FunSiNN: Predicting Functional Similarity of Protein Pairs

by

Frederic Guintu

Yana Bromberg
Adviser

Computer Science

2025

FunSiNN: Predicting Functional Similarity of Protein Pairs

By

Frederic Guintu

Yana Bromberg

Adviser

Abstract

FunSiNN: Predicting Functional Similarity of Protein Pairs
By Frederic Guintu

The rapid growth of unannotated protein sequence data presents a major challenge for functional annotation. This thesis proposes a scalable approach for predicting functional similarity between protein pairs using ProtT5 language model embeddings and a Siamese Neural Network (SNN). We first generate a similarity network by thresholding cosine similarity scores between embeddings, followed by subsampled Louvain clustering to produce functionally similar groups. A refinement step further improves cluster granularity. From these clusters, we sample labeled protein pairs to train the SNN, which learns to classify whether two proteins share function.

Experimental results show that refined clusters yield stronger labels and improved predictive performance (F1 = 0.7670, AUC = 0.8472). Our findings demonstrate the potential of combining pLM embeddings, unsupervised clustering, and deep learning to enable large-scale protein function prediction in the absence of curated annotations.

FunSiNN: Predicting Functional Similarity of Protein Pairs

By

Frederic Guintu

Yana Bromberg
Adviser

A thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Science with Honors

Computer Science

2025

Acknowledgements

I wish to convey my appreciation to my advisor, Dr. Yana Bromberg, and Prabakaran Ramakrishnan for their invaluable guidance and support with my project. Their expertise enhanced my grasp of research methodologies, improved my analytical skills for interpreting data, and helped me effectively present my findings.

Furthermore, my gratitude extends to the Emory Computer Science Honors program for facilitating a smooth and successful process for my undergraduate honors thesis. I am particularly grateful to Dr. Davide Fossati, the honors program coordinator, for his support in the application process and advisor selection.

Finally, I am thankful to the Bromberg Lab for providing the computing resources necessary for my work. Special thanks to Michael Heinzinger, Tobias Senoner, and Sergio Gramacho for enabling my access to the ProtT5 embeddings.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Large language models (LLMs) are computational models designed to understand the mechanisms behind structured data. Initially developed for natural language processing, training these models on large corpuses allows LLMs to identify patterns and operate on the information learned from training applied towards a variety of tasks. Traditionally trained on text corpora from books, articles, and online discussions, the techniques learned from the field of Natural Language Processing can also be applied to non-linguistic applications by adapting the training step and training data to the specific need. By extending their capabilities beyond human language, LLMs offer new possibilities for interpreting and learning from structured sequences across a wide range of disciplines, opening avenues for innovation in fields where traditional computational methods fall short.

Protein sequences are an example of structured data that can be used in LLM training and its applications. Built off of 20 basic amino acids notated by single uppercase letters [1], protein sequences are chains of amino acids linked together in linear order that represent the protein's primary structure. The order in which amino acids are sequenced is primarily determined by the genetic code, which translates nucleotide sequences into proteins.[8] However, evolutionary pressures, such as mutations and

[1]Standard Amino Acid vocabulary: ACDEFGHIKLMNPQRSTVWY

natural selection, shape and refine protein sequences over time. [13] Since a protein's function is dictated by its three-dimensional structure, understanding sequence patterns and how they translate into structural motifs can provide key insights into protein function. [1] Structural similarities often imply functional similarities, enabling computational models to predict unknown protein functions based on known structures.

Due to developments in large-scale sequencing of proteins, the number of unannotated protein sequences presents a significant challenge in understanding protein structure, function, and interactions. [6] Since traditional experimental annotation is time-consuming and costly, LLMs offer an alternative solution to learning structural patterns from vast amounts of unlabeled protein sequence data since LLM training only requires the sequences as input.[9]

This thesis aims to leverage the power of LLMs by extracting the information learned from a pre-trained Protein Language Model (pLM) as a feature representation in a transfer learning framework. PLMs provide a powerful tool for capturing underlying structural and functional patterns from sequence data by their embeddings. To utilize this, we employ clustering algorithms on the extracted protein embeddings to group similar proteins based on learned representations. These clusters are then used to generate labeled pairs of protein sequences, where proteins in the same cluster are considered functionally similar. Using these labeled pairs, a Siamese Neural Network (SNN) is then trained to predict whether two proteins share the same function, addressing the challenge of functional annotation in large-scale proteomics.

Chapter 2

Background

This literature review examines the role of pLMs in protein function prediction, additionally placing significance on the proposed method of embedding-based methods for similarity classification.

2.1 Protein Function Prediction Methods

Computational approaches in proteomics have seen multiple developments to keep pace with the exponentially growing discovery of natural proteins and their sequences. Data-driven methods provide valuable insight into unannotated protein sequences, offering an alternative to manually verified experimental annotations.[3] These approaches typically fall into two main categories depending on the data type of analysis: sequence-based and structure-based methods.

Early computational analysis of sequences began with homology-based function transfer, where a query protein sequence is compared against a database of known proteins using sequence alignment techniques. The reviewed annotations of biological function from the top hits of this search are then used to infer the query protein's function, with the idea that evolutionarily related proteins share similar functions.[16] As told by Lee et al.: "Inheritance through homology" is the principle that drove the process of early protein function prediction strategies.[12] However, this method faces limitations in cases where the queried protein has few homologs. Additionally, homology-based function transfer is ineffective in detecting similarities from other proteins that have low sequence similarity, which induces low alignment scores despite potentially sharing structural or functional features. This gap in traditional approaches has motivated the development of more advanced data-driven methods, including the use of machine learning.

Traditional machine learning (ML) strategies for protein function prediction not only relied on sequence-based features but also incorporated structural and functional information when available. Traditional ML algorithms such as Support Vector Machines, Random Forests, and Neural Networks have been employed to train on structural protein features alongside sequence representations.[3] These features were extracted from experimentally resolved protein structures and often improved performance and function prediction accuracy over purely sequence-based models.[10] However, the effectiveness of structure-based ML approaches is limited by the availability of experimentally determined protein structures, as only a small fraction of known protein sequences have corresponding 3D structures.[3] For instance, the DeepFRI model, which utilized a Graph Convolutional Network trained on structure-based features (contact maps from the Protein Data Bank) struggled in predicting functionality for proteins with novel structural folds not present in the training set.[10] These limitations led to further exploration towards sequence-only representations of protein function, as protein sequence data is far more abundant than experimentally resolved structures.[3] This shift laid the groundwork for the development of pLMs, which learn functionally relevant patterns directly from large-scale sequence datasets without relying on structural information.

ProtT5, developed by Elnaggar et al. as part of the ProtTrans framework, is a pLM that leverages self-supervised learning techniques to extract meaningful features of protein sequences using the T5 "Text-to-Text Transfer Transformer" architecture.[9] The ProtT5-XL-UniRef50 is based on the t5-3b model and was pretrained on UniRef50, with the task of masked language modeling, where randomly masked residues are predicted based on contextual information learned from known sequences. According to Elnaggar et al., "Embeddings extract constraints about protein function and structure learned by the protein LMs during self-supervised pre-training on raw (unlabeled) protein sequences," emphasizing the effectiveness in sequence-only representations of protein function and its robust applications toward downstream tasks. Furthermore, protein embeddings have been shown to outperform traditional sequence alignment methods, particularly in cases where proteins share low sequence identity, as demonstrated by their superior performance in alignment tasks of ProtT5's embeddings compared to conventional homology-based approaches.[11]

2.2 Protein Database

The UniProt Knowledgebase (UniProtKB) is a freely accessible and high-quality database of protein sequences. Protein sequences are contributed to the knowledge space by multiple collaborations of the research community, including the European Bioinformatics Institute (EBI), the Swiss Institute of Bioinformatics (SIB), and the Protein Information Resource (PIR), with the goal of providing access to proteins and their functional information. [6]

Section	**Number of entries in total**	**Number of entries with an annotation update**	**Number of entries with a sequence update**
UniProtKB	**253,206,171**	192,548,351	202,121
Reviewed (Swiss-Prot)	572,970	454,450	33
Unreviewed (TrEMBL)	252,633,201	192,093,901	202,088

Table 2.1: Total number of entries in the 2025_01 release of UniProtKB

The vast majority of protein sequences in UniProtKB remain unreviewed (TrEMBL), with over 252 million entries compared to only ∼573,000 reviewed (Swiss-Prot) entries. This highlights the immense gap between experimentally verified proteins and those predicted but not yet manually curated. Thus, PLMs such as ProtT5 provide a powerful alternative to manual curation by learning functional and structural constraints directly from the widely available sequence entries. Especially for the case of unreviewed sequences, pLMs help to bridge the annotation gap in large-scale proteomic datasets.

2.3 Transfer Learning

An innovative machine learning approach that provides distinctive insights into protein function prediction is the Siamese Neural Network (SNN) design. SNNs are employed for comparative tasks such as facial recognition or signature verification, where identical neural networks evaluate two inputs to yield comparable outcomes [4][14]. When tailored to protein representations, SNNs can identify relationships and common motifs between protein pairs. The key objective is to establish a reliable similarity metric that accurately reflects functional similarities between protein pairs, enabling the SNN model to effectively learn the relationship between sequence context and functional similarity.

Long short-term memory networks (LSTMs), particularly BiLSTMs, are well-suited for modeling protein sequences in SNNs by capturing long-range dependencies in both directions. In an SNN, BiLSTM encoders generate sequence embeddings that capture contextual relationships between tokens, which can then be utilized to learn similarity relationships. Neculoiu et al. demonstrated the effectiveness of Siamese LSTMs in learning similarity metrics, supporting their application in protein function prediction [14].

