**Protein Classification Using Embeddings from Language Models Trained on Amino Acid Sequences**

By

Emmanuel Ndubuisi

**A Thesis Submitted in Partial Fulfillment of the Requirements**

**For the Degree of Bachelor of Science in the**

Department of Mathematics and Computers Science

Claflin University

Orangeburg, South Carolina

August 2021

 

Dr. Shrikant Pawar                                                           Dr. Karina Liles

Thesis Advisor Committee Member

 

Dr. Ramaier Sriram               Dr. Karina Liles

Committee Member Academic Advisor

Received by:  Date: 

Dr. Karina Liles, Chairperson, Department of Mathematics and Computer Science

Received by:  Date: 

Dr. Verlie Tisdale, Dean, School of Natural Sciences and Mathematics

Received by: Date: 

Dr. Angela Peters, Vice Provost for Academic Affair

TABLE OF CONTENTS

[ACKNOWLEDGEMENTS 2](#_Toc98092306)

[ABSTRACT 3](#_Toc98092307)

[KEYWORDS AND ABBREVIATIONS 4](#_Toc98092308)

[LIST OF TABLES 5](#_Toc98092309)

[LIST OF FIGURES 6](#_Toc98092310)

[INTRODUCTION 7](#_Toc98092311)

[LITERATURE REVIEW 9](#_Toc98092312)

[**TRANSFORMER MODELS AND TRANSFER LEARNING** 9](#_Toc98092313)

[**TRANSFORMERS MEET BIOLOGY** 11](#_Toc98092314)

[METHODOLOGY 13](#_Toc98092315)

[**EXPLORATORY DATA ANALYSIS** 13](#_Toc98092316)

[**MODEL ARCHITECTURE & DESIGN** 16](#_Toc98092317)

[**TRAINING SETUP** 17](#_Toc98092318)

[**HARDWARE AND SOFTWARE** 18](#_Toc98092319)

[RESULTS 20](#_Toc98092320)

[**PERFORMANCE & ACCURACY** 20](#_Toc98092321)

[**INTEPRETABILITY** 22](#_Toc98092322)

[CONCLUSION 24](#_Toc98092323)

[REFERENCES 26](#_Toc98092324)

# **ACKNOWLEDGEMENTS**

I would like to thank my supervisor, Dr. Shrikant Pawar, for his insightful comments and suggestions. I would also like to thank my committee members, Dr. Karina Liles and Dr. Ramaier Sriram, for their guidance and support throughout this process. Lastly, I would like to give a special thanks to my family and friends for their encouragement and unwavering support.

# **ABSTRACT**

Proteins are complex organic compounds that serve as the building blocks of life. There are many different proteins in any given organism and each one has a specific function. Understanding the relationship between amino acid sequences and protein function is crucial for scientists who are trying to study the functional effect of protein mutations for disease discovery or for developing novel proteins for medical use. Traditional techniques for protein function prediction such as Hidden Markov Model (HMMs) [1] and BLASTp [2] have fallen short because they rely on nearest neighbor search algorithms that perform poorly on large datasets and are sensitive to noisy and high-dimensional data. While recent techniques for protein function prediction that rely on large ensembles of Convolutional Neural Networks (CNN) outperforms traditional approaches, these systems are impractical for many scientists because of the computational power required to use large ensemble models [3].

In this paper, I will validate the use of embeddings extracted from a pre-trained transformer model trained on a large corpus of unlabeled protein sequences to create a classification model in order to improve accuracy and reduce the computational overhead required to use deep learning techniques for protein function prediction.

# **KEYWORDS AND ABBREVIATIONS**

**Keywords**: Proteins, amino acid sequences, transformer architecture, multi-class classification, transfer learning, attention mechanism, self-supervised learning, activation layer, Pfam, pooling, mixed precision

**Abbreviations**

MLM – Masked Language Modeling

NSP – Next Sentence Prediction

BERT – Bidirectional Encoder Representations from Transformers

RNN – Recurrent Neural Networks

LSTM – Long Short-Term Memory

CNN – Convolution Neural Network

FNN – Feed-Forward Neural Network

TanH – Hyperbolic Tangent

CUDA – Compute Unified Device Architecture

vRAM – Virtual Random Access Memory

GPU – Graphical Processing Unit

PCIe – Peripheral Component Interconnect Express

BLOSUM – Blocks Substitution Matrix

MP – Mixed Precision

FP16 – Half-precision Floating Point Format

# **LIST OF TABLES**

[Table 1: The first 5 rows of the randomly split Pfam data. 16](#_Toc98024324)

