LATTE: A Structure-Aware Latent Model for Protein Sequence Embedding and Search

Danny Ahn[1*], Minjae Lee[1], Sihyeon Moon[1], Jooyoung Jung[1]

[1*]Biology, Daegu Science High School, 154 Dongdaegu-ro, Daegu, 42111, Republic of Korea.

Contributing authors: ahnd6474@gmail.com;

Abstract

Background: Large protein language models (PLMs) such as ESM-2 yield structure-sensitive sequence embeddings but are expensive to store and query at scale. We sought a compact latent model that preserves structural signal while supporting property prediction, and approximate sequence search.

Results: We introduce LATTE, a structure-aware encoder trained on UniRef50 with a composite loss that couples sequence reconstruction, Kullback–Leibler regularization, and a structural term supervising inter-residue similarity. LATTE reconstructs unseen UniRef50 sequences with 97.17% accuracy and remains robust to injected latent noise. Gaussian process models trained on the 256-dimensional latent space classify fluorescent proteins (FPs) versus non-FPs with five-fold cross-validation accuracy of 0.987 and predict excitation/emission peaks with root mean square errors of 2.70 nm and 3.80 nm, matching an ESM-2 (650M) embedding baseline while using a far smaller parameter budget. On a matched subset, pairwise cosine distance distributions are broader and heavier-tailed for LATTE than for ESM-2, yet the two spaces exhibit strong rank concordance (Spearman $\boldsymbol{\rho = 0.761}$) and similar $\boldsymbol{k = 3}$ clustering partitions, indicating preserved neighbour ordering with expanded dynamic range. Building on this representation, we construct the LATTE Latent Alignment Search Tool (LLAST), a latent-tree index used as a prefilter before BLAST. On a protein database of approximately $\mathbf{10^6}$ sequences, LLAST restricts the candidate set forwarded to BLAST to at most $\mathbf{10^4}$ sequences per query ($\boldsymbol{\approx}$ **0.97%** of the database) while recovering around 55% of baseline BLAST top-50 hits, demonstrating an aggressive but tunable trade-off between recall and workload reduction.

Conclusions: LATTE provides a compact, structure-aligned latent space that supports accurate reconstruction, downstream property prediction, and tree-based prefiltering prior to alignment. This combination suggests a general path toward lightweight, task-aware indices for scalable protein sequence search and design.

Keywords: protein language model; structural loss; fluorescent proteins; embedding geometry; cosine distance; bioinformatics

1 Background

1.1 Protein

Proteins are linear polymers of amino acids linked by peptide bonds, whose three dimensional conformations determine their biochemical functions. [1–3] The primary structure, the specific sequence of 20 canonical amino acids dictates folding pathways through intramolecular interactions such as hydrogen bonds, hydrophobic packing, electrostatic attractions, and van der Waals forces. These interactions give rise to secondary motifs (α helices, β sheets), tertiary folds, and quaternary assemblies in multimeric complexes.

1.2 Deep Learning

Deep learning accelerated protein modeling by learning transferable sequence representations at scale. We defer architectural details to Methods and focus here on the motivation and role of the latent space in design. Deep learning has transformed protein modeling by learning high dimensional sequence representations that implicitly encode structural and functional constraints from large unlabeled corpora. Masked language model PLMs (e.g., ESM-2) capture long range residue couplings and, via ESMFold, enable accurate single sequence structure prediction; ESM-3 further couples sequence, structure, and function for editing/design. However, these foundation models carry latency and memory costs that impede large scale screening and iterative design. To complement them, we adopt a compact VAE [4] style encoder–decoder whose latent variables remain active and informative, regularized by perceptual losses against ESMS so that geometry aligns with structure. The resulting latents support fast clustering/retrieval and downstream property models while remaining competitive for transfer at a fraction of the compute, and they serve as the basis for our pre alignment pruning (LLAST) that reduces search without sacrificing interpretability.

1.3 ESM-2 and ESMS

Using pretrained ESM-2 embeddings, we developed ESMS that reduces inference time relative to ESM-2's 0.5 s per sequence while preserving high embedding fidelity. ESM-2, introduced by Rives et al., is a masked language model that captures short and long range residue interactions [5], and it underpins ESMFold, a three dimensional structure predictor with performance comparable to AlphaFold 2 [6, 7], and is widely used for downstream tasks such as secondary structure and thermostability prediction. Although available at multiple parameter scales (650M parameters being most common), ESM-2's latency motivates lighter alternatives. On a held out test set, ESMS achieved cosine similarity of 0.9647 and RMSE of 1.2998.

We position our approach within recent progress on large protein language models. In particular, ESM-3 tightly couples sequence, structure, and function supervision and demonstrates design/editing capabilities beyond prior ESM-2 systems [8]. LATTE is intended as a lightweight, structure informed encoder that complements such foundation models by emphasizing controllable latent structure and efficient training.

1.4 BLAST

BLAST (Basic Local Alignment Search Tool) locates local similarities via a *seed and extend* heuristic, quickly forming high scoring segment pairs (HSPs) [9]. The statistical significance of alignments is modeled by the Karlin-Altschul framework, yielding the E-value and bit score [10]. While less sensitive than optimal dynamic programming alignment, BLAST scales effectively to large databases and remains the practical standard; the BLAST+ re architecture further streamlined performance and modularity [11].

1.5 Deep Learning-based Search Algorithms for Protein Retrieval

Transformer-based protein language models map an input sequence x to an embedding $z = f_\theta(x) \in \mathbb{R}^d$ [12]. Given a database $\mathcal{D} = \{z_i\}_{i=1}^N$ and a query z_q, protein retrieval is formulated as nearest-neighbor search under a similarity score $s(z_q, z_i)$ such as cosine similarity or an inner product in the maximum inner product search (MIPS) setting. A naive top-k search evaluates $s(z_q, z_i)$ for all N items, with $O(Nd)$ time per query, and becomes prohibitive as N grows.

Learning-based search algorithms alleviate this cost by building an index over $\mathcal{D}$ that exploits the geometry of the embedding space. Graph-based methods connect each database point to a small set of neighbors and perform a greedy search that walks toward higher-similarity nodes while exploring only a limited frontier, so the number of similarity evaluations is governed by graph degree and search depth rather than directly by N. Cluster-based methods instead partition the space into K cells via a coarse quantizer and probe only a few cells nearest to z_q, reducing the scanned candidates from N to roughly (N/K) `nprobe`; variants combine this with product quantization or hashing to further accelerate distance computation.

