable embedding vector (equivalent to a linear layer applied to its one-hot encoding). The embedded sequence is passed to a multi-scale 1D convolutional block with kernel sizes $\{3, 5, 7, 9\}$ and 64 filters per kernel size, which detects local sequence motifs at several receptive-field sizes. A bidirectional LSTM (BiLSTM) with 128 hidden units per direction then processes the concatenated multi-scale features and captures long-range dependencies in both directions. Attentive pooling merges the per-residue BiLSTM states into a fixed-length sequence embedding, which is passed to the cross-modal attention module (Section 3.5). The kernel sizes were chosen to cover the typical length of local structural motifs: a window of 3–5 residues spans a turn, while 7–9 residues span about two turns of an α-helix (3.6 residues per turn) or a short β-strand.

3.3 Structural Encoder

The structural encoder uses an adaptive graph neural network to capture 3D geometric patterns. We build a residue interaction graph $G = (V, E)$, where V is the set of residues and E the set of spatial relationships between them. Edges connect residues whose Cα atoms lie within 12 Å of each other (Section 4.4).

Instead of the static graph construction used in most existing methods, edge weights are computed from several criteria. For every edge (u, v), i.e., every residue pair within the cutoff, we compute

$$w_{uv} = \alpha \exp\left(-\frac{\|\mathbf{x}_u - \mathbf{x}_v\|_2^2}{\sigma_d^2}\right) + \beta\, \mathrm{sim}_{aa}(a_u, a_v) + \gamma\, \mathrm{sim}_{phy}(\boldsymbol{\phi}_u, \boldsymbol{\phi}_v) \tag{3}$$

where $\mathbf{x}_u$ is the Cα coordinate of residue u, a_u its amino acid type and $\boldsymbol{\phi}_u$ its physicochemical feature vector (Section 3.4); α, β and γ are learnable coefficients and σ_d is a distance scale. sim_{aa} is derived from the

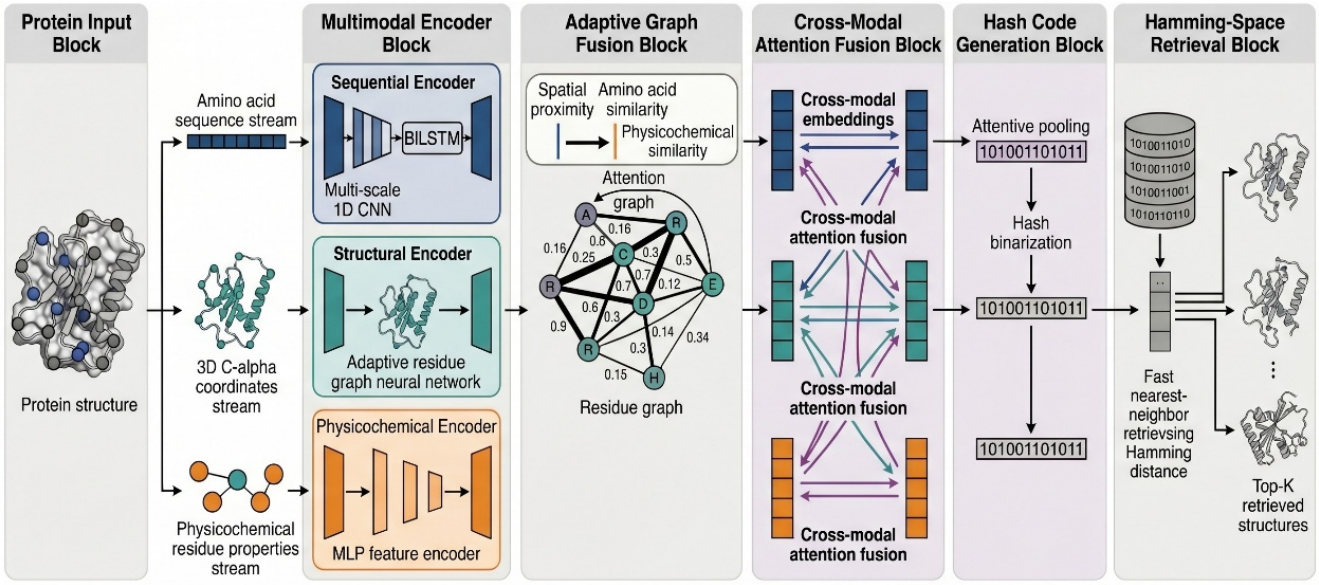

Figure 1. Architecture of the MDH-Pose framework. Sequence, structural and physicochemical streams pass through modality-specific encoders, adaptive graph fusion and cross-modal attention, followed by hash code generation and Hamming-space retrieval.