[Table 2: Performance evaluation results for model. 23](#_Toc98024325)

[Table 3: Model performance benchmarked against existing models. 24](#_Toc98024326)

# **LIST OF FIGURES**

[Figure 1: The self-attention mechanism showing the positional relationship for the sentence “Claire walks her dog to the park.” 12](#_Toc98068973)

[Figure 2: An illustration of the process of transfer learning in machine learning. 13](#_Toc98068974)

[Figure 3: Sequence length distribution of the protein sequences in the train dataset. 17](#_Toc98068975)

[Figure 4: Sequence length boxplot of the protein sequences in the train dataset. 18](#_Toc98068976)

[Figure 5: Frequency of each amino-acid code in the train data. 18](#_Toc98068977)

[Figure 6: Architecture of the protein classification model. 19](#_Toc98068978)

[Figure 7: Matrix derived from the effect of mutation on the model's prediction confidence on a particular protein sequence. 25](#_Toc98068979)

[Figure 8: Blocks Substitution Matrix (Blosum50). 25](#_Toc98068980)

# **INTRODUCTION**

Proteins are the largest and most complex molecules in the body. They are responsible for the nearly every biochemical process that occurs in cells such as energy production, gene expression regulation, and message transmission between cells. Protein molecules are composed of amino acids linked together with peptide bonds. The sequence of amino acids in a protein determines its three-dimensional structure, which can be used by scientists to determine the protein's function [4]. For example, the hemoglobin protein, which is responsible for carrying oxygen in the blood, has a specific sequence of amino acids that allows it to fold into a specific three-dimensional structure [5].

Predicting the function of a protein from its amino acid sequence is one of the most important tasks in bioinformatics because of its wide-ranging applicability for other crucial downstream tasks like drug discovery and disease prevention. While the function of a protein can be inferred from its 3D structure, accurately determining the 3D structure of a protein through biological experiments such as x-ray crystallography is an extremely tedious task [6]. Fortunately, the function of a protein can also be predicted from only its amino acid sequence since proteins with similar sequences often have similar functions [7]. For instance, proteins that are involved in transcription often have a helix-turn-helix motif [8].

Traditional methods for functional prediction of unknown protein sequences compares the unknown protein to sequences of identified proteins using algorithms such as BLASTp, or Hidden Markov Models (HMMs) built on sequences provided in databases like Pfam; a database of protein families and domains created to provide a comprehensive and consistent collection of protein families and their associated functional annotation [9]. Unfortunately, these traditional techniques fail to annotate at least one-third of microbial proteins [10]. Additionally, these techniques rely on proximity search algorithms such as nearest neighbor which performs poorly on large datasets and are sensitive to noisy and high-dimensional data. The computational cost associated with the inability of proximity search algorithms to scale linearly is very prohibitive because of the exponential rate at which proteins are being sequenced due the cost decline of sequencing [11].

Although there has been recent work that demonstrates a large ensemble of convolutional neural networks (CNNs) solves the challenges associated with traditional prediction techniques and even improves prediction accuracy by an order of magnitude, this deep learning technique is still not widely adopted by scientists because of the computational overhead associated with training and making predictions using large ensemble models.

This paper describes another deep learning approach for protein function prediction that doesn't require the computational cost of training and inferencing on a large ensemble model. The method described in this paper involves fine-tuning a transformer protein model, ProtBert, pre-trained on a large corpus of unlabeled protein sequences [12]. For this work, the ProtBert model was fine-tuned to create a multi-class classification model using the sequences and their functional class labels present in the Pfam database.

# **LITERATURE REVIEW**

In this section, I briefly review previous work related to this paper such as the concept of transformer models and its growing use in natural language processing tasks. I also contrast the advantages of using the transformer architecture over other deep learning architectures such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs). Finally, I review work that demonstrates how the transformer architecture can be used to develop foundational protein models that captures the biophysical nature of protein sequences and can be used for downstream tasks.