In large retrieval systems, such indices are typically used as the first stage of a cascade. The index returns a candidate set $\mathcal{S} \subset \mathcal{D}$ of size $M \ll N$, and a more accurate but slower scorer is applied only on $\mathcal{S}$. For protein retrieval, this corresponds to using the embedding-based index to discard clearly unrelated regions of sequence space and then running alignment tools such as BLAST on the reduced candidate set. Hyperparameters such as graph degree, probe budget, and search depth are tuned to balance recall at top-k against average and tail latency.

1.6 Our Contribution

It has been shown that transformer architectures are capable of capturing hidden information in a sequence. Also, rather than training big transformer models, using a pretrained model as an embedding was demonstrated to be effective. However, only

training with a sequence has clear limits on generalization, function, and versatility. LATTE, to overcome such limits, introduced a custom loss function that explicitly forces it to learn and represent structural information in the latent vector space.

BLAST, the NCBI sequence search system, is highly informative but its per query runtime scales roughly linearly with database size. In practice it still scans large portions of the library for each query, repeatedly revisiting near duplicate or highly similar sequences. To address this, we introduce a LATTE based prefilter that encodes sequences into a latent space and prunes unrelated entries prior to alignment. By removing off target candidates early, the prefilter reduces the search set and improves BLAST's efficiency and speed while preserving downstream alignment behavior.

Unlike prior work that (i) uses large PLMs as fixed feature extractors, (ii) trains autoencoders primarily for generative design, or (iii) accelerates BLAST using purely sequence-based filters or generic ANN indices, we focus on a compact, structure-aware latent space that is explicitly reused for both downstream modeling and a tree-based prefilter.

2 Methods

2.1 Shaping the latent space

We trained LATTE with a pseudo decoder which acts as a training signal generator that receives encoder memory and latent vectors. Moreover, we enforce structural consistency via a perceptual loss computed from pretrained ESMS embeddings, which capture residue position regularities at substantially lower cost than full structure predictors. Given an original sequence $\mathbf{x}_{\text{orig}}$ and its reconstruction $\mathbf{x}_{\text{recon}}$, we form two embedding based terms.

$$L_{\text{COS}} = \Big[1 - \cos\big(\text{ESMS}(\mathbf{x}_{\text{orig}}),\, \text{ESMS}(\mathbf{x}_{\text{recon}})\big)\Big], \tag{1}$$

$$L_{\text{MSE}} = \big\|\text{ESMS}(\mathbf{x}_{\text{orig}}) - \text{ESMS}(\mathbf{x}_{\text{recon}})\big\|_2^2. \tag{2}$$

These structural terms are combined with next token cross entropy L_{CE} (teacher forcing) and KL divergence L_{KL} into

$$L_1 \;=\; \lambda\,(L_{\text{COS}} + L_{\text{MSE}}) \;+\; \alpha\,L_{\text{CE}} \;+\; \beta\,L_{\text{KL}} \tag{3}$$

with $\lambda = 5$, α linearly decayed from 30 to 0.1 over the first 100 epochs, and β linearly annealed from 0 to 0.1 over the same interval. The cosine term tolerates substitutions among similarly likely residues, while the MSE term penalizes large deviations. Together, they discourage posterior collapse by making accurate reconstruction contingent on informative latents. [13].

LATTE is a comparably lightweight transformer with 5.5M parameters, composed of 4 layer transformer encoders (Figure 1). The hyperparameters are detailed in Table 1. The Adam optimizer [14] was used.

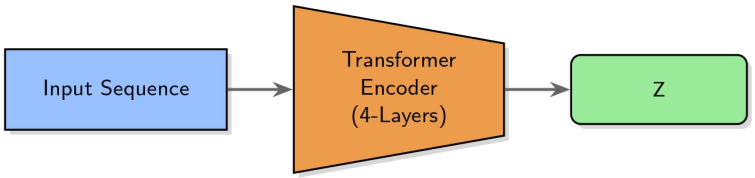


Fig. 1: The architecture of LATTE. An input sequence is processed by a 4 layer Transformer Encoder, which outputs the parameters (mean and log variance) of the latent distribution. A latent vector is sampled and then reconstructed into the output sequence by a 4 layer Transformer Decoder.

Table 1: Hyperparameters for LATTE.

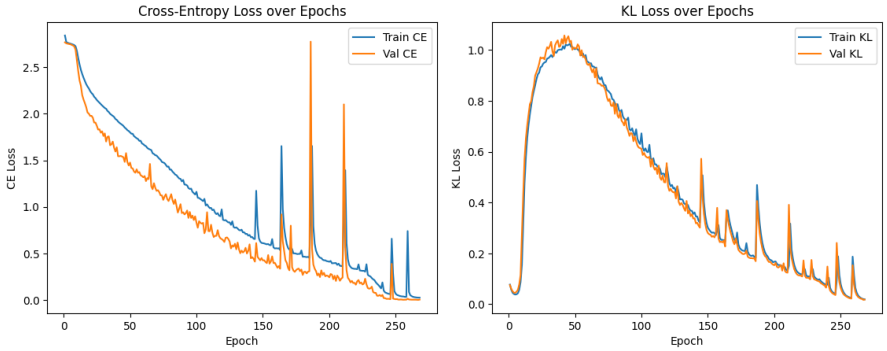
	Vocab Size	d_model	Latent Dim	n_heads	Feed Forward	Dropout
Value	33	256	256	4	512	0.3

2.2 Model Selection

Model selection favored the epoch 380 checkpoint (LATTE _ 380), which balanced the KL divergence near the active threshold (0.048) and achieved the lowest validation cross entropy loss. The model was trained using two T4 GPU sessions provided by Kaggle on a random subsample of the UniRef50 dataset. Monitored learning curves did not show any signs of overfitting (Supplementary Figure S1). Although the epoch 500 model (LATTE _ 500) reached a reconstruction rate of 99.976% with validation losses of Val CE = 0.000, COS = 0.003, MSE = 0.007, and KL = 0.002, its very low KL value suggested potential posterior collapse (Supplementary Figures S2a and S2b). Adding noise to the latent space did not significantly affect reconstruction performance. Instead, epoch 380 was chosen because its KL divergence (0.048) was closer to the recommended active value of 0.05, and it exhibited the lowest Val CE. At epoch 380, validation losses were Val CE = 0.072, COS = 0.010, MSE = 0.020, and KL = 0.048 (Supplementary Figures S3a and S3b). Figure 2(a) and (b) illustrate the epoch wise progression of CE and KL values, confirming that epoch 380 offers the optimal balance.