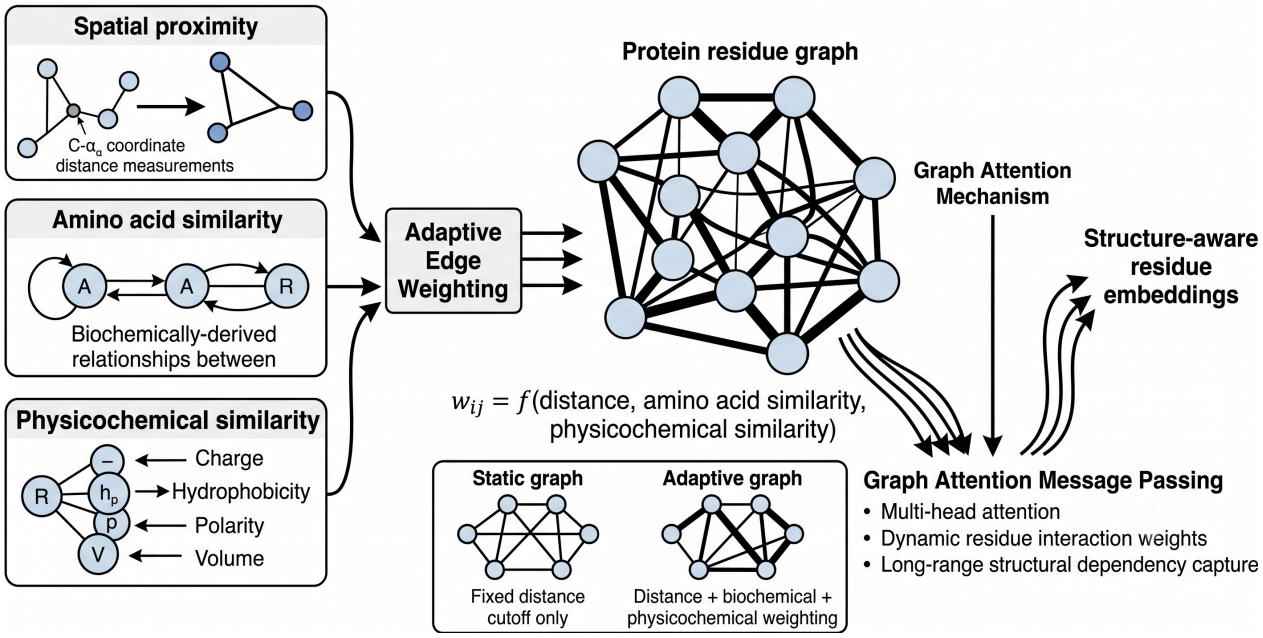

Figure 2. Adaptive graph fusion in MDH-Pose. Spatial proximity, amino acid similarity and physicochemical similarity are combined into adaptive edge weights, after which graph attention message passing produces structure-aware residue embeddings.

BLOSUM62 substitution matrix [36], and sim_{phy} is the cosine similarity between physicochemical feature vectors. With this weighting the network can learn interactions that are functionally relevant, not only those that are spatially close. The weighted graph is processed by a three-layer graph attention network (GAT) [37]; in each layer the outputs of all attention heads are concatenated and layer normalization is applied. The way w_{uv} enters the attention coefficients should be stated explicitly. The adaptive edge weighting and the message passing are illustrated in Figure 2.

The three terms of Eq. (3) behave differently over the graph. With $\sigma_d = 5$ Å, the spatial term decays quickly: at the 12 Å cutoff it falls to $\exp(-144/25) \approx 0.003$ of its maximum. Near the cutoff, the edge weight is therefore determined almost entirely by the substitution and physicochemical terms, which lets the network keep long-range edges between residues

with similar chemistry and suppress those between dissimilar residues. Because α, β and γ are learned, the model can also adjust the relative importance of geometry and chemistry to the dataset.

3.4 Physicochemical Encoder

The physicochemical encoder processes residue-level properties that affect folding, stability and function. For each residue r_u the feature vector $\boldsymbol{\phi}_u$ contains:

- standard physicochemical parameters: molecular mass, charge, hydrophobicity (Kyte-Doolittle scale [38]), polarity and volume;
- secondary structure propensities: the probabilities of α-helix, β-strand and coil predicted by PSIPRED [39];
- solvent accessibility: predicted relative solvent accessibility (RSA);
- structural disorder: the probability of lying in a disordered region, predicted by DISOPRED [40].

These features are processed by a two-layer multilayer perceptron (MLP), $\mathbf{H}^{(phy)} = \mathrm{MLP}(\boldsymbol{\phi}; \theta_{phy})$, where θ_{phy} denotes the network parameters. The two layers have 128 and 64 units, with ReLU activations and batch normalization [41].

3.5 Cross-Modal Attention Fusion

The outputs of the three modality-specific encoders, $\mathbf{H}^{(seq)}$, $\mathbf{H}^{(str)}$ and $\mathbf{H}^{(phy)}$, are fused before hash code generation. For each modality $m \in \mathcal{M} = \{seq, str, phy\}$ we compute a self-attention output Attn_m, and for each ordered pair of modalities a cross-attention output. The fused representation is their weighted sum:

$$\begin{aligned}\mathbf{H}^{(fuse)} = &\sum_{m\in\mathcal{M}} \lambda_m \,\mathrm{Attn}_m \\ &+ \sum_{\substack{(m_1,m_2)\in\mathcal{M}^2 \\ m_1\neq m_2}} \mu_{m_1,m_2} \\ &\times \mathrm{CrossAttn}\big(\mathbf{H}^{(m_1)}, \mathbf{H}^{(m_2)}\big)\end{aligned} \tag{4}$$

where λ_m and μ_{m_1,m_2} are trainable fusion weights, initialized uniformly and adjusted during training. With three modalities, Eq. (4) contains three self-attention terms and six cross-attention terms. Compared with concatenation, the weighted sum keeps the dimension of $\mathbf{H}^{(fuse)}$ independent of the number of modalities, and the learned weights indicate how much each modality pair contributes to the final code.

3.6 Hash Code Generation

A hash projection layer maps the protein-level representation $\mathbf{z} \in \mathbb{R}^{d_{fuse}}$ to a k-dimensional continuous embedding:

$$\hat{\mathbf{b}} = \tanh\big(\mathbf{W}_h \mathbf{z} + \mathbf{u}_h\big), \tag{5}$$

where $\mathbf{W}_h \in \mathbb{R}^{k\times d_{fuse}}$ and $\mathbf{u}_h \in \mathbb{R}^k$. The $\tanh$ activation constrains the embedding to $[-1,+1]^k$, which approximates the binary target. During training a quantization penalty encourages binarization:

$$\mathcal{L}_{quant} = \frac{1}{Nk} \sum_{i=1}^{N} \big\| \,|\hat{\mathbf{b}}_i| - \mathbf{1}\big\|_2^2, \tag{6}$$

where $|\cdot|$ is applied element-wise; the term penalizes deviations from $\{-1,+1\}$. At inference the continuous embedding is binarized with the sign function, $\mathbf{b} = \mathrm{sign}(\hat{\mathbf{b}})$. The quantization penalty reduces the gap between the continuous embeddings optimized during training and the binary codes used at retrieval time; without it, entries of $\hat{\mathbf{b}}$ close to zero could flip sign under small perturbations and change the ranking.

3.7 Adaptive Triplet Learning

A triplet objective, originally proposed for learning face embeddings [42], preserves fine-grained structural similarity in Hamming space. A constant margin is not ideal, for two reasons: the appropriate margin differs between triplets as their similarity gap changes, and it also changes during training as the embeddings evolve. We therefore schedule the margin adaptively.