Unlike previously discussed ML approaches to the task of protein function prediction, SNNs do not require exhaustive feature extraction to learn similarities of protein pairs. SNNs can be supervised using similarity labels (such as assigning label 1 for similar proteins A and B, and 0 for those that are not similar), enabling the model to learn how to assess protein similarity based on the input sequences. For instance, SENSE (SiamEse Neural network for Sequence Embedding) developed by Zheng et al. defined the similarity metric as the mean square error between their SNN's output embedding and a known alignment-distance score, which led to state-of-the-art performance in alignment-free sequence comparison methods. [17] A well-defined similarity metric can allow SNNs to learn pairwise protein functional similarity if the metric effectively translates this relationship. Additionally, by focusing on pairwise similarity scoring instead of assigning a protein to a fixed label, SNN approaches avoid being limited by the biases and gaps in annotation databases.

In summary, this project aims to leverage the strengths of pLMs and SNNs to address the question: "Given a pair of proteins, do they share functional similarity?" By capturing the "language" of proteins, pLMs encode structural and functional information into their embeddings, enabling rich representations of protein function and structure. Provided with an embedding-based similarity label, an SNN can then learn the relationships between proteins allowing it to determine whether two input sequences are functionally similar or dissimilar based on their learned representations.

Chapter 3

Approach

3.1 Protein Sequence Dataset

The data for this study was taken from UniRef90 of the UniProt Reference Clusters (UniRef) database. UniRef90 is a collection from UniProt that clusters protein sequences, combining them into non-redundant sets of proteins at 90% sequence identity, using UniRef100 as a starting point and focusing on sequences with 80% overlap with the longest sequence in the cluster. [6]

Due to UniRef90 being a large collection of proteins,[1] it remains computationally intensive for large-scale sequence analysis. To further reduce the dataset size to a manageable level given computational constraints, the following filter was applied to restrict the UniProt protein search:

```
(count:[10 TO *]) AND (length:[* TO 1200])
```

This query selected UniRef90 clusters with at least 10 members and limited representative sequence lengths to a maximum of 1200 residues, resulting in 5,190,657 protein entries in the dataset used for this project.[2]

[1]UniRef90 clustering reduced the UniProt database size by 58%, resulting in 204,806,910 entries as of March 2025.[6]

[2]The dataset used in this project was downloaded in October 2024, yielding 5,190,657 entries.

3.2 ProtT5 Embeddings

With the selected UniRef90 sequences, per-protein embeddings were extracted from ProtT5-XL-UniRef50[9] for use in the downstream functional similarity task. Per-protein embeddings were chosen over per-residue embeddings because the goal is to classify entire proteins in relation to one another, rather than analyzing individual residues. Since ProtT5 is a language model, this approach is analogous to treating protein sequences as full sentences in a sentence similarity task in natural language processing (NLP). Each per-protein embedding is a float array of shape (1024, 1).

3.3 Similarity Metric

3.3.1 Cosine Similarity Threshold Calculation

To define functional similarity between pairs of proteins, cosine similarity of protein embeddings is used as the metric to measure similarity, computed by the following equation which relates two vectors by the cosine of the angle between the vectors.[15]

$$\text{cosine\_similarity}(\mathbf{A}, \mathbf{B}) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} \tag{3.1}$$

Given the large scale of the protein embedding dataset (5190657, 1024), computing a similarity label for every possible pair of proteins would be computationally infeasible, as it would require $O(N^2)$ complexity in both time and space. Thus, an efficient method to parse through the embedding database is needed.

To tackle this, a cosine similarity threshold was determined to restrict the search space for protein pairs, ensuring that only highly likely functionally similar proteins were considered. Proteins with similarity scores below the threshold were deemed

Running the same query on UniProt as of March 2025 returns 5,923,877 entries due to database updates.

unlikely to share functional similarity. This threshold was established through preliminary testing on the dataset, where a random subset of real protein embeddings from the selected UniRef90 proteins were sampled and their cosine similarity scores were computed against the following:

Category	**Description**
Real-real pairs	Pairs of embeddings from actual proteins in the subsampled dataset.
Randomly generated sequence embeddings	**Uniform amino acid distribution:** Sequences of randomly generated lengths, where each amino acid was sampled with equal probability (5% per amino acid). **Dataset-based amino acid distribution:** Sequences of fixed length (equal to the mean sequence length of the subsampled dataset), with amino acid composition matching the empirical distribution of the subsampled dataset. See Table A.1 for details. **Shuffled protein sequences:** Sequences from the proteins in the subsampled dataset were shuffled before embedding generation, maintaining sequence lengths and amino acid composition.

Table 3.1: Types of protein pairs used for cosine similarity threshold determination.

By analyzing the distribution of cosine similarity scores between real-real protein pairs and real-random pairs, a threshold was selected such that 95% of pairs with cosine similarity exceeding this threshold were from the real-real pair distribution. This method was chosen to minimize false positives, as it is highly improbable that a real protein would share functional similarities with a randomly generated sequence. Similarly, in NLP, it is exceptionally rare for a meaningfully constructed sentence to convey the same meaning as a randomly generated sentence using word frequencies without consideration for linguistic rules.

3.3.2 Threshold Search

Using the threshold, I employed the Faiss library developed by Meta's Fundamental AI Research group for efficient vector similarity search.[7] Faiss enables fast nearest-neighbor searches on high-dimensional vectors by leveraging optimized indexing and search strategies.

The similarity search pipeline was constructed by initially normalizing the vectors by dividing it by its L2 norm:

$$\hat{\mathbf{A}} = \frac{\mathbf{A}}{\|\mathbf{A}\|}, \quad \hat{\mathbf{B}} = \frac{\mathbf{B}}{\|\mathbf{B}\|}, \quad \text{where} \quad \|\mathbf{A}\| = \sqrt{\sum_i A_i^2}, \quad \|\mathbf{B}\| = \sqrt{\sum_i B_i^2} \tag{3.2}$$

Building a Flat Inner Product (FlatIP) index, which stores all vectors and performs exact nearest-neighbor searches based on their inner product, ensures that the index search will be based on the cosine similarity.

$$\hat{\mathbf{A}} \cdot \hat{\mathbf{B}} = \left(\frac{\mathbf{A}}{\|\mathbf{A}\|}\right) \cdot \left(\frac{\mathbf{B}}{\|\mathbf{B}\|}\right) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \text{cosine\_similarity}(\mathbf{A}, \mathbf{B}) \tag{3.3}$$

The range search function (`index.range_search(X, threshold)`) was used to identify all protein pairs with a cosine similarity exceeding the predefined threshold. To ensure non-redundancy in the dataset, only unique pairs were considered: if a pair (A, B) was already included, its reverse counterpart (B, A) was excluded. Additionally, self-pairs (A, A) were removed to avoid trivial matches.

The resulting pairs make up the edge list of a protein network, where each pair connects two proteins corresponding to their embedding cosine similarity score. Each protein in the pair represents a node and the similarity score represents an edge in the protein network. The graph representation of protein-protein similarity can be used for clustering to identify functional groups within UniRef90. By structuring the protein embeddings in this way, the dataset allows for the discovery of functional subgroups within the protein similarity network. The complete list of pairs above the threshold amounted to 5,540,906,042 pairs.

3.3.3 Clustering

Because of the complex nature of proteins, relying solely on high cosine similarity of embeddings is inadequate for capturing functional, structural, and sequential similarities found within protein families.[8] Thus, clustering these embeddings offers a more accurate reflection of the similarity observed in real protein structures.

Ideally, clustering the complete network would provide the necessary similarity labels to learn functionally similar groups from. However, due to the limitations in computational resources, an alternative approach is required to represent functional groups.

3.3.4 Subsampled Clustering Method

The following approach was used to obtain a representative clustering of the whole network. Algorithm 1 takes the full weighted graph G, the fraction of nodes to sample in each iteration f, and the number of iterations n. By utilizing a random subsampling process that ensures coverage of every node in the complete graph, this allows multiple clustering iterations to be performed on smaller networks to retrieve the data needed to acquire a representative clustering. The clutering function used, LouvainClustering(G_i, r), was implemented by RAPIDS cuGraph tested at different resolutions.[2]

Algorithm 1: Random Subsampling and Louvain Community Detection

```
Input: Graph $G = (V, E, w)$, number of iterations $n$, subsample fraction $f$,
       resolution parameter $r$
Output: Collection of clusterings $\{C_1, C_2, \ldots, C_n\}$
1 $\{C_1, C_2, \ldots, C_n\} \leftarrow \emptyset$
2 for $i \leftarrow 1$ to $n$ do
3     $V_i \leftarrow$ random subset of $V$ with $|V_i| = f \cdot |V|$
4     $G_i \leftarrow$ subgraph of $G$ induced by $V_i$
5     $C_i \leftarrow$ LouvainClustering($G_i, r$)
6 end
7 return $\{C_1, C_2, \ldots, C_n\}$
```

Algorithm 1 returns $\{C_1, C_2, \ldots, C_n\}$, which is the collection of all clusterings from the n iterations, where each clustering may have different label assignments for the same communities.

Using the collection of subsampled clusterings as the input to Algorithm 2, cluster labels are aligned across all iterations by matching each cluster to its most similar counterpart based on Jaccard similarity in a reference clustering, ensuring that similar communities consistently receive the same label.