2.3 LLAST

Unlike generic Approximate nearest-neighbor (ANN) methods such as locality-sensitive hashing [15] or product quantization [16], which operate on fixed embeddings, LLAST builds a task-specific hierarchical index over the structure-informed LATTE latent space. This allows us to reuse the same latent representation for generative design, property prediction, and sequence search.



(a) CE over epochs (b) KL over epochs

Fig. 2: CE and KL values over epochs with structural loss.

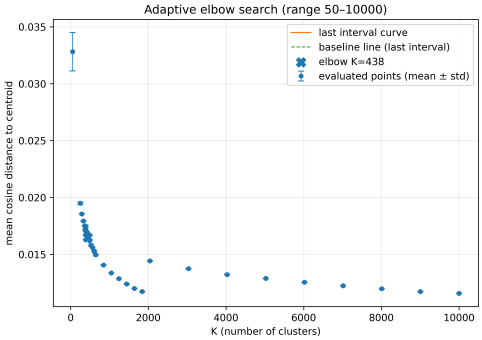


Fig. 3: Adaptive Elbow Search

We introduced Latte Latent Alignment Search Tool(LLAST). We clustered protein sequences in a latent space using k means under cosine distance (via L2 row normalization), then applied agglomerative hierarchical clustering [17] to obtain a tree index. This tree serves as a prefilter, pruning sequences with dissimilar latent vectors before downstream alignment. The number of clusters K was selected by Kneedle knee detection on the mean cosine k means cost curve, with adaptive 10 way interval refinement around the elbow.

3 Results

3.1 Case Study: Fluorescent Protein (FP) Analysis and Generation

We used pretrained ESM-2 embeddings. Using manually curated FP sequences from FPbase [18], we trained Gaussian process [19] (GP) models to (1) classify FPs versus

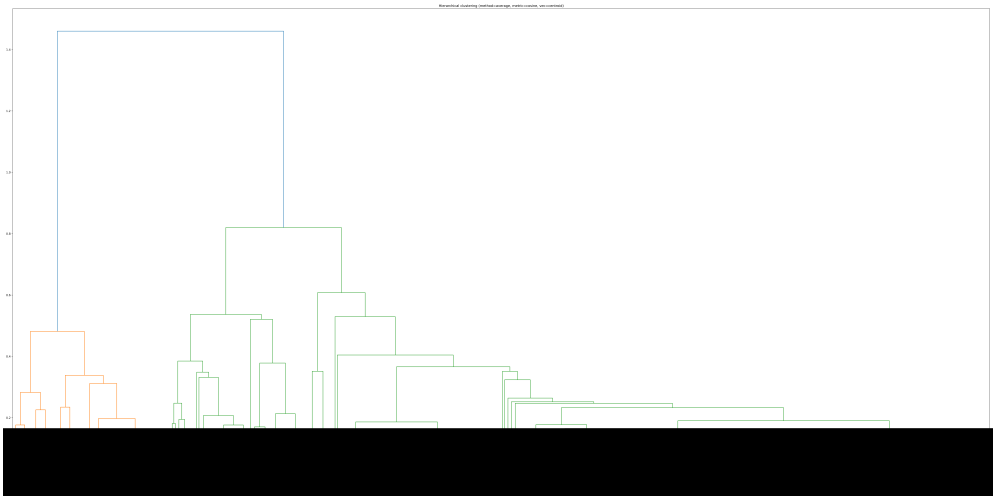


Fig. 4: Tree dendogram

Table 2: GFP spectral prediction pipelines on different embeddings.

Embedding	Classifier metric (5 fold)	λ_{abs} RMSE (nm)	λ_{em} RMSE (nm)
ESM-2 (650M)	AUC 0.997	2.70	3.80
LATTE latent (256 d, ∼5.5M)	AUC 0.987	2.70	3.80

non FPs and (2) regress their spectral peaks. The GP classifier achieved 0.987 accuracy under fivefold cross validation, while the GP regressor achieved root mean square errors of 2.70 nm for maximum absorption (A_{abs}) and 3.80 nm for emission (λ_{em}). Classification reports on both training and test sets confirm strong performance without signs of overfitting (Tables 3 and 4).

ESM-2 baseline and comparison. With the same Gaussian process (GP) pipelines, an *ESM-2 (650M)* embedding baseline produced strong discriminative and regression performance: 5 fold AUC 0.997 and wavelength RMSEs of 2.70/3.80 nm for $\lambda_{abs}/\lambda_{em}$. Using *LATTE*'s 256 d latent embeddings, the FP vs. non FP classifier reached 5 fold accuracy 0.987. This indicates that LATTE's compact latent space provides competitive downstream signal at roughly two orders of magnitude fewer parameters and substantially lower inference cost, while ESM-2 serves as an upper bound reference for fully supervised spectral peak regression.

The latent space visualization via t-SNE clearly separates fluorescent from non fluorescent proteins and encodes spectral properties as continuous gradients, confirming that structural information is captured effectively [20] (Supplementary Figure S4). Non fluorescent proteins cluster on the inner side of two curves, whereas fluorescent

Table 3: Classification report on the training set.

	Precision	Recall	F1-score	Support
Non FP (0)	0.9920	0.9940	0.9930	501
FP (1)	0.9880	0.9840	0.9860	250
Accuracy			0.9907	751
Macro Avg	0.9900	0.9890	0.9895	751
Weighted Avg	0.9907	0.9907	0.9907	751

Table 4: Classification report on the test set.

	Precision	Recall	F1-score	Support
Non FP (0)	0.9840	0.9840	0.9840	125
FP (1)	0.9683	0.9683	0.9683	63
Accuracy			0.9787	188
Macro Avg	0.9761	0.9761	0.9761	188
Weighted Avg	0.9787	0.9787	0.9787	188

proteins occupy the outer region. Moreover, color gradients corresponding to emission and absorption wavelengths indicate that proteins with similar spectral peaks are grouped together.

k-means clustering of the projected FP embeddings [21] revealed three distinct clusters (Table 5), whose consensus vectors were decoded into novel sequences truncated to each cluster's mean length. Classification of these generated sequences using the trained GP classifier showed that Cluster 1 achieved a 100% success rate (Supplementary Figure S5), suggesting that intra cluster sample quality and consistency are more critical than sample size for producing functional proteins. The generated proteins exhibited 75% - 96% identity with FPbase sequences, reflecting the high similarity of their latent representations.

