# CAFA 5 Protein Function Prediction

Name: **Emaad Ahmed**
Roll No: **210369**
Kaggle ID: **EmaadAhmed**
**Task Sheet:** Here
**PPT :** Here

Dept. : **BSBE**
email: **emaada21@iitk.ac.in**
Leaderboard: **332**
**Github:** Here

## Introduction:

The challenge at hand is to develop a model that can predict the biological function of a set of proteins by analyzing their amino-acid sequences along with other available data. The accurate assignment of protein function is crucial for understanding the underlying mechanisms of cells, tissues, and organs, which can ultimately lead to the development of new drugs and therapies for various diseases. Although many methods exist to determine protein function, the complexity, and ambiguity associated with the data make it difficult to achieve accurate predictions. Therefore, the problem at hand requires the development of an accurate and reliable deep learning model that can integrate multiple sources of data and provide a better understanding of protein function.
The Problem can be considered as an example of Multi-Label Classification because we have to take a protein as an input and output the predicted function it performs.

## Literature Review:

### Methods other than Deep Learning:

There are several different methods for protein prediction without using ML.

1. **Sequence Similarity:** Proteins of similar sequences are usually homologs and thus have a similar function. Hence proteins in a newly sequential genome are routinely annotated using the sequence of similar proteins in related genomes. However, closely related proteins do not always share the same function.
2. **Structure Similarity:** The 3D structure is generally more well conserved than the protein sequence, Structure similarity is a good indicator of similar function in two or more proteins. Many programs have been developed to screen an unknown protein structure against the protein data bank and report a similar structure.
3. **Sequence motif based:** The development of protein domain databases such as pfam allows one to find known domains within a query sequence, providing evidence for likely functions.

### Gene Ontology (GO) :

Gene Ontology (GO) is a Genome-wide function prediction approach using computational methods. GO utilizes the GO consortium, which is basically a hierarchically annotated database of proteins and their function for further predicting the functions of other proteins.
GO consortium uses unified functional attributes such as Biological process, Molecular Function, and Cellular Components to classify proteins hierarchically.

**Molecular Function:** It describes the gene's jobs or abilities
Biological Process: It describes the events or pathways
Cellular Component: It describes the locations (subcellular structures, macromolecular complexes)

**GO relationships** also further help in categorizing different proteins and eventually help in the prediction or recognition of a particular protein's function. Relationships are [i]: is a implying a subclass and [p]: Part of implying a membership.

The fact that we have a defined hierarchy and acyclic graphical relationship between proteins and their functions makes it easier to train deep learning models on this to get the desired predictions.

## BLAST (Basic Local Alignment Search Tool):

BLAST is a widely used algorithm that compares a query sequence against a database of known protein sequences to find similar sequences. It generates a similarity score based on sequence alignment and statistical significance. It uses some heuristic algorithm to give scores to rapidly search for local alignment matching rather than doing an exhaustive search.
Blast gives a non-ML naive approach for protein function prediction which is the current baseline in CAFA.
BLAST tool - https://blast.ncbi.nlm.nih.gov/Blast.cgi
Insight to Blast - https://www.nature.com/scitable/topicpage/basic-local-alignment-search-tool-blast-29096/
**Diamond** is a sequence aligner implementation through which we can apply the BLAST algorithm. It is an optimised implementation that greatly speeds up pairwise alignment of proteins and translated DNA at 100x-10,000x speed of BLAST.
Github - https://github.com/bbuchfink/diamond
A kaggle competition implementation for BLAST - https://www.kaggle.com/code/geraseva/diamond

## Embeddings:

## T5_Embeds:

T5 (Text-to-Text Transfer Transformer) is a state-of-the-art language model developed by Google Research. While T5 is primarily known for its text generation capabilities, it also features powerful text embeddings that capture rich semantic information from input text.
For generating T5 embeddings specific to CAFA5, the protein are converted into text format suitable for T5. This involves representing each amino acid as a character or token. Then preprocessed protein sequences are passed through the T5 model's encoder network to obtain the T5 embeddings. The encoder will process the protein sequences and produce contextual embeddings that capture the semantic information of the input sequences.

## ProtBert:

ProtBert is pre trained on a large corpus of protein sequences using unsupervised learning. During pre training, ProtBert learns to predict missing amino acids in a protein sequence based on the surrounding context. This process helps ProtBert capture meaningful representations that encode information about protein structure and function.
ProtBert embeddings are advantageous for protein function prediction as they are specifically trained on protein sequences and capture the unique characteristics of proteins. By leveraging ProtBert embeddings, the models can benefit from the knowledge and understanding encoded in the representations, leading to improved accuracy in predicting protein functions in the CAFA5 challenge.