Algorithm 2: Cluster Alignment and Consensus

```
**Input:** Collection of clusterings $\{C_1, C_2, \ldots, C_n\}$
**Output:** Consensus clustering $C_{\text{consensus}}$
1  $C_{\text{ref}} \leftarrow C_1$ ;                    // Use first clustering as reference
2  Aligned $\leftarrow \{C_{\text{ref}}\}$
3  for $i \leftarrow 2$ to $n$ do
4      $C_i^{\text{aligned}} \leftarrow \emptyset$
       /* Build mappings:  label → set of nodes                        */
5      RefClusters $\leftarrow$ {label $\rightarrow$ {nodes}} from $C_{\text{ref}}$
6      RunClusters $\leftarrow$ {label $\rightarrow$ {nodes}} from $C_i$
7      for each cluster $(\ell, nodes)$ in RunClusters do
           /* Find best matching RefCluster using Jaccard similarity
              */
8          bestMatch $\leftarrow \arg\max_{\ell_{\text{ref}} \in \text{RefClusters}} \frac{|\text{nodes} \cap \text{RefClusters}[\ell_{\text{ref}}]|}{|\text{nodes} \cup \text{RefClusters}[\ell_{\text{ref}}]|}$
9          if no match found then
10             Create new label bestMatch and add to reference
11         end
12         for each node $v$ in nodes do
13             $C_i^{\text{aligned}}[v] \leftarrow$ bestMatch
14         end
15     end
16     Add $C_i^{\text{aligned}}$ to Aligned
17 end
   /* Create consensus through majority voting                         */
18 AllNodes $\leftarrow$ union of nodes from all clusterings for each node $v$ in AllNodes
    do
19     $C_{\text{consensus}}[v] \leftarrow$ most common label for $v$ across Aligned
20 end
21 return $C_{consensus}$
```

One limitation to using the full consensus clustering ($C_{consensus}$) is the potential for merging distinct communities that are marginally similar in the alignment process. Since the alignment method matches clusters based on Jaccard similarity without a minimum threshold, clusters with weak connections (single or few node overlap) may receive the same label, creating artificially large groupings that lack meaningful functional representations.

To address this issue, algorithm 3 implements a refinement step applied on the full consensus clustering by re-clustering aligned groups and assigning new cluster labels.

```
Algorithm 3: Refine Large Clusters
Input: Consensus clustering $C$, refined resolution parameter $r$
Output: Refined clustering $C'$
1 $C' \leftarrow \{\}$
2 for each cluster $(\ell, nodes)$ in Clusters do
      /* Extract subgraph and re-cluster */
3     $G_\ell \leftarrow$ subgraph of $G$ induced by nodes
       SubClusters $\leftarrow$ LouvainClustering($G_\ell, r'$)
      /* Assign new unique labels to sub-clusters */
4     Assign new unique labels to all nodes in SubClusters Update $C'$ with these
       new labels
5 end
6 return $C'$
```

This clustering approach aims to obtain a representation of protein-protein functional similarity. The resulting clusters define the similarity metric: proteins that share the same cluster label share similar function (labeled 1) and protein pairs that reside in different clusters have distinct function (labeled 0).

Utilizing this method provides the advantage of being able to tackle extremely large protein networks by ensembling multiple iterations of computationally manageable clustering steps. However, because this clustering method relies on subsamplings of the full network, it introduces some degree of stochasticity to the clustering, meaning there is the chance for key functional relationships to be lost or misclassified as it is not a full clustering.

Despite these challenges, this clustering method provides a step towards learning protein similarity, ensuring that proteins are grouped based on their similarity within the ProtT5 embedding space while maintaining computational feasibility. The resulting labeled dataset where protein pairs within the same cluster are assigned a label of 1 (similar function) and pairs in different clusters are labeled 0 (distinct function), serves as the ground truth for training the SNN.

3.4 SNN Architecture

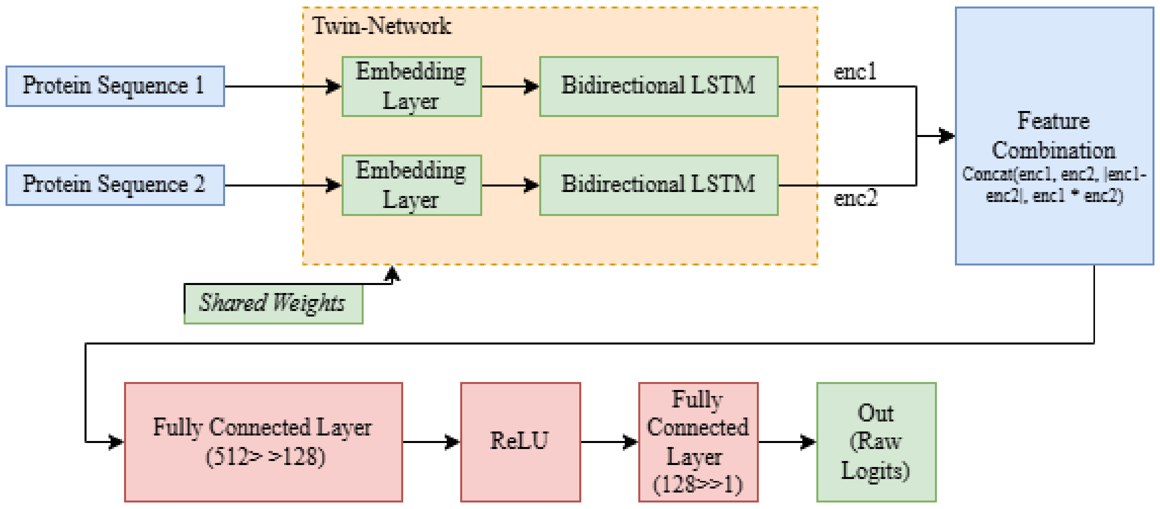

Figure 3.1: Overview of the Siamese BiLSTM Architecture for Protein Functional Similarity Prediction of pairs of protein sequences. The weights in the Twin-Network (orange) are shared.

In this project, a Siamese BiLSTM (Bidirectional Long Short-Term Memory) network is employed to predict whether two proteins share similar functions based on their sequence embeddings. This architecture is specifically designed to learn pairwise similarity relationships, making it well-suited for the functional similarity classification task.

The architecture consists of two identical BiLSTM encoders that capture amino acid dependencies in both directions of the sequence. Each pair of sequences are tokenized, mapped to separate embedding and Bidirectional LSTM layers with shared weights to result in a sequence-level representation. The features from this layer are then combined using their absolute difference and element-wise product and passed through fully connected layers that output raw logits for similarity classification.

The model is trained using pairs of protein sequences and their labels indicating if the proteins are found in the same functional cluster. The use of binary cross-entropy with logits loss (BCEWithLogitsLoss) combines a sigmoid activation and binary cross-entropy in a single numerically stable operation, ensuring that the model optimizes learning labels for functional similarity given the combined encoding features. By leveraging this deep-learning approach, the model learns functional similarities without the use of sequence homology and annotations.

3.4.1 Evaluation Metrics

Model performance is evaluated using a comprehensive set of metrics that account for classification accuracy, ranking ability, and decision boundaries. The primary metrics used include:

- **Accuracy**: The proportion of correctly classified protein pairs, calculated as $\frac{TP+TN}{TP+TN+FP+FN}$, where TP, TN, FP, and FN are true positives, true negatives, false positives, and false negatives, respectively.

- **Precision**: The proportion of correctly predicted positives out of all predicted positives, measuring the model's exactness: $\frac{TP}{TP+FP}$.
- **Recall**: The proportion of correctly predicted positives out of all actual positives, measuring the model's completeness: $\frac{TP}{TP+FN}$.
- **F1 Score**: The harmonic mean of precision and recall, providing a balance between the two: $2 \cdot \frac{precision \cdot recall}{precision+recall}$.
- **ROC-AUC**: Area Under the Receiver Operating Characteristic curve, which plots the true positive rate against the false positive rate at various threshold settings. This metric evaluates the model's ability to rank positive pairs higher than negative pairs, independent of threshold choice.

Additionally, threshold analysis was employed to identify the optimal decision boundary for converting the model's continuous outputs into binary predictions. This analysis examines how various performance metrics change when adjusting the classification threshold from its default value of 0.5. By systematically testing thresholds between 0.1 and 0.9, we identified the optimal threshold that maximizes the F1 score for our protein similarity prediction task. This optimization ensures that the model's predictions align with the specific requirements of protein functional similarity assessment, where both precision and recall are important considerations.

Confusion matrices further supplement these metrics by providing a detailed breakdown of correct and incorrect classifications, allowing for the identification of specific error patterns. Combined, these evaluation approaches provide a robust assessment of the model's capabilities in distinguishing between functionally similar and dissimilar protein pairs.

Chapter 4

Experiments and Analysis

4.1 Working Environment

All experiments were conducted on a Slurm-managed high-performance cluster, which consists of multiple compute nodes dedicated to CPU and GPU-accelerated tasks. The primary nodes for this experiment include:

- **CPU Nodes:** Up to 128 CPU cores and high memory availability up to 975G. These nodes were used for dataset analysis and embedding similarity search.

- **GPU Node:** Equipped with 2 NVIDIA Ada 6000 GPUs and 64 CPU cores. This node was used for deep learning tasks, including the training of SNN and clustering.

4.2 Results and Analysis

4.2.1 Theshold For Cosine Similarity

Figures A.1, A.2, and A.3 present the results of experiments conducted to determine the cutoff for pairwise embedding cosine similarity. As expected, real protein pairs exhibit higher similarity scores compared to real proteins paired with randomly generated sequences. This suggests that within the embedding space of protein sequences generated by ProtT5, the model captures biological information from the proteins. The observed separation between real-real and real-random similarity distributions supports the use of a cosine similarity threshold to distinguish functionally related protein pairs, ensuring that similarity scores above the cutoff predominantly correspond to biologically relevant functional similarities.