For each anchor protein P_i, a triplet (P_i, P_j, P_k) is formed from a positive P_j with high structural similarity to P_i and a negative P_k with low similarity. Positives are sampled with probability proportional to $1 - s(P_i, P_j)$. This is hard-positive mining: it favors positives that are similar but not identical and therefore lie near the decision boundary, where the model is most likely to err. Negatives are sampled with probability proportional to $s(P_i, P_k)$, which favors hard negatives. This is a similarity-weighted variant of distance-weighted sampling [43], which likewise draws negatives according to their distance from the anchor instead of uniformly. The similarity-dependent margin is

$$m_{ijk} = m_0 + \eta\big(s(P_i, P_j) - s(P_i, P_k)\big), \tag{7}$$

where m_0 is the base margin, η is a scaling factor and $s(P_i, P_j) - s(P_i, P_k)$ is the similarity gap between the

positive and negative pairs. The epoch-dependent schedule is

$$m_{epoch} = m_{\min} + \big(m_{\max} - m_{\min}\big)\frac{t}{T}, \tag{8}$$

where t is the current epoch and T the total number of epochs, so the margin increases gradually during training, in the spirit of curriculum learning [44]. The adaptive triplet loss is

$$\mathcal{L}_{triplet}^{(adapt)} = \max\Big(0,\, \tilde{d}_H(\hat{\mathbf{b}}_i, \hat{\mathbf{b}}_j) - \tilde{d}_H(\hat{\mathbf{b}}_i, \hat{\mathbf{b}}_k) + m_{ijk}\, m_{epoch}\Big), \tag{9}$$

where $\tilde{d}_H(\hat{\mathbf{b}}_i, \hat{\mathbf{b}}_j) = \frac{1}{2}\big(1 - \frac{1}{k}\hat{\mathbf{b}}_i^{\top}\hat{\mathbf{b}}_j\big)$ is the continuous relaxation of the normalized Hamming distance; for binary codes it equals d_H exactly.

To make the codes of each class more compact, we add a center loss [45], which was introduced to make face features of the same identity compact, and the total objective is

$$\mathcal{L} = \mathcal{L}_{triplet}^{(adapt)} + \lambda_{center}\mathcal{L}_{center} + \lambda_{quant}\mathcal{L}_{quant}, \tag{10}$$

$$\mathcal{L}_{center} = \frac{1}{N}\sum_{i=1}^{N}\big\|\hat{\mathbf{b}}_i - \mathbf{c}_{y_i}\big\|_2^2, \tag{11}$$

where y_i is the fold (class) label of protein i and $\mathbf{c}_{y_i}$ is the corresponding class center, updated during training as the running mean of the embeddings assigned to that class. The center term pulls the codes of structurally related proteins toward a common center, whereas the triplet loss only constrains relative distances within sampled triplets.

Since $\tilde{d}_H$ lies in $[0, 1]$, the effective margin $m_{ijk}\, m_{epoch}$ must also stay below 1; otherwise every triplet would violate the margin and the loss would never reach zero. The values of m_0, η, $m_{\min}$ and $m_{\max}$ should therefore be chosen so that $(m_0 + \eta)\, m_{\max} < 1$.

4 Experiments

This section describes the datasets, evaluation metrics, baselines and implementation details.

4.1 Datasets

We evaluate MDH-Pose on three benchmark datasets.

SCOPe (Structural Classification of Proteins–extended) [46] classifies protein structures by evolutionary and structural relationships. We use the SCOPe 2.07 40% sequence-identity subset.

CATH (Class, Architecture, Topology, Homologous superfamily) [47] provides an alternative hierarchical classification. We use CATH v4.3 with the non-redundant subset at 35% sequence identity.

PDB40. To test a large and realistic setting, we built a dataset from the PDB containing all proteins with a resolution of 3.0 Å or better and pairwise sequence identity below 40%.

To avoid data leakage, all datasets are split at the family level or above. For SCOPe, proteins of the same superfamily are kept in the same split. For CATH, we split at the topology (T) level. For PDB40, proteins were clustered at 40% sequence identity with CD-HIT [48, 49], and whole clusters were assigned to splits, so that training and test proteins share less than 40% sequence identity. Splitting at the superfamily (SCOPe) or topology (CATH) level is stricter than a random or family-level split: proteins in the same superfamily are usually homologous, so a family-level split would allow the model to recognize close relatives seen during training instead of learning transferable structural features. All structures were preprocessed by removing heteroatoms, standardizing atom names and extracting Cα coordinates.

4.2 Evaluation Metrics

Retrieval performance is measured with standard information retrieval metrics. The main metric is mean average precision (mAP):

$$\mathrm{mAP} = \frac{1}{|Q|}\sum_{q \in Q}\mathrm{AP}(q), \tag{12}$$

where Q is the query set and $\mathrm{AP}(q)$ is the average precision of query q:

$$\mathrm{AP}(q) = \frac{1}{|\mathrm{Rel}(q)|}\sum_{k=1}^{|\mathcal{D}|} P@k(q)\,\mathrm{rel}\big(R_q(k)\big). \tag{13}$$

Here R_q is the ranked list of database proteins, $R_q(k)$ its k-th element, $\mathrm{rel}(\cdot) \in \{0, 1\}$ the relevance indicator and $\mathrm{Rel}(q)$ the set of proteins relevant to q. Precision at K (P@K) is the fraction of relevant proteins among the top K results:

$$P@K(q) = \frac{1}{K}\sum_{k=1}^{K}\mathrm{rel}\big(R_q(k)\big). \tag{14}$$

We report P@1, P@5 and P@10. Recall at K (R@K) is the fraction of all relevant proteins that appear in the

top K:

$$R@K(q) = \frac{1}{|\mathrm{Rel}(q)|} \sum_{k=1}^{K} \mathrm{rel}\big(R_q(k)\big). \tag{15}$$

The normalized discounted cumulative gain (nDCG) gives more credit to relevant items ranked near the top:

$$\mathrm{nDCG@}K(q) = \frac{\mathrm{DCG@}K(q)}{\mathrm{IDCG@}K(q)}, \tag{16}$$

$$\mathrm{DCG@}K(q) = \sum_{k=1}^{K} \frac{2^{\mathrm{rel}(R_q(k))} - 1}{\log_2(k+1)}, \tag{17}$$

where $\mathrm{IDCG@}K(q)$ is the DCG of the ideal ranking. With binary relevance, the gain $2^{\mathrm{rel}} - 1$ is either 0 or 1, so nDCG@K differs from P@K mainly in rewarding relevant proteins that appear near the top of the list. We use mAP as the primary metric because it summarizes the whole ranking, while P@1 and P@10 reflect the short lists a user typically inspects.