## ESM2:

ESM2 (Evolutionary Scale Modeling) embeddings are specialized protein embeddings designed for protein function prediction tasks, such as the CAFA (Critical Assessment of Function Annotation) challenge. ESM2 is a deep learning model trained on a large dataset of protein sequences and their evolutionary profiles.
Before using ESM2, protein sequences are typically preprocessed by converting each amino acid into a numerical representation, such as one-hot encoding or using an amino acid embedding matrix.
Protein sequences, along with their corresponding evolutionary profiles, are input into the ESM2 model. The model uses a deep neural network architecture to process the sequences and profiles, producing embeddings that capture the underlying information and patterns in the protein data.

In summary, T5 embeddings are trained on general language tasks, ProtBert embeddings are specifically pretrained on protein sequences, and ESM2 embeddings consider both protein sequences and their evolutionary profiles.

# Experimentation:

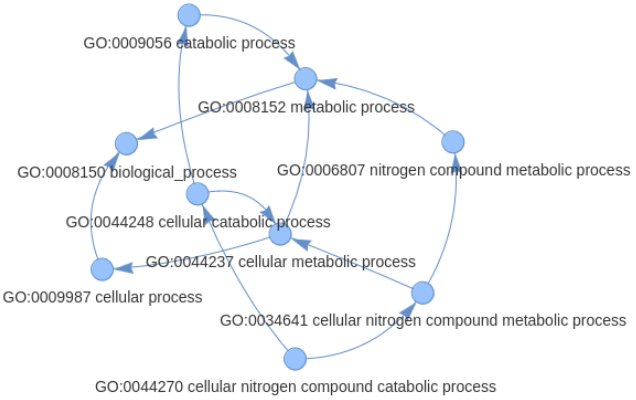
We started by doing an EDA (Exploratory Data Analysis) of the problem. Then we moved on to experimentation, starting with the most basic technique i.e **BLAST,** after that we tried using standard Machine Learning Model like **K-NN,** and eventually moved on to Deep Learning Models.
More detailed explanation of the above mentioned techniques is given below.

## EDA: Link

Before moving on to the actual experimentation part, we performed an Exploratory Data Analysis of the data provided to us by CAFA.
There are 5 files of interest given in the dataset namely go-basic.obo, train_sequences.fasta, train_terms.tsv, train_taxonomy.tsv and IA.txt.

**Go-basic.obo :** We made use of this file to represent the protein terms in Graph form to better understand their relationships.

The Graph given is for "GO:0034655"




Some other Notable features were:

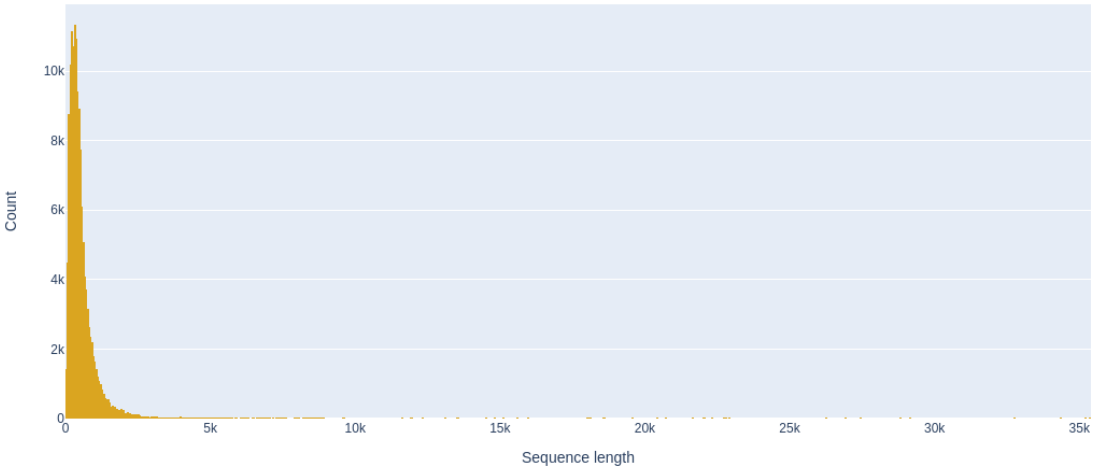
**General:**

| | |
|---|---|
| Number of Nodes | 43248 |
| Number of Edges | 84805 |
| Number of Unique Proteins | 142246 |
| Number of Unique GO Terms | 31466 |