Table 5: Statistics for FP clusters identified by k-means.

Cluster	n (samples)	Mean Length	Std (Length)
0	22	316	5.28
1	318	234	10.00
2	14	118	29.71

Close vectors in the latent space encode proteins with similar three dimensional structures, as confirmed by AlphaFold predictions. Cluster 1 proteins exhibited classic β barrel folds with pLDDT scores above 90; Cluster 0 proteins formed two groups of

Table 6: Pairwise cosine distance (1 − cosine similarity) summary for LATTE vs. ESM-2 embeddings.

Embedding	n	Mean	SD	p_{10} / p_{50} / p_{90}
LATTE (latent)	49,847	0.1694	0.3428	0.00257 / 0.01406 / 0.90065
ESM-2 (650M)	49,847	0.0381	0.0822	0.00365 / 0.00953 / 0.11329

three internal β strands wrapped by α helices; and although Cluster 2 proteins did not adopt β barrels, they nonetheless shared a common fold of three β strands surrounded by α helices.

3.2 Embedding geometry of LATTE vs. ESM-2 (cosine distance)

To compare the global structure of the embedding spaces, we computed pairwise cosine distances (1 − cosine similarity) on matched subsets and summarized both the distributions and their empirical cumulative distribution functions (ECDFs). LATTE latent distances are broader with a heavier tail (mean = 0.1694, SD = 0.3428, $p50$ = 0.0141, $p90$ = 0.9006; n = 49,847), whereas ESM-2 distances are tighter (mean = 0.0381, SD = 0.0822, $p50$ = 0.00953, $p90$ = 0.1133; n = 49,847) (Figure 5, Table 6). This suggests that ESM-2 clusters points more compactly, while LATTE yields a more diffuse geometry that can increase recall for remote neighbours; in practice, we combine LATTE's structure-aware prefilter with sequence alignment ("LLAST") to recover interpretability and precision.

Pairwise cosine distance matrices for the 354 sequence subset in the two embedding spaces. Left: ESM; right: LATTE. Brighter horizontal/vertical bands mark sequences or clusters with larger distances to many others, highlighting global structure and potential outliers. (Figure 7)

To directly compare the pairwise geometries, we plotted ESM (y) versus LATTE (x) cosine distances for matched pairs (Figure 6). Distances are systematically smaller in ESM (slope b=0.125; intercept a=0.017), yet the rank correlation remains high (Spearman ρ=0.761), supporting that LATTE preserves ESM's neighbor ordering while expanding the dynamic range, a property we exploit for latent prefiltering in LLAST.

We report a rerun of k=3 clustering to compare LATTE and ESM-2 embeddings. Silhouette scores (cosine) favor LATTE, and cross-partition agreement metrics indicate strong consistency between the two representations[22].(Table 8, 10, 9)

3.3 Ablation Study

An ablation study was conducted with the loss function

$$L = \alpha CE + \beta KL,$$

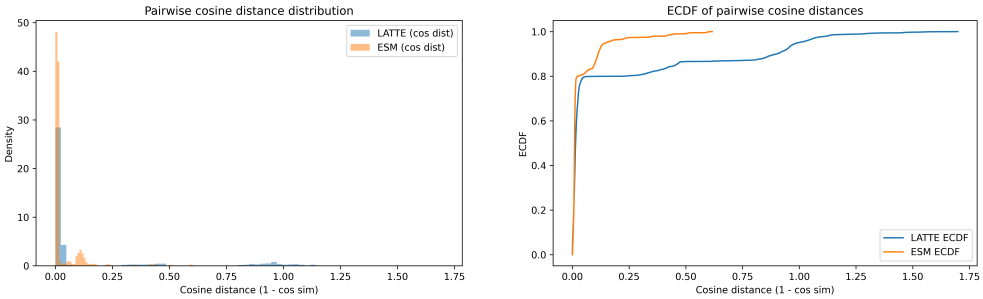


Fig. 5: Pairwise cosine distance distributions (left, density histograms) and ECDFs (right) for LATTE latent vs. ESM-2 embeddings.

Table 7: Silhouette (cosine), k=3.

Metric	LATTE	ESM-2
Silhouette (cosine)	0.9431	0.9022

where the weights are set to $\alpha = 30, \beta = 0$ for epochs ≤ 30, and $\alpha = 0.1, \beta = 0.1$ thereafter. In this setup, KL vanishing occurred right after epoch 100, demonstrating the crucial role of the structural loss term.

Figure 8 shows the CE and KL curves when no structural loss is used, and Figure 2 shows the corresponding curves with structural loss incorporated. Only the latter prevents KL collapse and yields more stable convergence.

Table 8: Summary of k=3 clustering agreement.

Metric	LATTE	ESM-2
Cross partition agreement (LATTE vs. ESM-2)		
Adjusted Rand Index (ARI)	0.7373	
Adjusted Mutual Information (AMI)	0.6055	
Fowlkes-Mallows Index (FMI)	0.9596	
Variation of Information (VI)	0.3529	
Purity (LATTE $\rightarrow$ ESM-2)	0.9633	

Table 9: Confusion matrix (rows: LATTE aligned clusters; columns: ESM-2 clusters).