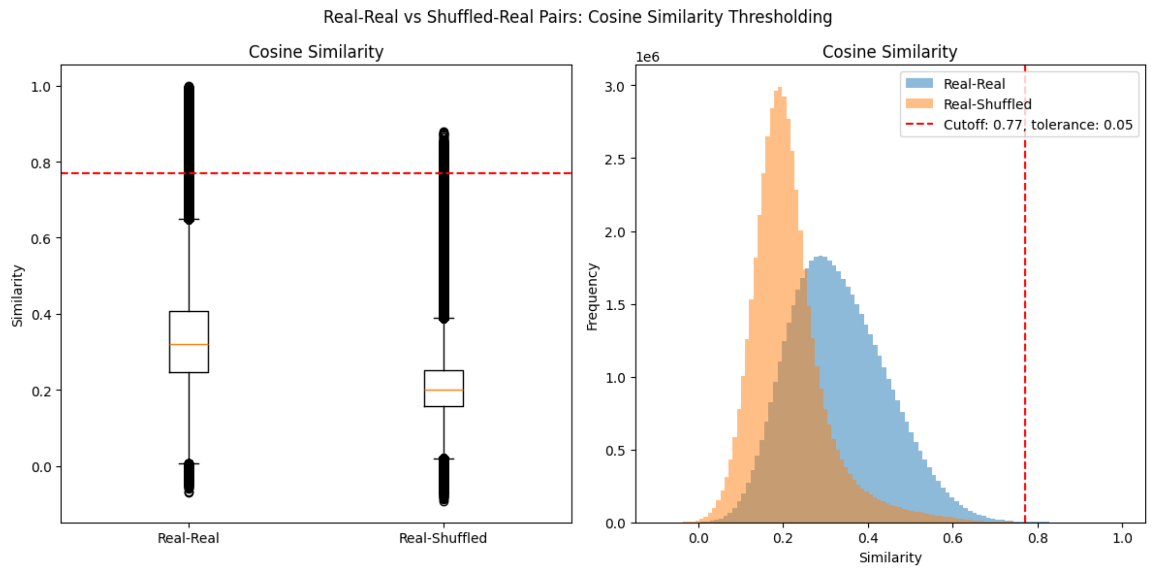

Figure 4.1: Cosine Similarity Thresholding for Functional Similarity. The left panel shows a boxplot of similarity scores for real-real and real-shuffled protein pairs, while the right panel presents their distribution. The red dashed line at 0.77 marks the threshold, effectively isolating high-confidence functional pairs while minimizing false positives from random sequences.

The cosine similarity threshold was selected based on the experiment done in Figure 4.1. This is because shuffled-real pairs, compared to other randomly generated sequences, preserved the length and amino acid composition of real proteins while only disrupting their biological sequence order.

The cutoff threshold of 0.77 (decided with 5% tolerance of random-real pairs exceeding the threshold) isolates high-confidence functional similarities while minimizing the rate of false positives. Although most real-real pairs fall below the threshold, the pairs that exceed this threshold are selected to represent true functional relationships. A lower threshold would increase noise by misclassifying shuffled sequences as similar, while a higher threshold would risk missing valid functional pairs. This balance prioritizes precision over recall, ensuring that clustering and SNN training rely on strong functional associations rather than weak, potentially misleading ones.

4.2.2 Clustering Results

Figures A.4 and A.5 show the results of the experiments done for the subsampled clustering method in Algorithm 1.

The figures illustrate the distribution of cluster sizes across 20 iterations of the subsampled Louvain clustering method. Each subplot represents a separate clustering iteration, showing the frequency of clusters of different sizes. Figure A.4 corresponds to a resolution parameter of 1.0, leading to larger clusters, whereas Figure A.5 uses a resolution of 10.0, which results in more fragmented and smaller clusters. The general trend indicates that most clusters remain small, while a few large clusters persist across iterations.

The results of Algorithm 2 conducted on the two clustering experiments are presented in figures 4.2 and 4.3. Both experiments used 20% subsampling across 20 iterations to achieve a representative clustering of the full dataset while keeping computational costs manageable. The choice of resolution affects the granularity of the final clustering, with lower resolutions leading to larger, more generalized clusters and higher resolutions producing more specific functional groupings.

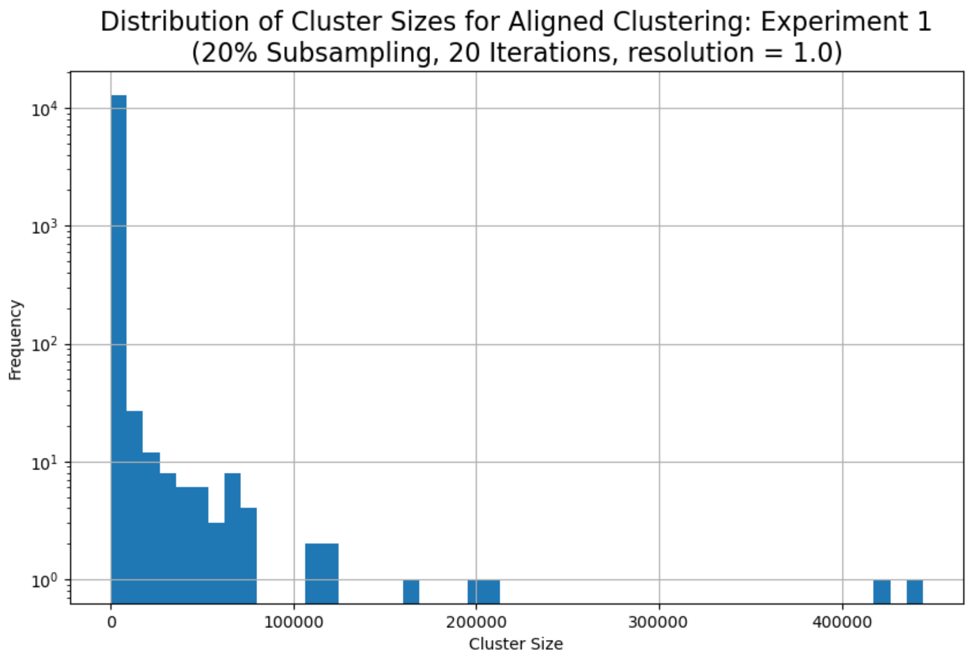

Figure 4.2: **Clustering Experiment 1 Aligned:** Distribution of cluster sizes after running alignment algorithm (Algorithm 2) on **Clustering Experiment 1**.

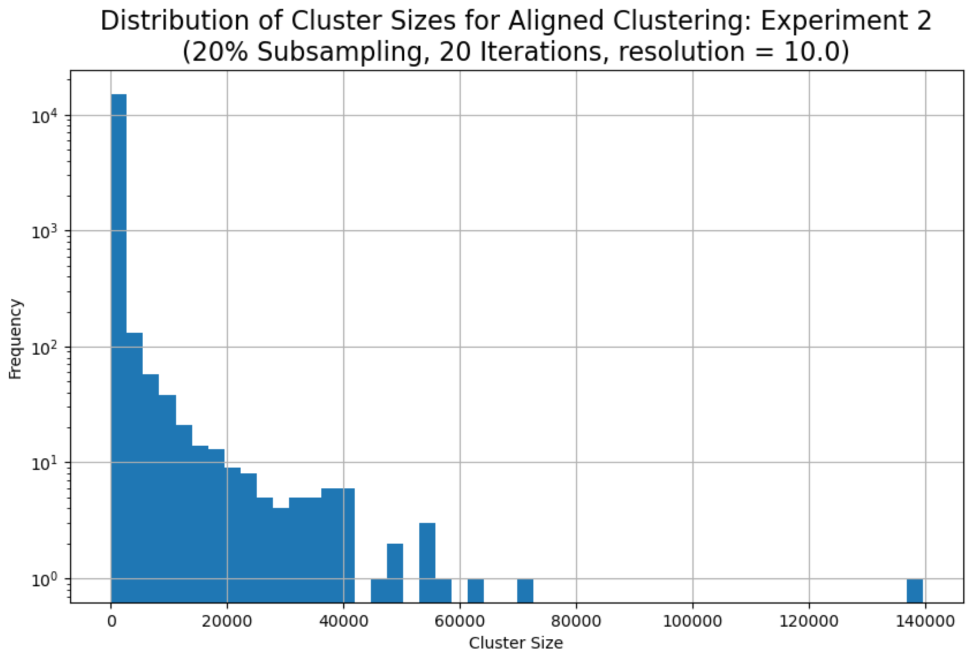

Figure 4.3: **Clustering Experiment 2 Aligned:** Distribution of cluster sizes after running alignment algorithm (Algorithm 2) on **Clustering Experiment 2**.

4.2.3 Assessing Clustering Quality

While the two experiments in subsampled clustering (at resolutions 1.0 and 10.0) successfully produced high-coverage groupings of proteins, inspection of the resulting cluster size distributions revealed the presence of several large, coarse-grained clusters. These oversized groupings are unlikely to represent fine-grained functional similarity due to their broad composition. To address this, we applied Algorithm 3 to further refine the clustering results by re-clustering the aligned clusters. This hierarchical approach helps subdivide heterogeneous groups into smaller, more functionally coherent subclusters, improving the biological relevance and quality of the similarity labels used for downstream tasks such as SNN training.

The refinement algorithm was applied to Experiment 1 with a resolution of 5.0, as shown in Figure 4.4. Most clusters fall within the small-to-moderate size range, indicating improved granularity in the functional grouping of proteins.

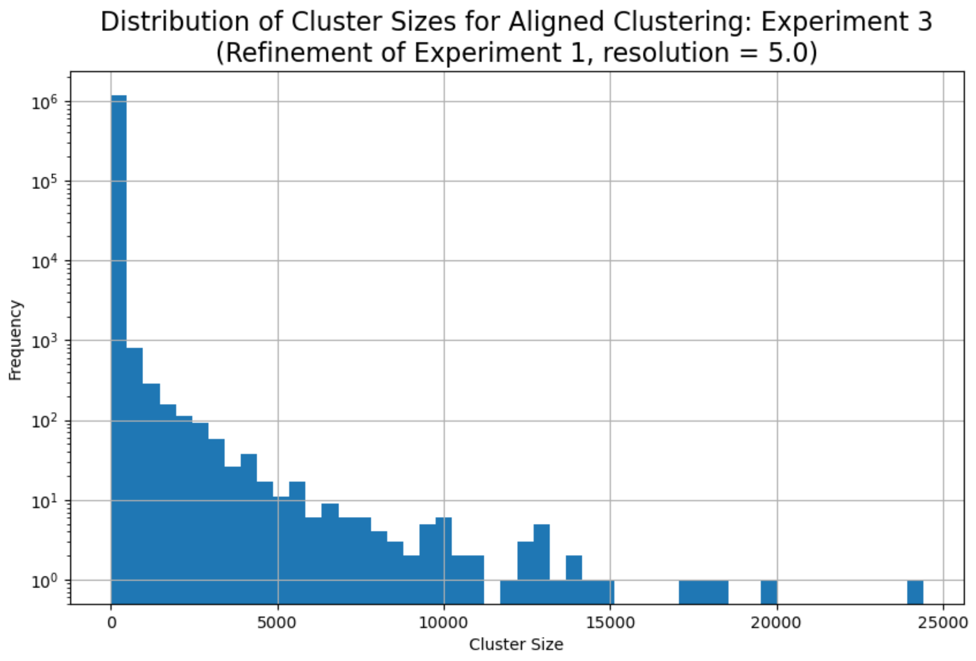

Figure 4.4: **Clustering Experiment 3:** Distribution of cluster sizes after refinement algorithm (Algorithm 3) with resolution = 5.0 on Experiment 1.