## **TRANSFORMER MODELS AND TRANSFER LEARNING**

Transformer models are deep learning models that are built with the self-attention mechanism [13]. This allows the model to consider the context and position of each part of an input data with respect to the entire data to make more accurate predictions, improve performance, and minimize information loss. The transformer attention mechanism can also be used to visualize the learned dependencies the model utilized to make predictions.

Unlike recurrent neural networks (RNNs), transformer models can learn to map different sequences of inputs to a desired sequence of outputs without needing to process the inputs sequentially. Additionally, these models can simultaneously consider both previous and future elements of a data without requiring twice the computational power needed for bidirectional RNNs to perform the same tasks [14].

Diagram, schematic

Description automatically generated

Figure : The self-attention mechanism showing the positional relationship for the sentence “Claire walks her dog to the park.”

Transformer models are trained on a large corpus of raw data in a self-supervised fashion where the training objective is independently learned from the model's input without the need for human labeled data. For example, the Bidirectional Encoder Representations from Transformers (BERT) model was trained using the masked language modeling (MLM) and the next sentence prediction (NSP) training objectives [15]. While transformer models can independently learn a mathematical representation of the data they are trained on, they are not always useful for any particular task. As such, these models must go through an additional supervised process called transfer learning [16].

Diagram

Description automatically generated

Figure : An illustration of the process of transfer learning in machine learning.

Transfer learning is one of the most important and widely used techniques in deep learning. It is a technique where a deep neural network is pre-trained on a large dataset, and then a smaller dataset is used to fine-tune the network for a specific task. This is advantageous because the pre-trained network has already learned to recognize certain patterns in the data, so the task-specific dataset can be processed more quickly and accurately. For example, a model pre-trained on a large corpus on English language text can be quickly adapted for the task of poem generation if it's fine-tuned on only a small amount of poem data.

## **TRANSFORMERS MEET BIOLOGY**

The work in this paper is based on the notion that the embeddings derived from transformer models, such as ProtBert, contains certain biophysical features that can be utilized for various downstream tasks through transfer learning. ProtBert is an auto-encoder transformer model pre-trained in a self-supervised fashion on a large corpus of protein sequences from Uniref100 [17], a dataset consisting of 217 million protein sequences. Unlike the original BERT model, ProtBert was only pre-trained using the masked language modeling objective and not next sentence prediction, since it treats each protein sequence as a complete document.

The researchers behind the ProtBert model validated the utility of training large language models on the vast amount of unlabeled data containing millions of protein sequences and up to 393 billion amino acids [12]. The ProtBert model and other models developed in the study achieved state-of-the-art performance in prediction accuracy and inference cost on several tasks; proving that language models trained on protein data can outperform traditional proximity search models that rely on evolutionary information. Some of the tasks includes per-residue prediction of protein secondary structure (3-state accuracy Q3=81%-87%); per-protein predictions of protein sub-cellular localization (ten-state accuracy: Q10=81%) and membrane vs. water-soluble proteins (2-state accuracy Q2=91%) [12].

In addition, Rives et al. demonstrates that the attention mechanism of a Transformer model trained to perform masked language modeling on amino acids can capture the high-level structural and functional properties of proteins [18]. Transformers have also demonstrated the capability of generating novel and viable protein sequences that match a certain molecular function or cellular components [19].

These results have shown the promise of applying Transformers to perform sequence level analysis for much tedious tasks such as the prediction of the functional family of a protein sequence.

# **METHODOLOGY**

## **EXPLORATORY DATA ANALYSIS**

For this experiment, I used a pre-partitioned version of the Pfam dataset available on Kaggle randomly split into training, evaluation, and testing folds [20]. Each fold has several files that contains these field:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **family\_id** | **sequence\_name** | **family\_accession** | **aligned\_sequence** | **sequence** |
| Penicillinase\_R | Q81U16\_BACAN/8-123 | PF03965.16 | ISEAELEIMKVL… | ISEAELEIMKVL… |
| Rtt106 | POB3\_CANAL/362-454 | PF08512.12 | AGVPCSVKA… | AGVPCSVKA… |
| F-actin\_cap\_A | Q8I3I2\_PLAF7/12-301 | PF01267.17 | IRHVLMNSPP… | IRHVLMNSPP… |
| HupF\_HypC | O28902\_ARCFU/1-65 | PF01455.18 | MCIAIPGR...I.ER..IDY… | MCIAIPGRIERIDY… |
| DUF3794 | R6BY75\_9CLOT/189-271 | PF12673.7 | NIFHI..LWEDVDL… | NIFHILWEDVDL… |

Table : The first 5 rows of the randomly split Pfam data.