Retrieval time is the average time per query for top-K retrieval, including hash code computation and Hamming distance comparison. **Storage efficiency** is the compression ratio of hash codes relative to storing full Cα coordinates. With 32-bit floating-point coordinates, a protein of L residues needs $12L$ bytes, whereas a 128-bit code needs 16 bytes; the 98.7% reduction corresponds to $L \approx 103$. The reported ratio therefore assumes proteins of about 103 residues; shorter proteins give a smaller reduction and longer ones a larger one.

4.3 Baseline Methods

We compare MDH-Pose with three groups of baselines. *Alignment-based*: DALI [1], which aligns distance matrices with Monte Carlo optimization; TM-align [2], which uses dynamic programming with TM-score evaluation; and CE [3], which uses combinatorial extension of the alignment path. *Learning-based embeddings*: DeepFRI [12], a graph convolutional network for function prediction whose embeddings we use for retrieval; ProtBERT [10], a transformer protein language model whose sequence embeddings we use for retrieval; and GearNet [13], a message-passing network on protein structure graphs. *Hashing-based*: LSH [16, 27], ITQ [17] and DSH [18], which produce 128-bit binary codes compared by Hamming distance. For the alignment methods, database proteins are ranked by each tool's native score (the DALI Z-score, the TM-score normalized by the query length, and the CE Z-score).

All learning-based baselines were retrained or fine-tuned on our training splits. Alignment methods were run with their publicly available default implementations. Hyperparameters of the learning-based baselines were tuned on the validation set with a fixed budget of 100 GPU hours per method; the search space covered learning rates in $[10^{-5}, 10^{-2}]$, batch sizes in $[16, 64]$ and embedding dimensions in $[64, 1024]$. The final values are listed in Table 1. Embedding-based methods were compared with cosine similarity and hashing-based methods with Hamming distance. Runtime was measured on a fixed set of 1,000 query proteins on an NVIDIA RTX 4090 GPU with an Intel Xeon Gold 6248R CPU. All experiments were repeated five times with random seeds $\{42, 123, 456, 789, 1011\}$, and results are reported as mean $\pm$ standard deviation. Paired t-tests ($p < 0.05$) were used to assess the difference between MDH-Pose and each baseline. Because each test compares five paired runs, it has four degrees of freedom; the p-values in Table 1 should be read with this sample size in mind.

4.4 Implementation Details

MDH-Pose is implemented in PyTorch 1.12 with CUDA 11.6. All models were run on a server with NVIDIA RTX 4090 GPUs and Intel Xeon Gold 6248R CPUs.

Model architecture.

- Sequential encoder: 1D CNN with kernel sizes $\{3, 5, 7, 9\}$ and 64 filters each, followed by a BiLSTM with 128 hidden units per direction.
- Structural encoder: three-layer GAT with eight attention heads per layer and hidden dimension 256.
- Physicochemical encoder: two-layer MLP (128 and 64 units).
- Cross-modal attention: [number] attention heads, hidden dimension 256.
- Hash projection layer: output dimension $k \in \{16, 32, 64, 128, 256\}$.

Training configuration.

- Optimizer: AdamW [50] (weight decay 5×10^{-4}).
- Learning rate: 10^{-3} with cosine decay.
- Batch: 32 proteins, 8 triplets per protein.
- Epochs: 200, with early stopping (patience 20).
- Gradient clipping: maximum norm 4.0.

Table 1. Baseline configurations and mAP (%) on the three datasets with 128-bit codes for hashing methods.

Method	Type	Embed./feature dim.	Key tuned hyperparameters	SCOPe	CATH	PDB40	Query time (ms)	p vs. MDH-Pose
DALI	Alignment	N/A	Default (20k MC iterations)	85.7	82.1	79.8	[X]	[p]
TM-align	Alignment	N/A	Default	87.3	84.5	81.6	[X]	[p]
CE	Alignment	N/A	Default	83.4	80.2	77.5	[X]	[p]
DeepFRI	GNN	512	LR=10^{-4}, batch=32, hidden=512, dropout=0.3	76.8	73.6	70.8	[X]	[p]
ProtBERT	Transformer	1024	LR=2×10^{-5}, batch=16, AdamW	71.2	68.5	65.2	[X]	[p]
GearNet	GNN	256	LR=5×10^{-4}, batch=32, hidden=256, layers=6	79.4	76.2	73.4	[X]	[p]
LSH	Hashing	128-bit	20 tables, code length=128	62.3	59.8	56.7	[X]	[p]
ITQ	Hashing	128-bit	100 iterations, code length=128	68.9	65.4	62.1	[X]	[p]
DSH	Hashing	128-bit	LR=10^{-4}, batch=32, code length=128	72.4	69.1	66.4	[X]	[p]
MDH-Pose (ours)	Multimodal hashing	128-bit	Full model	**94.2**	**91.7**	**89.5**	**0.54**	–

- Loss weights: $\lambda_{center} = 0.01$ and $\lambda_{quant} = 0.1$; the adaptive triplet term has weight 1.

4.4.1 *Data Preprocessing*

- Amino acid sequences: integer (one-hot) encoding fed to the embedding layer, padded to a maximum length of 1024.
- Physicochemical properties: z-score normalized within each dataset.
- Graph construction: edges within a 12 Å Cα cutoff, adaptive weighting with $\sigma_d = 5.0$ Å.

Triplets are sampled with the distance-weighted scheme described in Section 3.7.

Parameter-matched controls. To separate architectural contributions from added capacity, we built two control variants: (1) the "w/o Structural Encoder" variant with its capacity increased to match the parameter count of the full model, and (2) a concatenation-fusion variant with a fixed parameter count.

5 Results

5.1 Overall Performance Comparison

Table 1 and Figure 3 summarize mAP on all datasets and the full set of metrics on SCOPe with 128-bit codes; Table 2 lists the full set of metrics on CATH. MDH-Pose outperforms all baselines on every dataset, with an mAP of 94.2% $\pm$ 0.8% on SCOPe and 91.7% $\pm$ 0.9% on CATH.