	ESM-2-0	ESM-2-1	ESM-2-2
LATTE-0	0	0	1
LATTE-1	0	23	1
LATTE-2	6	5	318

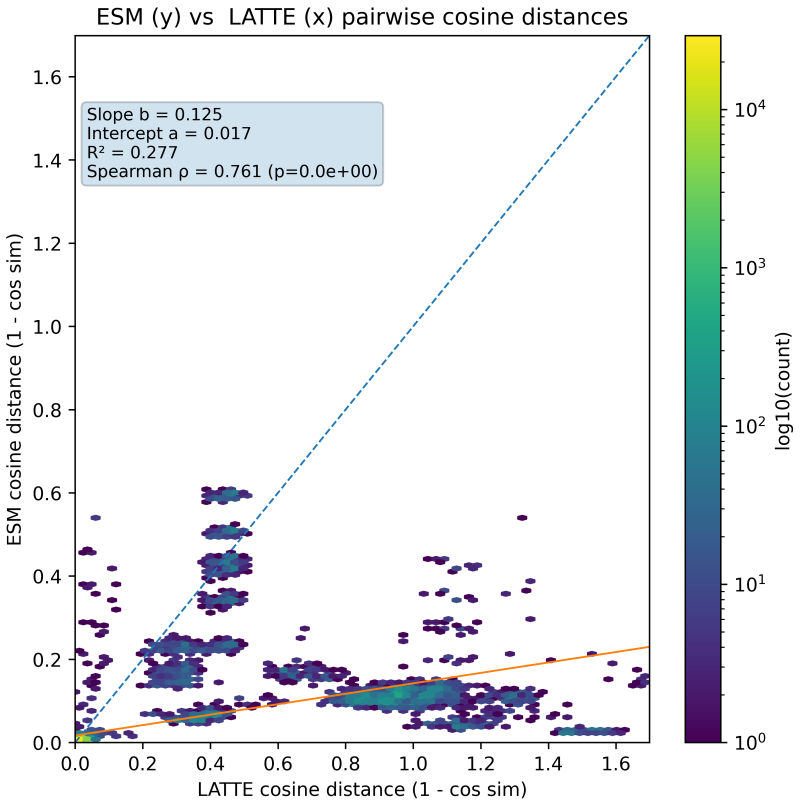


Fig. 6: ESM (y) vs. LATTE (x) pairwise cosine distances for matched pairs. The dashed line marks $y{=}x$; the solid line is an OLS fit ($b{=}0.125$, $a{=}0.017$; $R^2{=}0.277$). Spearman $\rho{=}0.761$ indicates strong rank concordance despite scale compression in ESM.

Table 10: Per cluster sizes and reported means (μ).

Cluster	size	LATTE μ	ESM-2 μ
0	6	0.000	0.728
1	28	0.904	0.737
2	320	0.949	0.920

3.4 LLAST Results

To assess the functionality of the latent tree prefilter placed before conventional BLAST, we quantified both its coverage and efficiency. Coverage was measured as the fraction of baseline BLAST top-50 unique hits that were recovered after prefiltering

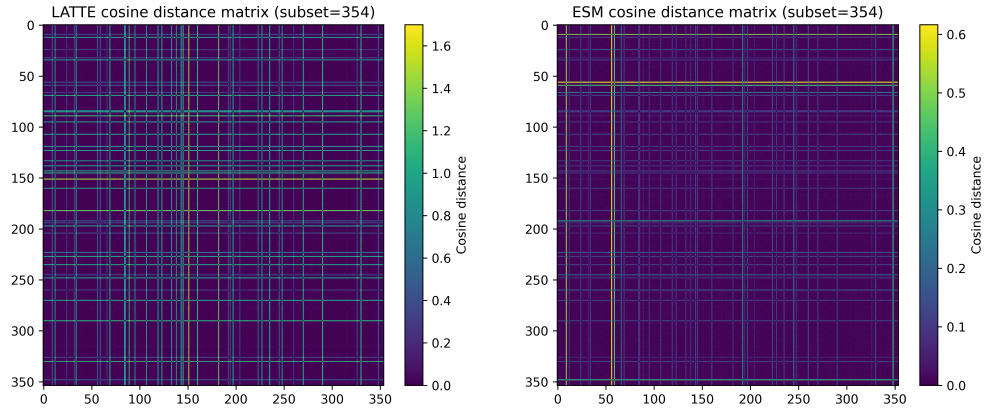
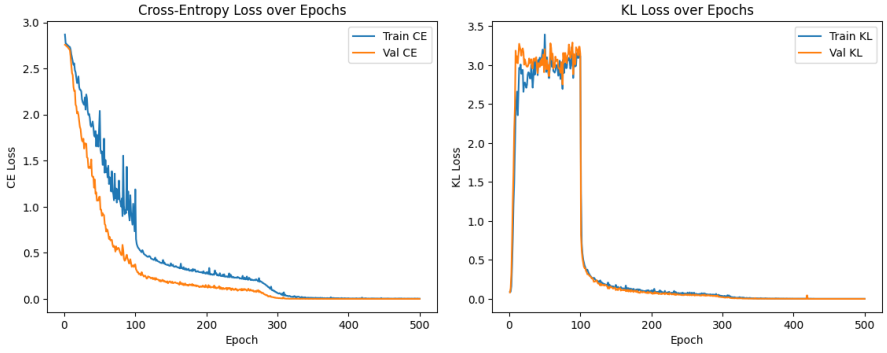


Fig. 7: Pairwise cosine distance matrices for the 354 sequence subset in the two embedding spaces. Left: LATTE; right: ESM.



(a) CE over epochs

(b) KL over epochs

Fig. 8: CE and KL values over epochs for the ablation study model without structural loss.

(SeqRecall@50). Efficiency was evaluated as the number of sequences actually forwarded from the prefilter to BLAST per query, i.e. the effective candidate set size. The underlying database contained approximately $N \approx 10^6$ protein sequences. The latent-tree index was constructed with a target leaf (cluster) capacity of about $S \approx 2000$ sequences, and each query was restricted to visiting a fixed number K of leaves (top-K nodes in the latent-tree search). We further imposed an explicit upper bound $N_{\max} \in \{2000, 4000, 6000, 8000, 10000\}$ on the total number of candidate sequences passed to BLAST per query, yielding a family of operating points that trade off coverage and speed.

Table 11: LLAST operating points on a $\sim 10^6$-sequence database (991 queries). $N_{\max}$ is the upper bound on the number of candidates forwarded to BLAST per query.

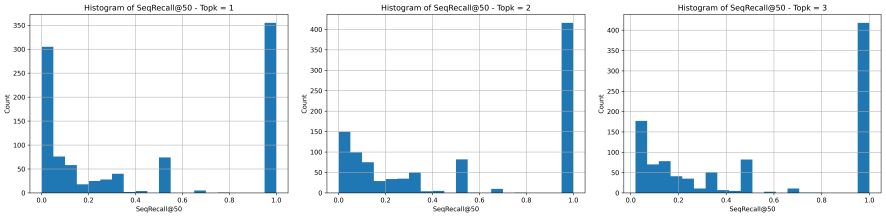
TopK	$N_{\max}$	Mean SeqRecall@50	Mean candidates/query	Approx. reduction
1	2000	0.45	1.8k	$\sim 540\times$
2	4000	0.53	3.8k	$\sim 260\times$
3	6000	0.54	5.6k	$\sim 180\times$
4	8000	0.55	7.7k	$\sim 130\times$
5	10000	0.55	9.7k	$\sim 100\times$

Across 991 protein queries, the most conservative configuration ($N_{\max} = 10{,}000$, corresponding to $\sim 0.97\%$ of the database) achieved a mean SeqRecall@50 of approximately 0.55 while reducing the BLAST search space by roughly $100\times$ relative to scanning all $N \approx 10^6$ sequences. A more aggressive configuration with $N_{\max} = 4000$ (about 0.38% of the database) maintained a mean SeqRecall@50 of ~ 0.53, i.e. only a 3–4% relative decrease in coverage, while shrinking the candidate set by a factor of $\sim 260\times$. The most aggressive setting ($N_{\max} = 2000$) reduced the average candidate count to $\sim 0.18\%$ of the database (over $500\times$ reduction), but at the cost of a pronounced drop in SeqRecall@50 and occasional zero-recall queries on the hardest inputs. These operating points illustrate that the latent prefilter can substantially reduce BLAST workload while allowing practitioners to tune the recall–latency trade-off via a single hyperparameter.