1. The *family\_id* column is family name of the corresponding protein sequence.
2. The *sequence\_name* is the accession ID of the sequence.
3. The *family\_accession*contains the labels for the model in the form of PFxxxxx.y where xxxx is the family accession, and y is the version number.
4. The *aligned\_sequence* contains a single sequence from the multiple sequence alignment.
5. The *sequence* column is the training feature for the model. This column contains a list of sequences made up of 20 different amino acids.

This data is described in-depth in the publication: "Deep Learning Classifies the Protein Universe.", Bileschi et al [21].

While the ProtBert model was pre-trained using a sequence length of 512 and 2048 [12], I performed an exploratory data analysis to understand the sequence length distribution of the sequences in the Pfam dataset and discovered that the length for most of the sequences lie between 70 and 200. As such, I set the maximum sequence length for the model to 1536 to reduce the training time while considering the full sequence length of more than 90% of the sequences in the training set. This meant that longer sequences were truncated, and shorter sequences were padded with empty tokens. The attention layers of the sequences were then set to ignore the padding tokens by using an attention mask to get the same encodings when passing sequences with varying lengths through the model.

Chart, line chart

Description automatically generated

Figure : Sequence length distribution of the protein sequences in the train dataset.

Chart, box and whisker chart

Description automatically generated

Figure : Sequence length boxplot of the protein sequences in the train dataset.

Exploratory data analysis also showed that certain amino acid codes, "B, O, U, and Z", rarely occurred in the sequences, so following the technique used in the ProtBert [12] paper, these amino acids were mapped to a designated unknown code "X".

Chart, timeline, bar chart

Description automatically generated

Figure : Frequency of each amino-acid code in the train data.

Finally, while there are 17,929 unique *family\_accession* labels in the Pfam dataset, I only consider the 1000 most frequent classes in the experiment outlined in this paper because of limited computational power.

## **MODEL ARCHITECTURE & DESIGN**

The model for this experiment was built from a pre-trained protein model, ProtBert [12], using transfer learning for sequence classification, that is, the representations generated from the model was used for the downstream protein family classification task.

Diagram

Description automatically generated

Figure : Architecture of the protein classification model.

The authors of this model claim it can take any protein sequence as input (given its length is no more than 40000) and return a vector representation that captures some of the sequence's biophysical features suitable for downstream tasks. To use the embeddings extracted from this model for the Pfam classification task, its pre-training head was replaced with a pooling layer and a randomly initialized classification head.

The pooling layer is used to reduce the dimensions of the features generated by the encoder layer to a single, fixed-size vector. In the BERT architecture, the pooling layer takes the representation for the first token [CLS] to represent the whole sequence. This flattened representation aims to extract only useful information from embeddings to help reduce the amount of computation performed by the neural network. The global-average pooling method was used for this model [22].

The resulting embedding from the pooling layer is then used as input into the classification layer. The classification head is a simply a combination of a linear layer and a hyperbolic tangent (TanH) activation layer. The linear layer specifies the number of input and output features; the input features is the size of the feed-forward layer in the ProtBert encoder, and the output features is the size of the prediction classes. The output of the linear layer is then applied to the activation layer and an increasing function to predict probability.

## **TRAINING SETUP**

For the training, the embeddings from the last layer of the ProtBert language model are pooled to generate a fixed-size representations for all the protein sequences. The pooling method ignores any padding tokens generated by the ProtBert tokenizer. The resulting vector representation from the pooling layer is then used as an input to a single fully connected feed forward layer with 32 neurons which makes the final predictions. The model is trained to minimize the cross-entropy loss between the predictions from its final layer and the ground truth protein family label.