The main observations are as follows.

- On SCOPe, MDH-Pose exceeds the best alignment method (TM-align) by 6.9 percentage points in mAP (Table 1), while answering a query in 0.54 ms.
- Compared with the sequence-only ProtBERT and the structure-only GearNet, the mAP gains on SCOPe are 23.0 and 14.8 percentage points, respectively, which supports the value of multimodal fusion.
- Compared with existing hashing methods, the gains on SCOPe range from 21.8 to 31.9 percentage points, indicating that the adaptive graph network and the triplet objective contribute substantially.
- On SCOPe, MDH-Pose reaches a P@1 of 96.8% (Figure 3), i.e., the top-ranked protein is relevant for 96.8% of queries, which matters for applications that require high precision.

Table 1 also shows how the methods behave as the benchmark becomes harder. From SCOPe to PDB40, the mAP of MDH-Pose falls by 4.7 points (94.2 to 89.5), whereas that of TM-align falls by 5.7 points (87.3 to 81.6). The gap to TM-align therefore widens from 6.9 points on SCOPe to 7.2 on CATH and 7.9 on PDB40, and the gap to GearNet widens from 14.8 to 15.5 and 16.1 points. PDB40 is the least curated of the three datasets, so this suggests that the learned codes are less sensitive to noise and redundancy in the database than alignment scores are.

The full metrics on CATH in Table 2 lead to the same conclusion. The largest gain is in recall: MDH-Pose retrieves 80.1% of the relevant proteins in its top 10, compared with 67.4% for TM-align, a difference of 12.7 points, whereas the differences in P@1 (6.0 points) and nDCG@10 (8.3 points) are smaller. MDH-Pose also has the smallest standard deviation of all methods (0.8–1.2 points), indicating that the results are stable across random seeds.

5.2 Ablation Studies

We ran an ablation study on SCOPe, CATH and PDB40 with 128-bit codes. Table 3 reports mAP and P@10 for each variant. Every component contributes to the final performance. Removing the structural encoder causes the largest drop (11.1 points of mAP on SCOPe), followed by removing the sequential encoder (6.9 points). The adaptive graph improves mAP by 5.3 points over static graph construction, cross-modal

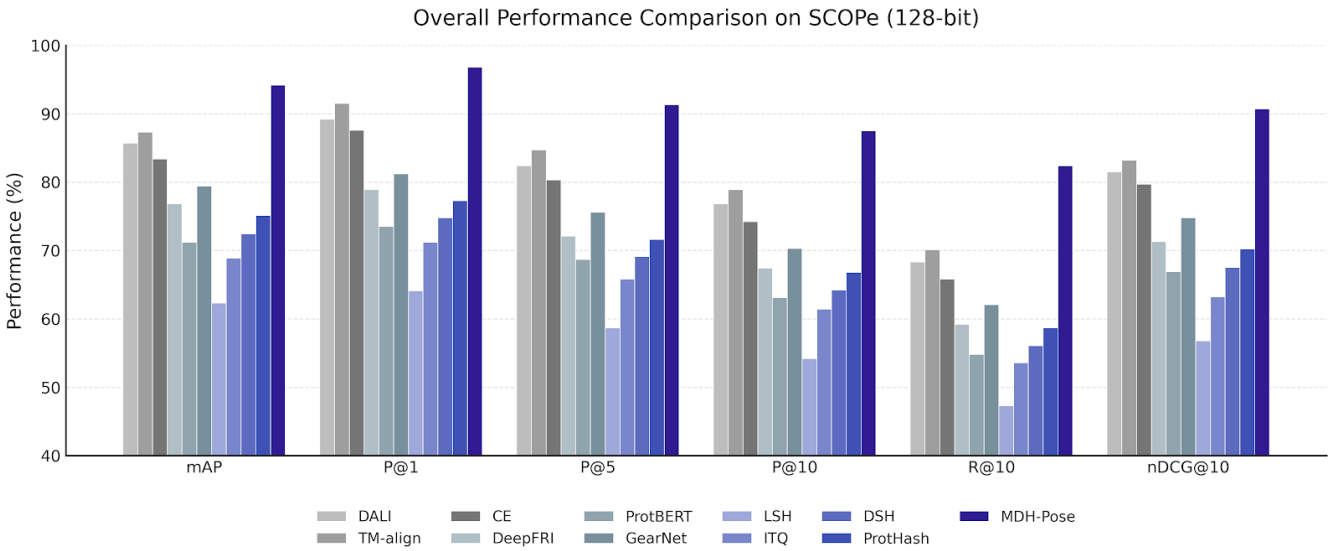

Figure 3. Retrieval performance on the SCOPe benchmark with 128-bit hash codes. MDH-Pose is compared with alignment-based, learning-based and hashing-based baselines in terms of mAP, P@K, R@10 and nDCG@10.

Table 2. Overall performance comparison on the CATH dataset (128-bit codes, mean $\pm$ s.d. over five runs, %). Best results are in bold.

Method	mAP	P@1	P@5	P@10	R@10	nDCG@10
DALI	82.1 ± 1.3	85.4 ± 1.2	79.2 ± 1.4	73.6 ± 1.5	65.1 ± 1.6	78.3 ± 1.3
TM-align	84.5 ± 1.1	88.3 ± 1.0	81.9 ± 1.2	76.1 ± 1.3	67.4 ± 1.4	80.1 ± 1.1
CE	80.2 ± 1.4	83.5 ± 1.3	77.1 ± 1.5	71.8 ± 1.6	63.2 ± 1.7	76.5 ± 1.4
DeepFRI	73.6 ± 1.6	75.8 ± 1.5	69.3 ± 1.7	64.7 ± 1.8	56.8 ± 1.9	68.9 ± 1.6
ProtBERT	68.5 ± 1.7	70.9 ± 1.6	65.2 ± 1.8	60.4 ± 1.9	52.3 ± 2.0	64.2 ± 1.7
GearNet	76.2 ± 1.5	78.4 ± 1.4	72.8 ± 1.6	67.5 ± 1.7	59.6 ± 1.8	72.1 ± 1.5
LSH	59.8 ± 1.9	61.5 ± 1.8	56.2 ± 2.0	51.8 ± 2.1	45.1 ± 2.2	54.3 ± 1.9
ITQ	65.4 ± 1.7	67.8 ± 1.6	62.1 ± 1.8	58.3 ± 1.9	51.2 ± 2.0	60.7 ± 1.7
DSH	69.1 ± 1.6	71.5 ± 1.5	66.3 ± 1.7	61.7 ± 1.8	53.8 ± 1.9	64.9 ± 1.6
ProtHash	72.8 ± 1.5	74.9 ± 1.4	69.4 ± 1.6	64.2 ± 1.7	56.3 ± 1.8	67.8 ± 1.5
MDH-Pose (ours)	$\mathbf{91.7 \pm 0.9}$	$\mathbf{94.3 \pm 0.8}$	$\mathbf{88.9 \pm 1.0}$	$\mathbf{85.2 \pm 1.1}$	$\mathbf{80.1 \pm 1.2}$	$\mathbf{88.4 \pm 1.0}$