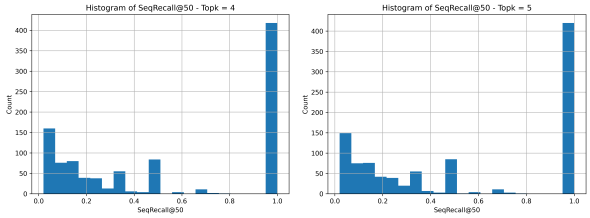
We observed that the most aggressive setting ($N_{\max} = 2000$) produced a small fraction of zero-recall queries (approximately 15% of the 991 queries). These hard-tail cases typically had very few unique baseline BLAST hits (mean Top50_uniqueSeqs $\approx$ 10) and correspond to sparsely connected or nearly orphan sequences in the database. Relaxing the budget to $N_{\max} \geq 4000$ eliminated almost all zero-recall cases while still maintaining a $\sim 260\times$ reduction in the candidate set.

From a computational standpoint, the prefilter changes the asymptotic dependence of per-query cost on the database size. Let N denote the total number of sequences in the database, S the (approximate) maximum number of sequences stored in each leaf cluster, and K the number of leaves visited per query. In the hierarchical index, the total number of leaves scales as $C \approx N/S$, and the tree depth scales as $\log C$. For a single query, the prefilter first descends the tree (cost $O(\log C)$), selects K leaves, and then runs BLAST only on the sequences within these leaves. The resulting candidate set size is bounded by KS, so the overall per-query cost of the prefilter+BLAST pipeline can be written as

$$T_{\text{prefilter}}(N) \;=\; O\!\left(T_{\text{encode}} + \log \frac{N}{S} + KS\right), \tag{4}$$



(a) TopK=1 (b) TopK=2 (c) TopK=3



(d) TopK=4 (e) TopK=5

Fig. 9: Sequence recall histograms of the prefiltering step for different TopK values (1–5). Each panel shows the distribution of per-query recall, i.e., the fraction of the baseline BLAST top hits that are preserved after latent-space prefiltering.

where T_{encode} is the (database-independent) cost of encoding a single query into the latent space. In contrast, conventional BLAST without prefiltering scales approximately linearly with the database size,

$$T_{\text{BLAST}}(N) \;=\; O(N). \tag{5}$$

When S and K are treated as fixed hyperparameters (i.e. the leaf capacity and the search budget are held constant), experimentation revealed that the dominant term in $T_{\text{prefilter}}(N)$ becomes KS, and the residual dependence on N is only logarithmic via $\log(N/S)$. In practice, this makes the per-query BLAST workload effectively bounded by a constant over database sizes up to at least 10^7–10^8 sequences, whereas the cost of conventional BLAST continues to grow proportionally with N. This bounded candidate set is precisely what allows LLAST to maintain practical query times as the database scales to tens or hundreds of millions of sequences.

4 Discussion and future directions

In summary, we propose a compact, structure-aware latent model (LATTE) and show that the resulting 256-dimensional space simultaneously supports reconstruction, property prediction, and a scalable latent-tree prefilter (LLAST) for BLAST.

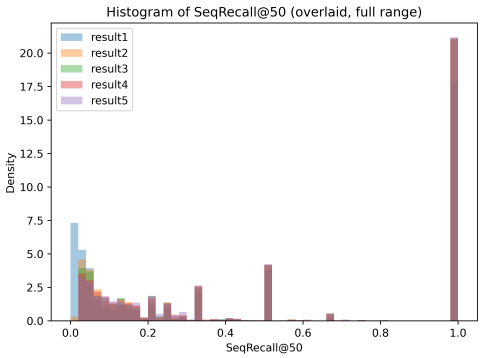


Fig. 10: Overlapped histogram

4.1 LATTE

Directly enforcing structural consistency with AlphaFold-based losses or contact maps is impractical at the scale of millions of sequences: end-to-end structure prediction requires minutes per sequence, contact maps demand substantial memory and curated structures, and both approaches introduce alignment and coverage complications for unannotated regions. In this work we instead leverage lighter protein language models (PLMs) as structural teachers, using ESMS embeddings derived from ESM-2 to shape LATTE's latent space.

The empirical results suggest that this strategy is sufficient to impose a useful geometry on a compact latent space. Compared with ESM-2, LATTE yields broader, heavier-tailed cosine distance distributions while preserving neighbour ordering and clustering structure, as reflected by the high Spearman rank correlation and the strong agreement of $k = 3$ partitions between the two spaces. Together with the ablation study, where removing the structural loss led to rapid KL collapse and degenerate latents, this supports the view that ESMS-based perceptual losses actively prevent posterior collapse and encourage informative latent variables rather than merely matching reconstruction statistics.

The GFP case study reinforces this interpretation. A 256-dimensional latent vector with an active KL divergence near 0.05 is sufficient for Gaussian process models to separate fluorescent from non-fluorescent proteins and to predict excitation and emission peaks with errors comparable to an ESM-2 embedding baseline, despite LATTE having a far smaller parameter budget and lower inference cost. This indicates that structural supervision via PLMs can produce low-dimensional, structure-aligned representations that are competitive for downstream functional tasks without relying on direct sequence generation.

Looking forward, the same framework could be extended to other structural teachers (e.g., ESM-3 or medium-sized PLMs) and additional supervision signals such as contact probabilities or disorder predictions. These directions would test whether

LATTE-style encoders can be tuned toward specific functional families while retaining the generic geometric properties that make the latent space suitable for retrieval and design.