Compared to the initial clustering distribution, the refinement step significantly reduced the presence of extremely large clusters. Instead, the resulting cluster size distribution exhibits a sharper drop-off, with the vast majority of clusters falling into smaller size ranges. The improved granularity suggests that the refined clusters better capture local sequence-function relationships and offer more reliable labels for similarity learning.

The clusters from **Clustering Experiments 2 and 3** formed the basis for generating the final training dataset for the SNN. Specifically, 2,000,000 protein pairs were sampled from within and across clusters of size 10 to 1,000, with 25% of the dataset drawn from each of the following categories:

1. Pairs from the same cluster and connected by an edge (label = 1)
2. Pairs from the same cluster but not connected (label = 1)
3. Pairs from different clusters but connected (label = 0)
4. Pairs from different clusters and not connected (label = 0)

This stratified sampling ensured that the model received diverse supervision encompassing both similar and dissimilar pairwise comparisons. Additionally, clusters used to evaluate the model (validation and test) were kept separate from the clusters used in training to test the models generalization to unseen data. By leveraging more biologically coherent clusters and incorporating information from the graph structure, the SNN is better positioned to learn a robust and generalizable protein function similarity metric.

4.2.4 SNN Results

Tables and figures relevant to this section are placed in A.3. Of the two SNN experiments carried out, the model that was trained using protein pairs from **Clustering**

Experiment 2 is labeled **SNN-Experiment 1**, while the one trained on protein pairs from **Clustering Experiment 3** is labeled **SNN-Experiment 2**.

Metric	**SNN-Experiment 1**	**SNN-Experiment 2**
Test Loss	0.8926	**0.5184**
Accuracy	0.6327	**0.7701**
Precision	0.6041	**0.7774**
Recall	0.7457	**0.7569**
F1 Score	0.6675	**0.7670**
AUC	0.6758	**0.8472**
Threshold	0.1000	0.3000
Avg. Precision	0.5665	**0.6640**

Table 4.1: SNN performance metrics on data from Clustering Experiments 2 and 3. Bold indicates stronger performance.

The results in Table 4.1 highlight a substantial improvement in SNN performance when trained on protein pairs derived from Clustering Experiment 3 compared to Clustering Experiment 2. Notably, the model trained on Experiment 3 data achieved a significantly lower test loss (0.5184 vs. 0.8926) and higher scores across all major evaluation metrics, including accuracy (77.01% vs. 63.27%), precision (77.74% vs. 60.41%), F1 score (76.70% vs. 66.75%), and AUC (0.8472 vs. 0.6758).

These improvements suggest that the refinement step applied in Clustering Experiment 3, re-clustering within large initial clusters, led to more functionally coherent groupings, thereby enhancing the quality of training data used for the SNN. Additionally, the shift in the optimal decision threshold (from 0.1 to 0.3) and the increase in average precision reflect a more confident and reliable separation between functionally similar and dissimilar protein pairs. Overall, this comparison supports the utility of hierarchical refinement strategies for generating biologically meaningful similarity labels in protein function prediction tasks.

Chapter 5

Analysis

5.1 Limitations and Future Directions

5.1.1 Overfitting and Model Stability

Analysis of the training and validation curves (Figures A.7 and A.10) suggests that the SNN experiences a degree of overfitting. While the training loss continues to decrease steadily, the validation loss begins to rise after a few epochs and plateaus, indicating that the model is likely memorizing the training set rather than generalizing to unseen protein pairs. This is also reflected in the validation F1 score and accuracy curves, which peak early and then gradually decline. These patterns suggest that the model may be learning decision boundaries too tightly fitted to the training data.

To mitigate overfitting, future work could explore incorporating data augmentation strategies, such as amino acid mutations or other forms of input noise for the model to generalize better.

5.1.2 Clustering Label Improvements

The quality of the similarity labels used for SNN training is critical, especially since the current approach relies on unsupervised clustering to approximate functional similarity.

While this provides scalability, the lack of biological ground truth introduces label noise. Future directions include incorporating curated annotations such as Gene Ontology (GO) terms or known protein families to evaluate or improve clustering quality.

5.1.3 Architectural Improvements for Prediction

While the current model leverages a BiLSTM encoder, this architecture may not be optimal for detecting subtle but functionally important differences between sequences, such as single residue mutations that can drastically alter function.[5] Future iterations of this model could benefit from architectures that specifically addresses the edge cases of small degree mutations that significantly affect function.

In summary, while the proposed approach demonstrates the feasibility of using pLM-derived embeddings and SNNs for learning protein functional similarity, addressing overfitting, refining training labels, and improving model architectures will be essential steps toward achieving higher biological accuracy and broader applicability.

Chapter 6

Conclusion

This thesis presents a scalable framework for predicting functional similarity between protein pairs using pLM embeddings and a SNN. By clustering high-dimensional ProtT5 embeddings and generating protein pair labels from the resulting communities, a weakly supervised dataset is created for training a similarity model without requiring curated annotations.

This thesis explored multiple clustering strategies, including a refinement step that produced smaller, more functionally coherent clusters. The results demonstrate that SNN performance is strongly influenced by the quality of these cluster-derived labels. The refined clustering approach yielded higher predictive performance, with a test F1 score of 0.7670 and AUC of 0.8472.

While challenges such as overfitting and noisy labels remain, this study highlights the potential of embedding-based similarity learning for large-scale protein annotation. Future work will explore integrating curated annotations, improving model architectures, and leveraging biologically grounded clustering to further advance protein function prediction.

Appendix A

Appendix

Amino Acid	**Composition (%)**
A (Alanine)	9.59
C (Cysteine)	1.14
D (Aspartic Acid)	5.47
E (Glutamic Acid)	6.26
F (Phenylalanine)	3.79
G (Glycine)	7.49
H (Histidine)	2.19
I (Isoleucine)	5.42
K (Lysine)	5.03
L (Leucine)	9.83
M (Methionine)	2.40
N (Asparagine)	3.67
P (Proline)	4.95
Q (Glutamine)	3.87
R (Arginine)	5.77
S (Serine)	6.45
T (Threonine)	5.49
V (Valine)	7.05
W (Tryptophan)	1.24
Y (Tyrosine)	2.90

Table A.1: Average Amino Acid Composition - Subsampled 10,000 protein dataset

A.1 Cosine Similarity Threshold Experiments

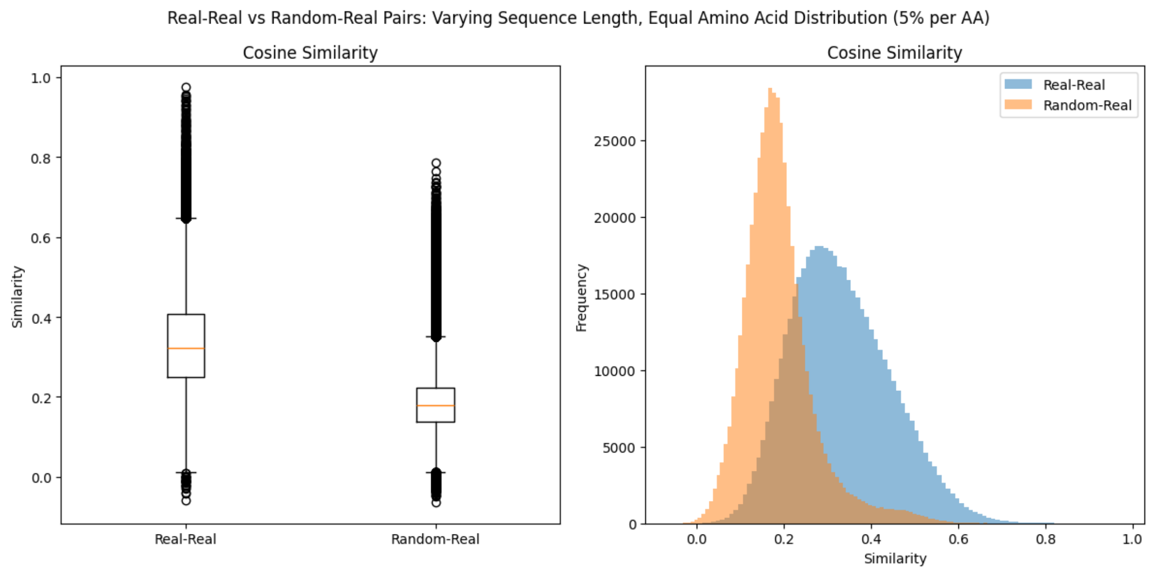

Figure A.1: Comparison of Cosine Similarity Distributions for Real-Real and Random-Real Protein Pairs. The left panel shows a boxplot of cosine similarity scores, while the right panel presents a histogram of similarity distributions. "Real-Real" pairs consist of actual protein sequences, whereas "Random-Real" pairs include embedding cosine similarity of real proteins and randomly generated sequences with varying lengths and an equal amino acid distribution (5% per AA).