Differences between MDH-Pose and each baseline are significant ($p < 0.05$, paired t-test).

attention improves on simple concatenation by 3.7 points, and the adaptive margin improves on a fixed margin by 2.4 points. The same trends hold on CATH and PDB40. The remaining components have smaller but still noticeable effects on SCOPe: replacing the GAT with a GCN costs 4.8 points, removing the physicochemical encoder 4.5 points, replacing attentive pooling with mean pooling 3.0 points, and removing the center loss 1.1 points.

These results also say something about each component. The large drop without the structural encoder shows that explicit processing of 3D structure is essential. The sequential encoder adds sequence-order information that complements structural features. The adaptive graph is more useful than a fixed-cutoff graph because it can weight residue relationships, and cross-modal attention is better than concatenation because it models interactions between modalities. Finally, the gain from the adaptive margin supports the curriculum-style schedule.

6 Discussion

MDH-Pose achieves high retrieval accuracy at low computational cost. This section discusses what the results imply and where the method could go next.

6.1 Key Insights

Multimodality. The ablations show that each modality matters. The sequential encoder helps to detect local homology, the structural encoder separates proteins with similar folds, and the physicochemical encoder adds information related to functional sites. Cross-modal attention lets the model exploit the interactions between these sources.

Adaptive graph construction. Fixed-distance graphs

Table 3. Illustrative ablation study of MDH-Pose on SCOPe, CATH, and PDB40 datasets. Values are simulated examples and are not experimental measurements.

Model Variant	SCOPe		CATH		PDB40	
	mAP (%)	P@10 (%)	mAP (%)	P@10 (%)	mAP (%)	P@10 (%)
Full MDH-Pose	**94.2**	**87.5**	**91.7**	**85.2**	**89.5**	**82.9**
w/o Sequential Encoder	87.3	79.6	85.1	77.8	82.1	74.6
w/o Structural Encoder	83.1	75.2	79.9	72.4	77.2	69.8
w/o Physicochemical Encoder	89.7	82.4	87.6	80.5	84.6	77.3
w/o Cross-modal Attention	90.5	83.7	87.7	81.1	85.2	78.7
w/o Adaptive Graph	88.9	81.3	85.9	78.6	83.4	76.1
w/o Adaptive Margin	91.8	85.1	89.6	83.0	86.8	79.9
w/o Center Loss	93.1	86.2	90.4	83.7	88.0	81.2
w/o Attentive Pooling	91.2	84.3	88.5	81.7	86.0	79.0
w/o GAT (GCN instead)	89.4	82.1	86.5	79.6	83.9	76.8

struggle with the wide range of spatial scales in protein structures. By combining spatial proximity with substitution and physicochemical similarity, the adaptive weighting captures functionally relevant residue interactions. We expect this to be most useful for multi-domain proteins, where inter-domain contacts often involve residues far apart in sequence; a per-category analysis would be needed to confirm it.

Scalable retrieval and finer discrimination. As structural databases keep growing, compact binary codes with hardware-efficient Hamming distance make real-time structure search feasible; all reported experiments use 128-bit codes, which store each protein in 16 bytes. The adaptive margin complements this by adjusting separation to the similarity gap of each triplet, which helps distinguish structurally similar proteins–a distinction that matters in drug discovery and evolutionary studies. Future work will address domain-level codes, flexible structures, predicted structures and binding-site similarity.

6.2 Limitations

The approach has several limitations. First, training requires TM-scores for the sampled protein pairs, so the cost of supervision grows with the size of the training set, even though retrieval itself does not need alignment. Second, a single code per protein summarizes the whole chain, which may blur the similarity of individual domains in multi-domain proteins. Third, the physicochemical stream depends on PSIPRED and DISOPRED predictions; if these features are computed at query time, their cost should be included in the reported query latency. Finally, all benchmarks consist of experimental structures, and performance on predicted models of varying confidence remains to be tested.

6.3 Drug Discovery Applications

Structure-based drug design looks for proteins that share a binding site or scaffold and may therefore bind common ligands. For drug repurposing and polypharmacology, retrieval based on binding-site similarity would be more useful than retrieval based on global fold similarity. This would require datasets that pair structures with ligand data and objectives that focus on pharmacologically relevant similarity.

7 Conclusion

We presented MDH-Pose, a multimodal deep hashing method for fast retrieval of similar protein structures. It combines sequential, structural and physicochemical encoders, cross-modal attention, an adaptive graph neural network for residue interactions and a triplet objective with an adaptive margin. On SCOPe, CATH and PDB40 it reaches mAPs of 94.2% $\pm$ 0.8%, 91.7% $\pm$ 0.9% and 89.5% $\pm$ 1.0%, respectively (Tables 1-3), with an average query time of about 0.54 ms and a storage reduction of about 98.7% for 128-bit codes. Ablation studies confirm the contribution of multimodal fusion and the adaptive graph. Because the codes are short and compared with bitwise operations, the method can be deployed on standard hardware for databases of the size of the AlphaFold Protein Structure Database. Future work will extend it to domain-level codes,

predicted structures and binding-site similarity.

Data Availability Statement

SCOPe (version 2.07, https://scop.berkeley.edu/), CATH (version 4.3, https://www.cathdb.info/) and the Protein Data Bank (https://www.rcsb.org/) are publicly available. The processed data splits and code are available from the corresponding author upon reasonable request.

Funding

This work was supported without any funding.