4.2 Interpreting discrepancies between LLAST and baseline BLAST

Although LLAST is explicitly designed to act as a prefilter for BLAST, the per-query top-50 hit sets recovered by LLAST and by baseline BLAST are not identical. We see this not as a failure of LLAST to replicate BLAST, but as a consequence of several structural and algorithmic differences between the two procedures.

First, LLAST operates under an explicit budget constraint. For each query, the latent-tree search is restricted to a fixed number of leaves (TopK) and forwards at most $N_{\max}$ candidate sequences to BLAST. Any relevant sequences that reside in unvisited leaves are, by design, excluded from the candidate pool regardless of how well they would score under BLAST. In contrast, baseline BLAST conceptually searches the entire database and is not subject to a comparable cluster-level visitation constraint.

Second, LLAST ranks candidates in a structure-aware latent space, whereas BLAST ranks them by local alignment scores. LATTE is trained to place sequences with similar global fold and residue–residue similarity patterns close together, and LLAST searches this space using cosine distance between 256-dimensional latent vectors. BLAST, in contrast, is driven by local high-scoring segment pairs and substitution matrices, and can assign high scores to sequences that share short motifs, low-complexity segments or partial domain matches even when their overall fold or global context differs. As a result, LLAST tends to prioritize candidates that are consistent with the latent geometry learned from structure-supervised training, while BLAST may emphasize different aspects of similarity.

Third, the evaluation metric we use is intentionally conservative. Our SeqRecall@50 metric treats the baseline BLAST top-50 set as a flat reference and counts a miss for every baseline hit that does not appear among the LLAST+BLAST hits, including near-duplicate alignments to the same subject sequence and marginal hits with weak scores or low coverage. In many queries, the BLAST top-50 includes far fewer unique targets than 50, yet SeqRecall@50 penalizes LLAST equally for missing duplicate hits and for missing genuinely distinct homologues. This means that SeqRecall@50 should be interpreted as a lower bound on the recall of meaningful targets under an aggressive candidate budget, rather than as an upper bound on what LLAST could achieve with a more permissive search.

Fourth, LLAST performs an approximate nearest-neighbour search through a discretised tree index. The latent-tree traversal relies on centroid distances and hierarchical pruning decisions, which introduce quantisation effects: sequences that lie near cluster boundaries can fall just outside the visited region even if they are slightly closer in latent space than some of the retrieved candidates. Such approximation is inherent to scalable tree-based indices and trades a controlled loss in recall for substantial reductions in the number of sequences forwarded to BLAST.

Together, these factors explain why LLAST does not reproduce the baseline BLAST top-50 sets exactly. LLAST aims to provide a compact, structure-aware candidate pool under a fixed budget, not to clone BLAST's full-database behaviour. In this sense, the observed discrepancies reflect both the intentional restrictions of the latent index (limited cluster visitation and approximate search) and the different notions of similarity captured by a structure-supervised latent model and by local alignment statistics.

In practice, these discrepancies are acceptable in view of the operating points we observe: configurations with $N_{\max} = 4{,}000$–$10{,}000$ retain roughly 0.53–0.55 SeqRecall@50 while reducing the average BLAST workload by $\times 100 \sim 260$ on a 10^6 sequence database, making exact replication of baseline BLAST hits a poor trade-off compared with the achievable efficiency gains.

4.3 Relation to MMseqs2 and sequence-based prefilters

A natural point of comparison for LLAST is MMseqs2, which is widely used as a highly optimized prefilter and as a stand-alone alternative to BLAST in large-scale sequence search. MMseqs2 employs a cascade of k-mer based filters and ungapped extensions to rapidly prune the search space prior to gapped alignment, and its implementation has been carefully optimized with SIMD vectorization and parallel execution to achieve near-linear scaling on large databases. LLAST is not intended as a drop-in replacement for MMseqs2, nor as a new state-of-the-art sequence-only filter.

Conceptually, LLAST differs from MMseqs2 in two principal respects. First, LLAST operates in a compact, structure-aware latent space learned by LATTE, and its tree traversal is driven by cosine distances between 256-dimensional latent vectors rather than by k-mer matches on raw sequences. The same latent representation is reused for reconstruction, downstream property prediction and retrieval, whereas MMseqs2 is specialized for fast sequence comparison and does not expose a shared latent geometry that can be directly integrated into downstream models. Second, LLAST enforces an explicit, user-controllable candidate budget (via $N_{\max}$ and TopK) and delegates the final scoring to BLAST, effectively turning BLAST into a budgeted re-ranking layer on top of a learned index. By contrast, MMseqs2 integrates its own filtering and alignment stages and is typically employed either as a complete search pipeline or as a drop-in replacement for BLAST in high-throughput workflows.

From this perspective, LLAST should be regarded as complementary to tools such as MMseqs2 rather than as a direct competitor. In scenarios where the primary objective is to scan very large databases with maximal throughput, sequence-based engines such as MMseqs2 remain the method of choice. The present work instead demonstrates that a single structure-aligned latent space can support both functional modeling and a latent-tree prefilter that substantially reduces the BLAST workload under an explicit candidate budget. A systematic, wall-clock comparison with MMseqs2 on shared hardware and databases would be valuable, but lies beyond the scope of this study.

5 Conclusion

We introduced LATTE, a lightweight variational encoder that uses ESMS-based perceptual losses to align a 256-dimensional latent space with protein structural constraints while maintaining an active KL divergence. On fluorescent proteins, Gaussian process models trained on LATTE embeddings separate FPs from non-FPs with high accuracy and predict excitation and emission peaks with RMSEs comparable to an ESM-2 embedding baseline, despite LATTE having orders-of-magnitude fewer parameters and substantially lower inference cost. Together with the broader, heavier-tailed cosine distance distributions, strong rank concordance with ESM-2, and the ablation study on structural losses, these results indicate that LATTE's low- dimensional latents are informative, structure-aligned representations suitable for downstream functional modeling.

Building on this representation, we constructed LLAST, a latent-tree index that acts as a prefilter for BLAST. On a $\sim 10^6$-sequence database, LLAST restricts the candidate set forwarded to BLAST to at most $N_{\max} = 10{,}000$ sequences per query (about 0.97% of the database) while recovering roughly 55% of baseline BLAST top-50 unique hits. More aggressive configurations with $N_{\max} = 4{,}000$ retain mean SeqRecall@50 around 0.53 yet reduce the average BLAST workload by approximately 260×, illustrating a tunable trade-off between recall and candidate set size. Under fixed leaf capacity and search budget, the combined prefilter+BLAST pipeline exhibits only logarithmic dependence on the database size, making per-query costs effectively bounded over practical scales.