While multiple training experiments were performed with varying performance, the model architecture remained the same, and as such, the performance difference for each model is attributed to the hyper-parameters used during training. Six training experiments were performed varying the hyper-parameters for this model. The models during each experiment were trained for a total of 1726 steps and a single epoch using a learning rate ranging from 3e-06 to 3e-05, a batch size of 1, and a maximum sequence length ranging from 256 to 1536. To reduce computational time, the model encoder layers were frozen while the classification layer remained open to weight modification for all the experiments. This also ensure that the ProtBert layers retain the knowledge they acquired during pre-training. Finally, the Adam optimizer was used to find the best set of parameters for the model and update its weights accordingly.

## **HARDWARE AND SOFTWARE**

The models with 100 protein classes were trained on two different types of machines. The first machine used was a single Tesla P100 with 3584 CUDA cores and 16GB HBM2 vRAM linked via a 4096-bit PCIe brick. The second machine used was a single NVIDIA Tesla V100-NVLink (3x faster than the P100) with 5120 CUDA cores and 16 GB HMB2 vRAM operating at 900 GB/s. A single NVIDIA A100-PCIE-40GB GPU (the most powerful deep learning GPU, 2.5x faster than V100s) was used to trained with the complete dataset containing 17,929 protein classes and over million rows.

The model was trained using the PyTorch Lightning framework, an open-source machine learning framework that abstracts the complexity of training large-scale deep learning models and allows for fast iteration. The model was also trained using the Nvidia’s APEX library for mixed precision; a deep learning technique that combines low-precision and high-precision computation. This makes it possible to fit large models into a single GPU memory thus reducing the memory required to train and improving performance and accuracy.

# **RESULTS**

## **PERFORMANCE & ACCURACY**

The protein classification model in this paper was evaluated on its accuracy. Six experiments were performed varying the model’s hyper-parameters. The learning rate and sequence length were the two hyper-parameters found to significantly affect the model’s performance. These experiments only consider the top 1000 most frequent protein family in the Pfam dataset. However, a 7th and 8th training experiment were conducted to test the model’s performance on all the 17,929 classes in the dataset. These full experiments were conducted with several performance trade-offs to make training on the limited computational resources feasible. These trade-offs include reducing the protein sequence length to 256 and using a half-precision floating point format (FP16) for computing the model weights. The batch size and epoch were set to 1 for all the experiments. Table 2 outlines the results of the experiments performed.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Device | Training Time | Learning Rate | Precision | Sequence Length | Test Accuracy | Validation Accuracy |
| Tesla P100-PCIE-16GB | 1h 47m 24s | 3e-05 | MP | 1024 | 0.9991 | 0.9985 |
| 1h 49m 11s | 3e-03 | MP | 1536 | 0.1002 | 0.1005 |
| 1h 45m 30s | 3e-05 | MP | 512 | - | 0.9991 |
| NVIDIA Tesla V100-SXM2-16GB | 1h 14m 10s | 3e-05 | MP | 1536 | 0.9993 | 0.9993 |
| 1h 13m 52s | 3e-04 | MP | 2048 | 0.8646 | 0.8646 |
| 1h 13m 58s | 3e-06 | MP | 2048 | 0.9939 | 0.9911 |
| 20h 20m 5s | 3e-05 | FP16 | 256 | 0.728 | 0.704 |
| NVIDIA A100-PCIE-40GB | 24h 15m 40s | 3e-05 | FP16 | 512 | 0.891 | 0.883 |

Table : Performance evaluation results for model.

The best experiment yielded a model that achieved an error rate of 0.0007% when trained on 100 unique protein classes; compared to the state-of-the-art, ProtCNN, which only achieves an error rate of 0.012% on 100 classes. However, ProtCNN, trained on 17,929 classes achieves an error rate of 0.495% and the ensemble model, ProtENN, achieves 0.15% [3]. However, since the model described in this paper was trained with significant performance trade-offs when considering all 17929 classes, it performs poorly with an error rate of 10.9%. With the adequate computational resources, the error rate after training should significantly decrease.

|  |  |
| --- | --- |
| Model | Error Rate |
| Top Pick HMM | 1.414 |
| phmmer | 1.531 |
| BLASTp | 1.654 |
| k-mer | 9.994 |
| RNN | 1.8 |
| ProtCNN | 0.495 |
| ProtENN | 0.159 |
| Model Result | **10.9** |
| ProtCNN (100 classes) | 0.012 |
| Model Result (100 classes) | **0.0007** |

Table : Model performance benchmarked against existing models.