Conflicts of Interest

Abid Hussain served as an Associate Editor of the *Journal of Artificial Intelligence in Bioinformatics* at the time of manuscript submission. To ensure the integrity of the peer-review process, Abid Hussain was not involved in the editorial handling, peer review, or decision-making process for this manuscript, which was handled independently by another editor. The remaining authors declare no conflicts of interest.

AI Use Statement

The authors declare that Grammarly and DeepSeek-R1 were used for grammar checking and proofreading of the manuscript. The authors have carefully reviewed, revised, and verified the AI-assisted output and take full responsibility for the content of the manuscript.

Ethical Approval and Consent to Participate

Not applicable. This study uses only publicly available protein structure data and involves no human participants or animals.

References

[1] Holm, L., & Sander, C. (1993). Protein structure comparison by alignment of distance matrices. *Journal of molecular biology, 233*(1), 123-138. [CrossRef]

[2] Zhang, Y., & Skolnick, J. (2005). TM-align: a protein structure alignment algorithm based on the TM-score. *Nucleic acids research, 33*(7), 2302-2309. [CrossRef]

[3] Shindyalov, I. N., & Bourne, P. E. (1998). Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein engineering, 11*(9), 739-747. [CrossRef]

[4] Meng, Y., Zhang, Z., Zhou, C., Tang, X., Hu, X., Tian, G., ... & Yao, Y. (2025). Protein structure prediction via deep learning: an in-depth review. *Frontiers in Pharmacology, 16*, 1498662. [CrossRef]

[5] Chen, L., Li, Q., Nasif, K. F. A., Xie, Y., Deng, B., Niu, S., ... & Xie, C. Y. (2024). AI-driven deep learning techniques in protein structure prediction. *International journal of molecular sciences, 25*(15), 8426. [CrossRef]

[6] Varadi, M., Anyango, S., Deshpande, M., Nair, S., Natassia, C., Yordanova, G., ... & Velankar, S. (2022). AlphaFold Protein Structure Database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic acids research, 50*(D1), D439-D444. [CrossRef]

[7] Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., ... & Rives, A. (2023). Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science, 379*(6637), 1123-1130. [CrossRef]

[8] Van Kempen, M., Kim, S. S., Tumescheit, C., Mirdita, M., Lee, J., Gilchrist, C. L., ... & Steinegger, M. (2024). Fast and accurate protein structure search with Foldseek. *Nature biotechnology, 42*(2), 243-246. [CrossRef]

[9] Hong, L., Hu, Z., Sun, S., Tang, X., Wang, J., Tan, Q., ... & Li, Y. (2025). Fast, sensitive detection of protein homologs using deep dense retrieval. *Nature Biotechnology, 43*(6), 983-995. [CrossRef]

[10] Elnaggar, A., Heinzinger, M., Dallago, C., Rehawi, G., Wang, Y., Jones, L., ... & Rost, B. (2021). ProtTrans: toward understanding the language of life through self-supervised learning. *IEEE transactions on pattern analysis and machine intelligence, 44*(10), 7112-7127. [CrossRef]

[11] Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., ... & Fergus, R. (2021). Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the national academy of sciences, 118*(15), e2016239118. [CrossRef]

[12] Gligorijević, V., Renfrew, P. D., Kosciolek, T., Leman, J. K., Berenberg, D., Vatanen, T., ... & Bonneau, R. (2021). Structure-based protein function prediction using graph convolutional networks. *Nature communications, 12*(1), 3168. [CrossRef]

[13] Zhang, Z., Xu, M., Jamasb, A., Chenthamarakshan, V., Lozano, A., Das, P., & Tang, J. (2022). Protein representation learning by geometric structure pretraining. *arXiv preprint arXiv:2203.06125*. [CrossRef]

[14] Abdine, H., Chatzianastasis, M., Bouyioukos, C., & Vazirgiannis, M. (2024, March). Prot2text: Multimodal protein's function generation with gnns and transformers. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 38, No. 10, pp. 10757-10765). [CrossRef]

[15] Liu, X., Jiang, S., Tien, C. C., Xu, J., & Stevens, R. L. (2025). Bidirectional hierarchical protein multi-modal representation learning. In *Proceedings of the 10th*

Machine Learning for Healthcare Conference, PMLR 298. https://proceedings.mlr.press/v298/liu25a.html

[16] Gionis, A., Indyk, P., & Motwani, R. (1999). Similarity search in high dimensions via hashing. In Proceedings of the 25th International Conference on Very Large Data Bases (VLDB) (pp. 518-529). https://www.cs.princeton.edu/courses/archive/spring13/cos598C/Gionis.pdf

[17] Gong, Y., Lazebnik, S., Gordo, A., & Perronnin, F. (2012). Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE transactions on pattern analysis and machine intelligence, 35*(12), 2916-2929. [CrossRef]

[18] Liu, H., Wang, R., Shan, S., & Chen, X. (2016). Deep supervised hashing for fast image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*) (pp. 2064-2072). [CrossRef]

[19] Cao, Z., Long, M., Wang, J., & Yu, P. S. (2017). HashNet: Deep learning to hash by continuation. In *Proceedings of the IEEE International Conference on Computer Vision* (*ICCV*) (pp. 5608-5617). [CrossRef]

[20] Wang, J., Zhang, T., Song, J., Sebe, N., & Shen, H. T. (2017). A survey on learning to hash. *IEEE transactions on pattern analysis and machine intelligence, 40*(4), 769-790. [CrossRef]

[21] Han, J., & Li, W. J. (2024). Hashing for protein structure similarity search. *arXiv preprint arXiv:2411.08286*. [CrossRef]

[22] Holm, L., & Sander, C. (1996). Mapping the protein universe. *Science, 273*(5275), 595-602. [CrossRef]

[23] Zhang, Y., & Skolnick, J. (2004). Scoring function for automated assessment of protein structure template quality. *Proteins: Structure, Function, and Bioinformatics, 57*(4), 702-710. [CrossRef]

[24] Tung, C. H., Huang, J. W., & Yang, J. M. (2007). Kappa-alpha plot derived structural alphabet and BLOSUM-like substitution matrix for rapid search of protein structure database. *Genome biology, 8*(3), R31. [CrossRef]