Taken together, LATTE and LLAST demonstrate that a compact, structure-aware latent space can simultaneously support accurate reconstruction, property prediction, and scalable sequence search. Rather than viewing generative encoders and retrieval indices as separate systems, our results suggest treating them as different uses of the same latent geometry. Extending this approach to other functional families, larger pretraining corpora, and additional structural or functional supervision offers a promising path toward task-aware protein indices that remain efficient at the scale of tens or hundreds of millions of sequences.

References

[1] Anfinsen CB. Principles that Govern the Folding of Protein Chains. Science. 1973;181(4096):223–230. https://doi.org/10.1126/science.181.4096.223.

[2] Brändén CI, Tooze J. Introduction to Protein Structure. 2nd ed. New York: Garland Science; 1999.

[3] Dill KA, MacCallum JL. The Protein-Folding Problem, 50 Years On. Science. 2012;338(6110):1042–1046. https://doi.org/10.1126/science.1219021.

[4] Kingma DP, Welling M. Auto-Encoding Variational Bayes. arXiv preprint arXiv:13126114. 2014;.

[5] Rives A, Meier J, Sercu T, et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. Proceedings of the National Academy of Sciences. 2021;118(15):e2016239118. https://doi.org/10.1073/pnas.2016239118.

[6] Lin Z, Akin H, Rao R, et al. ESMFold: End-to-end single-sequence protein structure prediction. bioRxiv. 2022;https://doi.org/10.1101/2022.07.20.500902.

[7] Jumper J, Evans R, Pritzel A, et al. Highly accurate protein structure prediction with AlphaFold. Nature. 2021;596(7873):583–589. https://doi.org/10.1038/s41586-021-03819-2.

[8] Hayes T, Rao R, Akin H, Sofroniew NJ, Oktay D, Lin Z, et al. Simulating 500 million years of evolution with a language model. Science. 2025;387(6736):850–858. https://doi.org/10.1126/science.ads0018.

[9] Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. Journal of Molecular Biology. 1990;215(3):403–410. https://doi.org/10.1016/S0022-2836(05)80360-2.

[10] Karlin S, Altschul SF. Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. Proceedings of the National Academy of Sciences. 1990;87(6):2264–2268. https://doi.org/10.1073/pnas.87.6.2264.

[11] Camacho C, Coulouris G, Avagyan V, Ma N, Papadopoulos J, Bealer K, et al. BLAST+: Architecture and applications. BMC Bioinformatics. 2009;10:421. https://doi.org/10.1186/1471-2105-10-421.

[12] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention Is All You Need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS); 2017. p. 5998–6008. ArXiv:1706.03762.

[13] Johnson J, Alahi A, Fei-Fei L. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. In: European Conference on Computer Vision (ECCV). vol. 9906 of Lecture Notes in Computer Science. Springer; 2016. p. 694–711.

[14] Kingma DP, Ba J. Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:14126980. 2014;ArXiv:1412.6980 [cs.LG]. Accessed 3 Oct 2025. arXiv:1412.6980. [cs.LG].

[15] Indyk P, Motwani R. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In: Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC); 1998. p. 604–613.

[16] Jégou H, Douze M, Schmid C. Product Quantization for Nearest Neighbor Search. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2011;33(1):117–128. https://doi.org/10.1109/TPAMI.2010.57.

[17] Murtagh F, Legendre P. Ward's Hierarchical Agglomerative Clustering Method: Which Algorithms Implement Ward's Criterion? Journal of Classification. 2014;31(3):274–295. https://doi.org/10.1007/s00357-014-9161-z.

[18] Lambert TJ. FPbase: a community-editable fluorescent protein database. Nature Methods. 2019;16(3):239–244. https://doi.org/10.1038/s41592-019-0352-8.

[19] Rasmussen CE, Williams CKI. Gaussian Processes for Machine Learning. MIT Press; 2006.

[20] van der Maaten L, Hinton G. Visualizing data using t-SNE. Journal of Machine Learning Research. 2008;9:2579–2605.

[21] Lloyd SP. Least Squares Quantization in PCM. IEEE Transactions on Information Theory. 1982;28(2):129–137. https://doi.org/10.1109/TIT.1982.1056489.

[22] Vieira LC, Handojo ML, Wilke CO. Medium-sized protein language models perform well at transfer learning on realistic datasets. Scientific Reports. 2025;15(1):21400. https://doi.org/10.1038/s41598-025-05674-x.

[23] FPbase contributors.: FPbase: A community-editable fluorescent protein database. FPbase. Accessed 3 Oct 2025. https://www.fpbase.org/.

[24] Ahn D.: Fluorescent Protein Dataset (Kaggle). Kaggle. Accessed 3 Oct 2025. https://www.kaggle.com/datasets/dannyahn/fluorescent-protein-dataset.

[25] Ahn D.: LATTE: Structure-Informed Latent Model for Protein Sequence Embedding — GitHub Repository. GitHub. Accessed 28 Oct 2025. https://github.com/Ahnd6474/LATTE.

Abbreviations

CE: cross entropy; KL: Kullback-Leibler divergence; GP: Gaussian process; RMSE: root mean square error; AUC: area under the ROC curve; FP: fluorescent protein.

Declarations

Ethics approval and consent to participate

This study involved only computational analyses on publicly available protein sequence datasets and did not involve human participants, animals, or identifiable personal data. Ethics approval and consent to participate were therefore not required.

Consent for publication

Not applicable.

Availability of data and materials

All data, code, and trained models supporting the conclusions of this study are publicly available at the repositories listed below. Direct hyperlinks are provided; no registration is required.

- **Fluorescent protein dataset (primary source)**: FPbase https://www.fpbase.org/ [23].
- **Dataset mirror (exact copy for reproducibility)**: Kaggle dannyahn/fluorescent-protein-dataset [24].
- **Source code and trained models**: GitHub https://github.com/Ahnd6474/LATTE [25].

Where applicable, version tags/commit hashes used in this manuscript are recorded in the repository release notes and in the supplementary materials. Licensing and any third party terms are as stated on each repository page.

Competing interests

The authors declare that they have no competing interests.

Funding

This work was supported by the Daegu Science High School student independent research program. No specific grant number was assigned. The funder had no role in study design, data collection/analysis, decision to publish, or manuscript preparation