## **INTEPRETABILITY**

Deep learning models are notorious for being a black box. While you can observe the model’s behavior on a superficial level, they are often difficult to understand or interpret. This interpretability study shows that the model in described in this paper recognizes the evolutionary consequences and biological effect of mutations in protein sequences. The Block Substitution Matrix is a matrix of pairwise substitution probability between amino acids in a protein sequence. It assigns a score to every possible substitution between two amino acids. This score reflects the likelihood that the substitution will occur in nature. For example, the hydrophobic interaction between the Methionine and Isoleucine amino acid is known to stabilize the bonding of a protein in its folded state. As such, a Methionine to Leucine substitution in a protein sequence does not alter the protein because this substitution conserves the protein’s hydrophobic interaction since they are both highly hydrophobic. However, a Methionine to Arginine substitution will disrupt this hydrophobic interaction because Arginine is charged [23]. Therefore, Methionine to Leucine substitution is more likely to occur than a Methionine to Arginine substitution.

Chart, scatter chart

Description automatically generatedChart, scatter chart

Description automatically generated

Figure : Matrix derived from the effect of mutation on the model's prediction confidence on a particular protein sequence.

Figure : Blocks Substitution Matrix (Blosum50).

To study this effect, the model was tasked with predicting a protein sequence, and then each amino acid code in the sequence were substituted with another code. After each substitution, the deviation from the model’s original prediction confidence was normalized and added to a matrix under the corresponding location (i.e., given a matrix , the normalized prediction difference will be stored under where *o* is original amino-acid code and *m* is the mutated code). This matrix was then compared with the *Blosum50* and showed a significant structural similarity.

# **CONCLUSION**

In this work, a transfer learning approach of fine-tuning a large language model pre-trained on protein sequences for subsequent protein classification task is benchmarked against other traditional classification methods such as HMMs and BLASTp. It is also compared to deep learning approaches such as using a CNN, ProtCNN, and an ensemble of CNN models, ProtENN. On randomly split data, the model described in this paper achieves an error rate of 0.0007% when trained to classify 100 protein families; and 10.9% when trained to classify 17,929 protein families. The model surpasses the state-of-the-art classification approach, ProtCNN, which achieves an error of 0.012% when trained to classify 100 protein families. The model in this paper was also trained to classify all 17,929 protein families in the Pfam train dataset but with performance trade-offs to manage the limited computational resources. However, it doesn’t perform well with these trade-offs when compared to the state-of-the-art model which achieves an error rate of 0.495%.

Lastly, work was done to understand if the deep learning model described in this paper exhibited behavior that is consistent with a generally accepted biological phenomenon. The Block Substitution Matrix based on a 50% sequence identity cutoff, *Blosum50*, was compared to a matrix generated from observing the change in the model’s prediction confidence when an amino-acid code was substituted with another code in a particular protein sequence. This experiment generated a matrix with a similar distribution to the *Blosum50* matrix showing that the model recognized the biophysical effect of mutations in a protein sequence.

While the performance demonstrated by the model in the experiments discussed in this paper is impressive. Future work still needs to be done to scale up the model without performance trade-offs. There are also several interpretability studies that need to be performed to understand what kind of protein sequences examples the model performs poorly on; the consistency of the model behavior; and if it can gather adversarial behavior, or unwanted priors from the Pfam training set.

# **REFERENCES**

[] Sonnhammer, E. L., Eddy, S. R., Birney, E., Bateman, A., & Durbin, R. (1998). Pfam: multiple sequence alignments and HMM-profiles of protein domains. *Nucleic acids research*, *26*(1), 320-322.

[2] Jacob, A., Lancaster, J., Buhler, J., Harris, B., & Chamberlain, R. D. (2008). Mercury BLASTP: Accelerating protein sequence alignment. ACM Transactions on Reconfigurable Technology and Systems (TRETS), 1(2), 1-44.