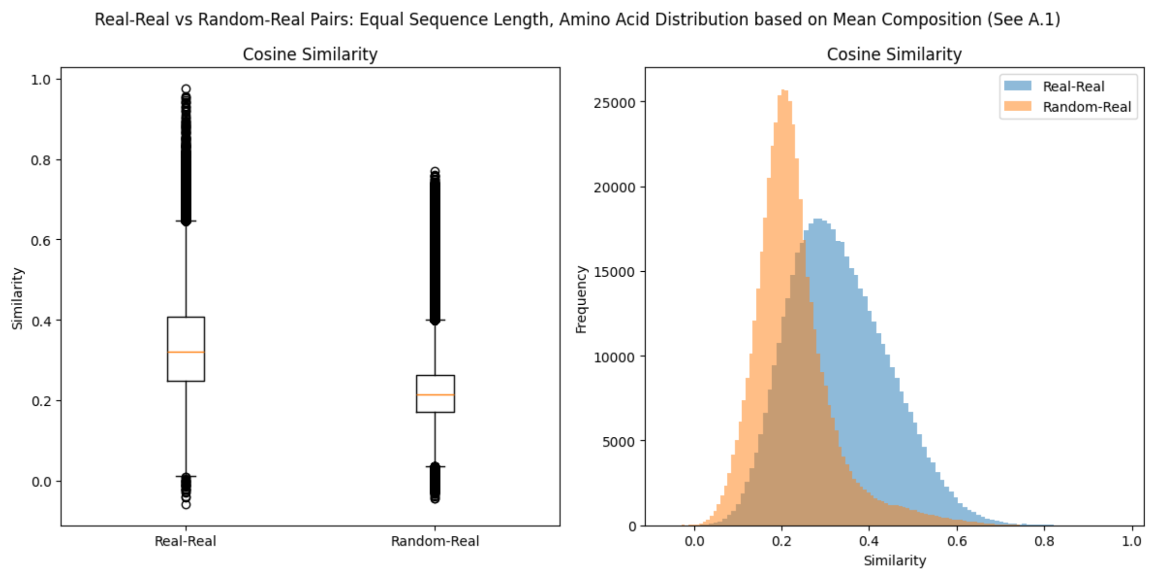

Figure A.2: Comparison of Cosine Similarity Distributions for Real-Real and Random-Real Protein Pairs. The left panel shows a boxplot of cosine similarity scores, while the right panel presents a histogram of similarity distributions. “Real-Real” pairs consist of actual protein sequences, whereas “Random-Real” pairs include embedding cosine similarity of real proteins and randomly generated sequences with equal length and amino acid composition matching the distribution of the subsampled dataset. See A.1

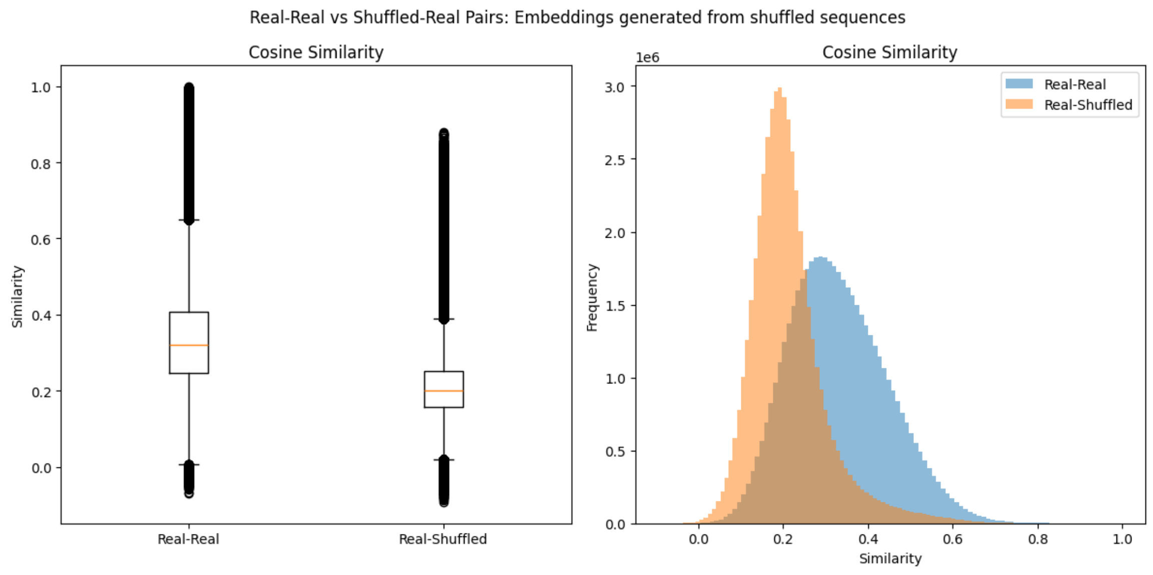

Figure A.3: Comparison of Cosine Similarity Distributions for Real-Real and Shuffled-Real Protein Pairs. The left panel shows a boxplot of cosine similarity scores, while the right panel presents a histogram of similarity distributions. “Real-Real” pairs consist of actual protein sequences, whereas “Real-Shuffled” pairs include embedding cosine similarity of real proteins and shuffled versions of real proteins.

A.2 Clustering Experiments

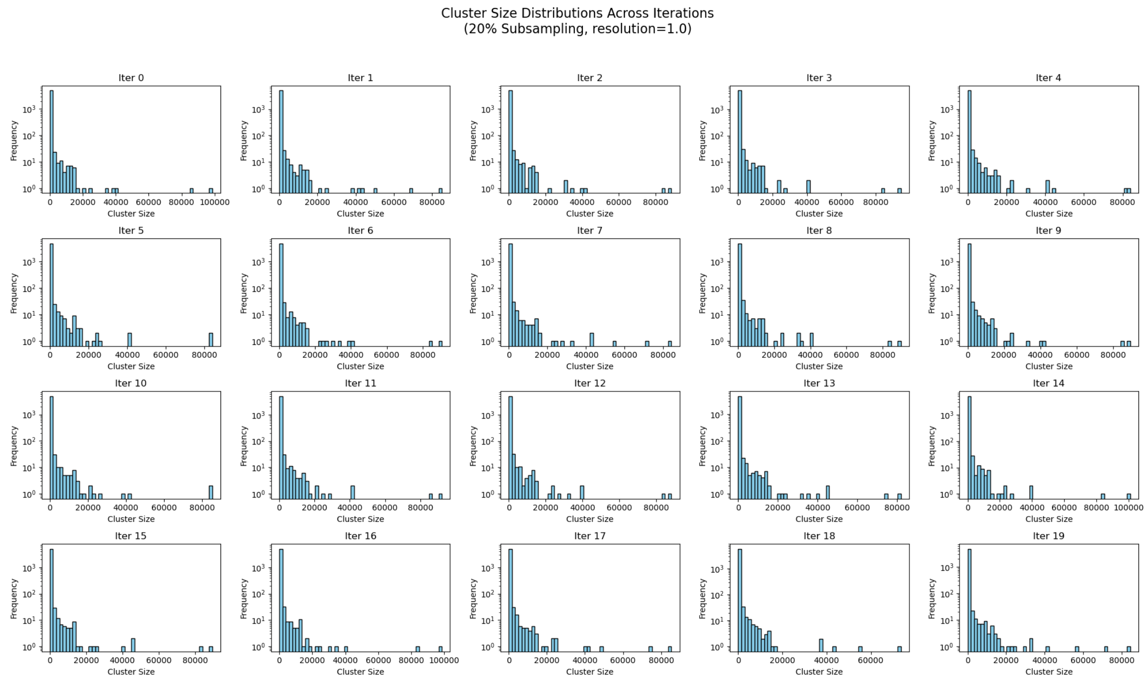

Figure A.4: **Clustering Experiment 1:** Distribution of cluster sizes 20 iterations with 20% subsampled nodes and resolution = 1.0

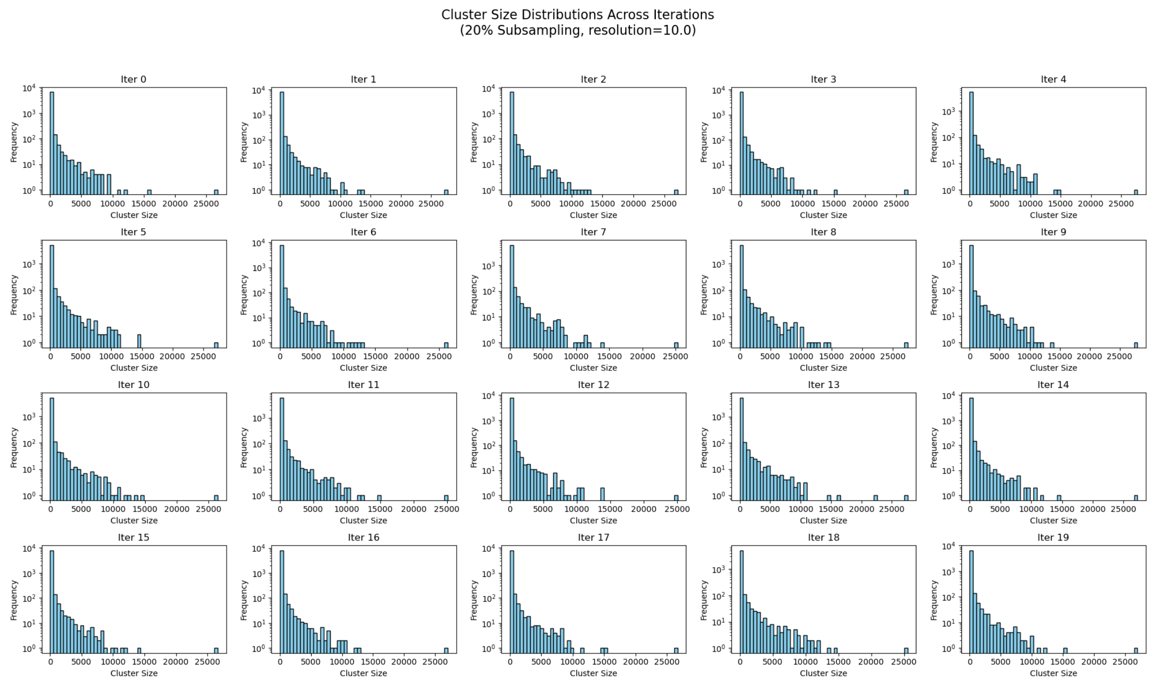

Figure A.5: **Clustering Experiment 2:** Distribution of cluster sizes 20 iterations with 20% subsampled nodes and resolution = 10.0