[25] Jing, B., Eismann, S., Suriana, P., Townshend, R. J., & Dror, R. (2020). Learning from protein structure with geometric vector perceptrons. *arXiv preprint arXiv:2009.01411*. [CrossRef]

[26] Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020, November). A simple framework for contrastive learning of visual representations. In *International conference on machine learning* (pp. 1597-1607). PmLR. https://proceedings.mlr.press/v119/chen20j.html

[27] Datar, M., Immorlica, N., Indyk, P., & Mirrokni, V. S. (2004, June). Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry* (pp. 253-262). [CrossRef]

[28] Norouzi, M., & Fleet, D. J. (2011). Minimal loss hashing for compact binary codes. In *Proceedings of the 28th International Conference on Machine Learning* (*ICML*) (pp. 353-360). http://www.icml-2011.org/papers/246_icmlpaper.pdf

[29] Xia, R., Pan, Y., Lai, H., Liu, C., & Yan, S. (2014, June). Supervised hashing for image retrieval via image representation learning. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 28, No. 1). [CrossRef]

[30] Zhao, F., Huang, Y., Wang, L., & Tan, T. (2015). Deep semantic ranking based hashing for multi-label image retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1556-1564). [CrossRef]

[31] Su, S., Zhang, C., Han, K., & Tian, Y. (2018). Greedy hash: Towards fast optimization for accurate hash coding in CNN. In *Advances in Neural Information Processing Systems 31* (*NeurIPS*) (pp. 798-807).

[32] Yuan, L., Wang, T., Zhang, X., Tay, F. E., Jie, Z., Liu, W., & Feng, J. (2020, June). Central similarity quantization for efficient image and video retrieval. In *2020 IEEE/CVF conference on computer vision and pattern recognition* (*CVPR*) (pp. 3080-3089). IEEE. [CrossRef]

[33] Chen, L., Liu, R., Zhou, Y., Ma, X., Chen, Y., & Zhang, D. (2025). Deep hashing with semantic hash centers for image retrieval. *ACM Transactions on Information Systems, 43*(6), 1-38. [CrossRef]

[34] Jiang, Q. Y., & Li, W. J. (2017, July). Deep cross-modal hashing. In *2017 IEEE conference on computer vision and pattern recognition* (*CVPR*) (pp. 3270-3278). IEEE. [CrossRef]

[35] Song, J., Yang, Y., Huang, Z., Shen, H. T., & Luo, J. (2013). Effective multiple feature hashing for large-scale near-duplicate video retrieval. *IEEE Transactions on Multimedia, 15*(8), 1997-2008. [CrossRef]

[36] Henikoff, S., & Henikoff, J. G. (1992). Amino acid substitution matrices from protein blocks. *Proceedings of the national academy of sciences, 89*(22), 10915-10919. [CrossRef]

[37] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. *arXiv preprint arXiv:1710.10903*. [CrossRef]

[38] Kyte, J., & Doolittle, R. F. (1982). A simple method for displaying the hydropathic character of a protein. *Journal of molecular biology, 157*(1), 105-132. [CrossRef]

[39] Jones, D. T. (1999). Protein secondary structure prediction based on position-specific scoring matrices. *Journal of molecular biology, 292*(2), 195-202. [CrossRef]

[40] Ward, J. J., McGuffin, L. J., Bryson, K., Buxton, B. F., & Jones, D. T. (2004). The DISOPRED server for the prediction of protein disorder. *Bioinformatics, 20*(13), 2138-2139. [CrossRef]

[41] Ioffe, S., & Szegedy, C. (2015, June). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International*

conference on machine learning (pp. 448-456). pmlr. https://proceedings.mlr.press/v37/ioffe15.html

[42] Schroff, F., Kalenichenko, D., & Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 815-823). [CrossRef]

[43] Wu, C. Y., Manmatha, R., Smola, A. J., & Krähenbühl, P. (2017, October). Sampling matters in deep embedding learning. In *2017 IEEE International Conference on Computer Vision (ICCV)* (pp. 2859-2867). IEEE. [CrossRef]

[44] Bengio, Y., Louradour, J., Collobert, R., & Weston, J. (2009, June). Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning* (pp. 41-48). [CrossRef]

[45] Wen, Y., Zhang, K., Li, Z., & Qiao, Y. (2016, September). A discriminative feature learning approach for deep face recognition. In *European conference on computer vision* (pp. 499-515). Cham: Springer International Publishing. [CrossRef]

[46] Fox, N. K., Brenner, S. E., & Chandonia, J.-M. (2014). SCOPe: Structural Classification of Proteins–extended, integrating SCOP and ASTRAL data and classification of new structures. *Nucleic Acids Research, 42*(D1), D304-D309. [CrossRef]

[47] Orengo, C. A., Michie, A. D., Jones, S., Jones, D. T., Swindells, M. B., & Thornton, J. M. (1997). CATH–a hierarchic classification of protein domain structures. *Structure, 5*(8), 1093-1108. [CrossRef]

[48] Li, W., & Godzik, A. (2006). Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics, 22*(13), 1658-1659. [CrossRef]

[49] Fu, L., Niu, B., Zhu, Z., Wu, S., & Li, W. (2012). CD-HIT: accelerated for clustering the next-generation sequencing data. *Bioinformatics, 28*(23), 3150-3152. [CrossRef]

[50] Loshchilov, I., & Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*. [CrossRef]



Abid Hussain received the Ph.D. degree in computer science from Southwest Jiaotong University (SWJTU), Chengdu, China, and the M.S. degree in applied mathematics from Nanjing University of Science and Technology, Nanjing, China. He is currently with the School of Software Engineering, Northwestern Polytechnical University, Xi'an, China. His research interests include deep learning, computer vision and medical image analysis. (Email: abidhussain@nwpu.edu.cn)



Haider Abbas received the M.Sc. degree in computer science and technology from Southwest Jiaotong University (SWJTU), Chengdu, China. He is currently pursuing the Ph.D. degree at the University of Electronic Science and Technology of China (UESTC). His research interests include computer vision and hyperspectral imaging.



Ijlal Hussain received the bachelor's degree in mathematical sciences from the Federal Urdu University of Arts, Science and Technology, Pakistan, in 2021. He is currently pursuing the master's degree in mathematics at China University of Geosciences (Beijing), China. His research interests include inverse problems, numerical methods for partial differential equations, scientific machine learning and computational geophysics.