[3] Bileschi, M. L., Belanger, D., Bryant, D. H., Sanderson, T., Carter, B., Sculley, D., ... & Colwell, L. J. (2022). Using deep learning to annotate the protein universe. Nature Biotechnology, 1-6.

[4] Wang, Y., Addess, K. J., Chen, J., Geer, L. Y., He, J., He, S., ... & Bryant, S. H. (2007). MMDB: annotating protein sequences with Entrez's 3D-structure database. Nucleic Acids Research, 35(suppl\_1), D298-D300.

[5] Gribaldo, S., Casane, D., Lopez, P., & Philippe, H. (2003). Functional divergence prediction from evolutionary analysis: a case study of vertebrate hemoglobin. Molecular biology and evolution, 20(11), 1754-1759.

[6] Shi, Y. (2014). A glimpse of structural biology through X-ray crystallography. Cell, 159(5), 995-1014.

[7] Sikosek, T., & Chan, H. S. (2014). Biophysics of protein evolution and evolutionary protein biophysics. *Journal of The Royal Society Interface*, *11*(100), 20140419.

[8] Rosinski, J. A., & Atchley, W. R. (1999). Molecular evolution of helix–turn–helix proteins. Journal of molecular evolution, 49(3), 301-309.

[9] Bateman, A., Coin, L., Durbin, R., Finn, R. D., Hollich, V., Griffiths‐Jones, S., ... & Eddy, S. R. (2004). The Pfam protein families database. Nucleic acids research, 32(suppl\_1), D138-D141.

[10] Morgan N Price, Kelly M Wetmore, R Jordan Waters, Mark Callaghan, Jayashree Ray, Hualan Liu, Jennifer V Kuehl, Ryan A Melnyk, Jacob S Lamson, Yumi Suh, et al. Mutant phenotypes for thousands of bacterial genes of unknown function. Nature, page 1, 2018

[11] Noah M Daniels, Andrew Gallant, Jian Peng, Lenore J Cowen, Michael Baym, and Bonnie Berger. Compressive genomics for protein databases. Bioinformatics, 29(13): i283–i290, 2013.

[12] Elnaggar, A., Heinzinger, M., Dallago, C., Rihawi, G., Wang, Y., Jones, L., ... & Rost, B. (2020). ProtTrans: towards cracking the language of Life's code through self-supervised deep learning and high performance computing. arXiv preprint arXiv:2007.06225.

[13] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.

[14] Karita, S., Chen, N., Hayashi, T., Hori, T., Inaguma, H., Jiang, Z., ... & Zhang, W. (2019, December). A comparative study on transformer vs rnn in speech applications. In 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU) (pp. 449-456). IEEE.

[15] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[16] Marcelino, P. (2018). Transfer learning from pre-trained models. *Towards Data Science*, *10*, 23.

[17] B. E. Suzek, Y. Wang et al., “UniRef clusters: A comprehensive and scalable alternative for improving sequence similarity searches,” Bioinformatics, vol. 31, no. 6, pp. 926–932, Mar. 2015.

[18] Vig, J., Madani, A., Varshney, L. R., Xiong, C., Socher, R., & Rajani, N. F. (2020). Bertology meets biology: Interpreting attention in protein language models. arXiv preprint arXiv:2006.15222.

[19] Madani, A., McCann, B., Naik, N., Keskar, N. S., Anand, N., Eguchi, R. R., ... & Socher, R. (2020). Progen: Language modeling for protein generation. arXiv preprint arXiv:2004.03497.

[20] Maxwell, B., & Drew, B. (2019, April). Pfam Seed Random Split: Using Deep Learning to Annotate the Protein Universe. Retrieved February 9, 2022 from https://www.kaggle.com/rtatman/r-vs-python-the-kitchen-gadget-test.﻿

[21] Bileschi, M., Belanger, D., Bryant, D., Sanderson, T., Carter, B., Sculley, D., ... & Colwell, L. (2019). Deep Learning Classifies the Protein Universe.

[22] LeCun, Y., et al. Handwritten digit recognition with a back-propagation network. in Advances in neural information processing systems. 1990.

[23] Rom, O., & Aviram, M. (2017). High‐density lipoprotein‐associated paraoxonase 1: a possible prognostic biomarker for heart failure?.