A.3 SNN Experiments

```
hyperparameters:
batch_size: 512
learning_rate: 0.0005
epochs: 30
vocab_size: 21
embedding_dim: 64
lstm_hidden_dim: 128
lstm_layers: 4
dropout: 0.5
max_seq_length: 1200
num_workers: 8
```

```
pin_memory: True
```

A.3.1 SNN Experiment 1

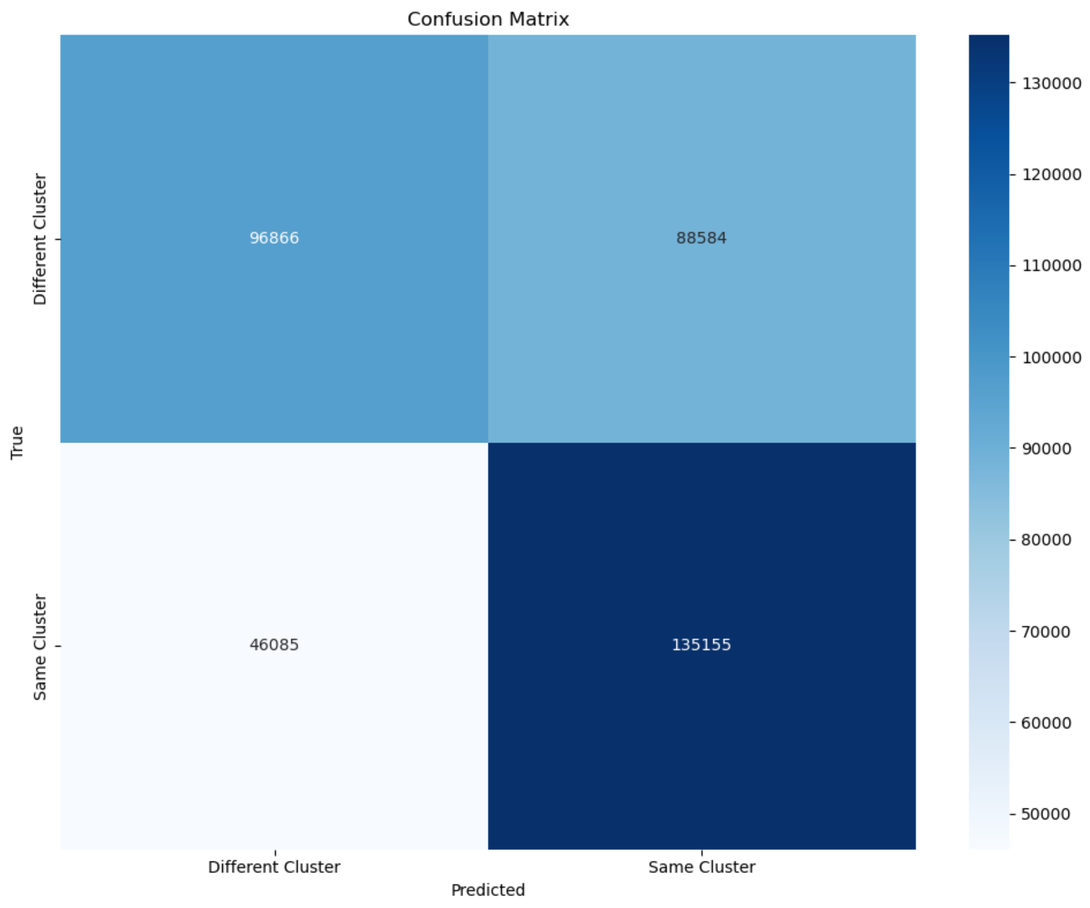

Figure A.6: Confusion Matrix: Test Set Evaltuation on SNN-Experiment 1

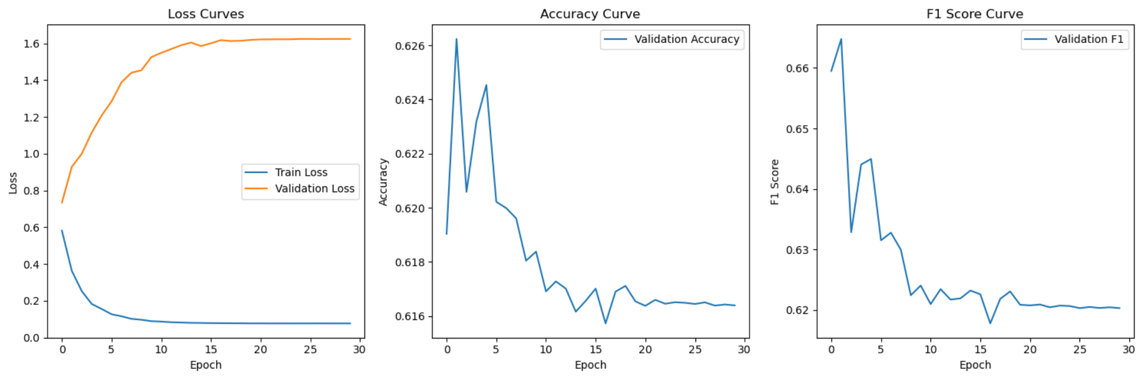

Figure A.7: Training Curves: SNN-Experiment 1

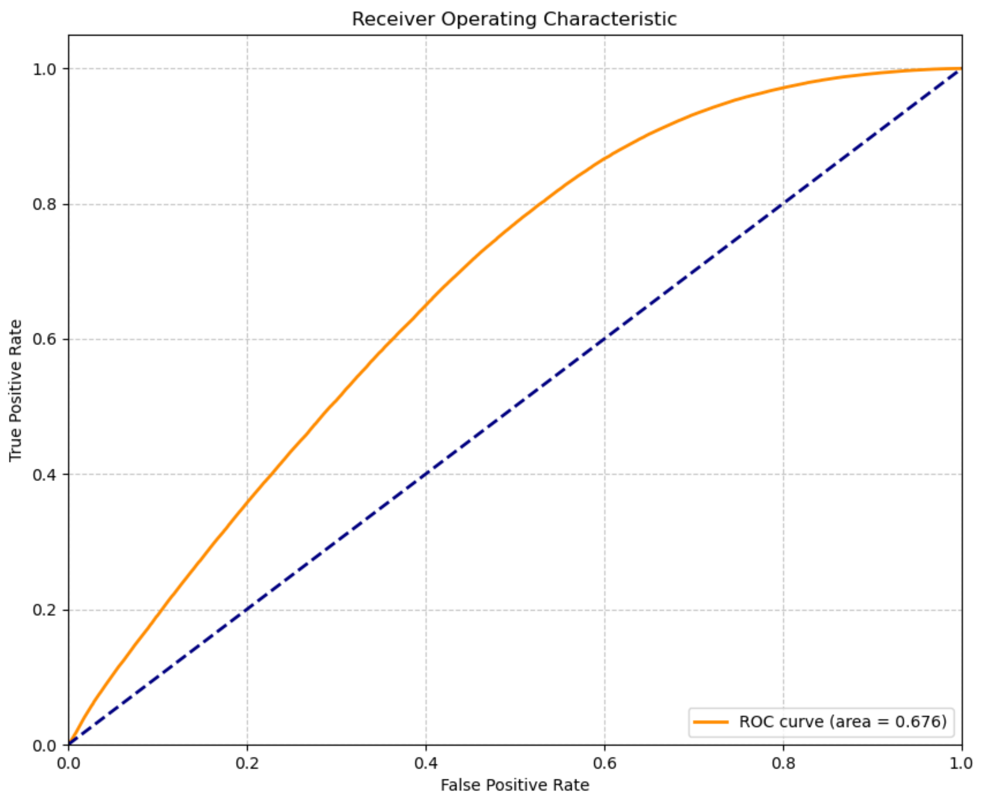

Figure A.8: ROC Curve: Test Set Evaluation on SNN-Experiment 1

A.3.2 SNN Experiment 2

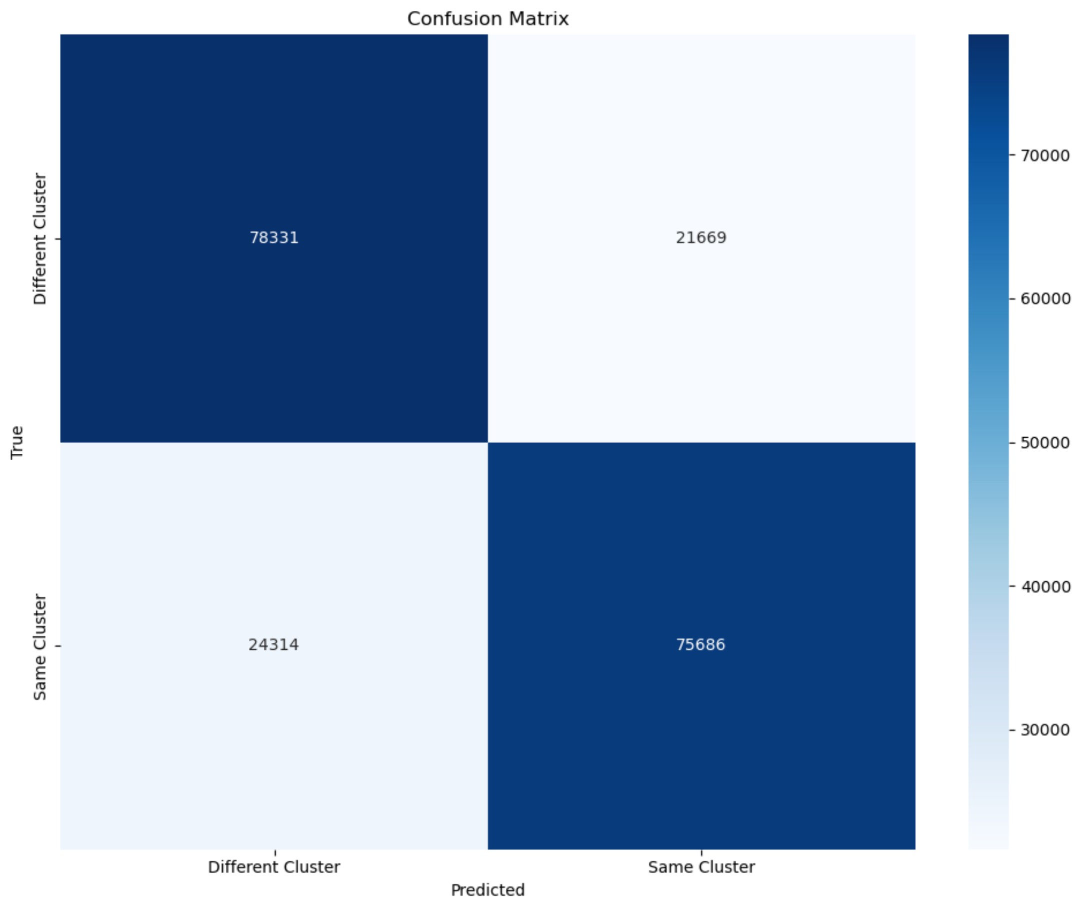

Figure A.9: Confusion Matrix: Test Set Evaltuation on SNN-Experiment 2

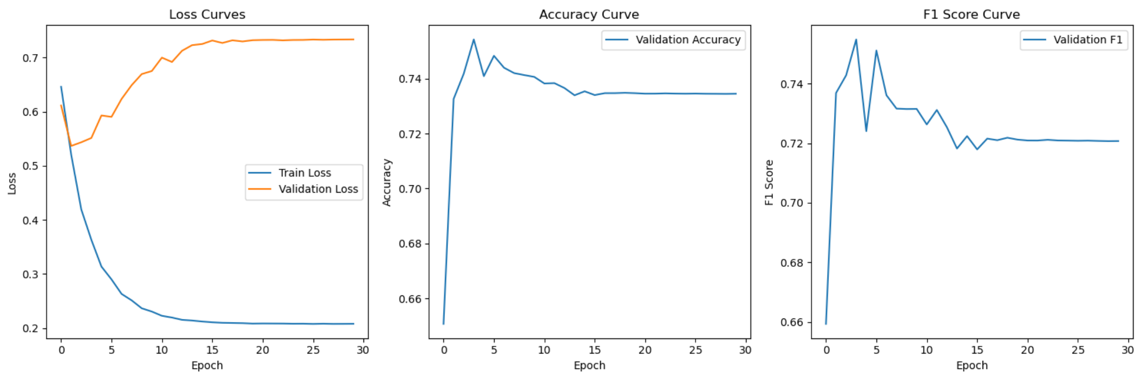

Figure A.10: Training Curves: SNN-Experiment 2

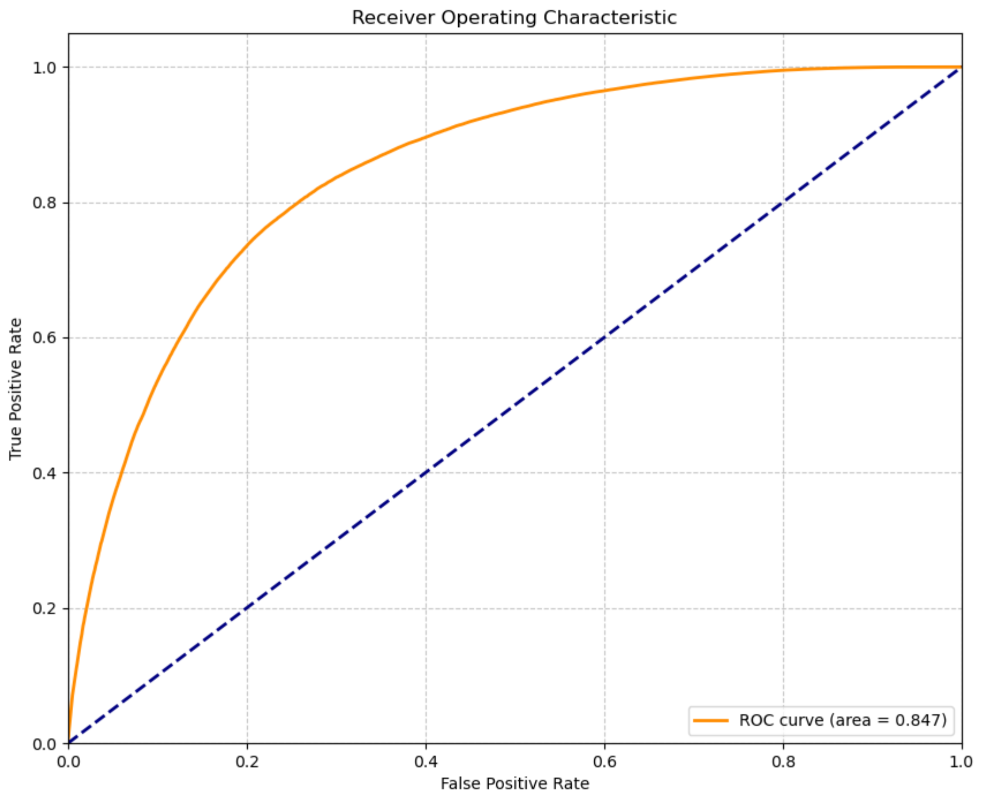

Figure A.11: ROC Curve: Test Set Evaluation on SNN-Experiment 2

Bibliography

[1] Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, and Peter Walter. *Molecular Biology of the Cell.* Garland Science, New York, 4th edition, 2002. URL https://www.ncbi.nlm.nih.gov/books/NBK26820/. Available from NCBI Bookshelf.

[2] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008. URL http://arxiv.org/abs/0803.0476v2.

[3] Jiaxiao Chen, Zhonghui Gu, Luhua Lai, and Jianfeng Pei. In silico protein function prediction: the rise of machine learning-based approaches. *Med. Rev.*, 3(6):487–510, 2023. doi: 10.1515/mr-2023-0038. URL https://doi.org/10.1515/mr-2023-0038.

[4] Davide Chicco. *Siamese Neural Networks: An Overview*, pages 73–94. Springer US, New York, NY, 2021. ISBN 978-1-0716-0826-5. doi: 10.1007/978-1-0716-0826-5_3. URL https://doi.org/10.1007/978-1-0716-0826-5_3.

[5] Suzanne Clancy. Genetic mutation, 2008. URL https://www.nature.com/scitable/topicpage/genetic-mutation-441/.

[6] The UniProt Consortium. Uniprot: the universal protein knowledgebase in 2025.

Nucleic Acids Research, 53(D1):D609–D617, 11 2024. ISSN 1362-4962. doi: 10.1093/nar/gkae1010. URL `https://doi.org/10.1093/nar/gkae1010`.

[7] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library. 2024.

[8] Nature Education. Protein structure, 2014. URL `https://www.nature.com/scitable/topicpage/protein-structure-14122136/`. Accessed: 2025-03-04.

[9] Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Wen Yu, Muntasir Wahed Nabi, Zhiqiang Liu, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Clemens Angerer, Martin Steinegger, Debsindhu Bhowmik, and Burkhard Rost. Prottrans: Toward understanding the language of life through self-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):7112–7127, 2021. doi: 10.1109/TPAMI.2021.3080866. URL `https://arxiv.org/abs/2007.06225`.

[10] Vladimir Gligorijević, P. Douglas Renfrew, Tomasz Kosciolek, Julia Koehler Leman, Daniel Berenberg, Tommi Vatanen, Chris Chandler, Bryn C. Taylor, Ian M. Fisk, Hera Vlamakis, Ramnik J. Xavier, Rob Knight, Kyunghyun Cho, and Richard Bonneau. Structure-based protein function prediction using graph convolutional networks. *Nature Communications*, 12:3168, 2021. doi: 10.1038/s41467-021-23303-9. URL `https://doi.org/10.1038/s41467-021-23303-9`.

[11] Benjamin Giovanni Iovino and Yuzhen Ye. Protein embedding based alignment. *BMC Bioinformatics*, 25:85, 2024. doi: 10.1186/s12859-024-05699-5. URL `https://doi.org/10.1186/s12859-024-05699-5`.

[12] David Lee, Oliver Redfern, and Christine Orengo. Predicting protein function from

sequence and structure. *Nature Reviews Molecular Cell Biology*, 8(12):995–1005, 2007. doi: 10.1038/nrm2281. URL `https://doi.org/10.1038/nrm2281`.

[13] Nature Education. Dna is constantly changing through the process of mutation, 2014. URL `https://www.nature.com/scitable/topicpage/dna-is-constantly-changing-through-the-process-6524898/`. Accessed: 2025-03-04.

[14] Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. Learning text similarity with siamese recurrent networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 148–157, Berlin, Germany, August 2016. Association for Computational Linguistics. URL `https://aclanthology.org/W16-1617/`.

[15] Amit Singhal. Modern information retrieval: A brief overview. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 2001. URL `https://ieeexplore.ieee.org/document/956348`.

[16] Chengxin Zhang and Lydia Freddolino. A large-scale assessment of sequence database search tools for homology-based protein function prediction. *Briefings in Bioinformatics*, 25(4):bbae349, 2024. doi: 10.1093/bib/bbae349. URL `https://doi.org/10.1093/bib/bbae349`.

[17] Wei Zheng, Le Yang, Robert J. Genco, Jean Wactawski-Wende, Michael Buck, and Yijun Sun. Sense: Siamese neural network for sequence embedding and alignment-free comparison. *Bioinformatics*, 35(11):1820–1828, 2019. doi: 10.1093/bioinformatics/bty887. URL `https://academic.oup.com/bioinformatics/article/35/11/1820/5146727`.