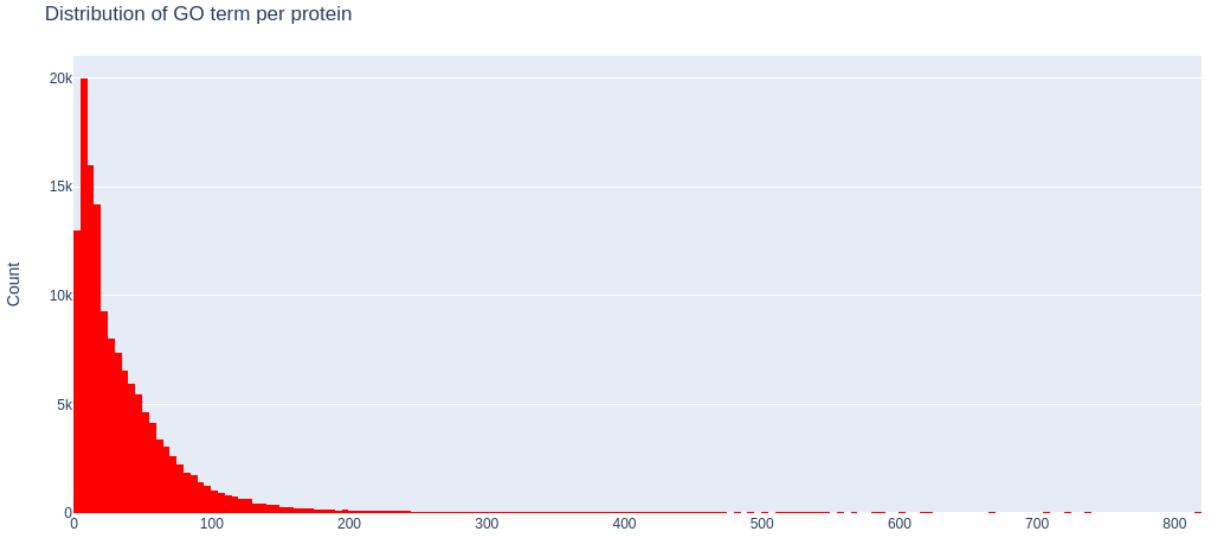
**Sequence Length:**

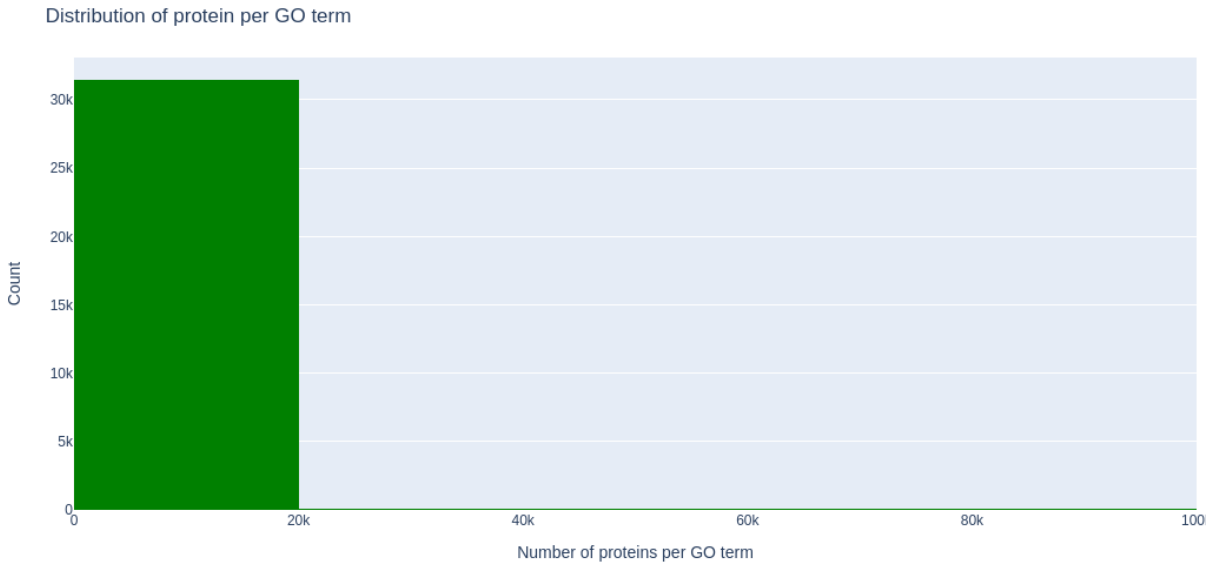
| | |
|---|---|
| Number of Sequences | 142246 |
| Average Length of Sequences | 553.64 |
| Median Length of Sequences | 411 |
| 90%ile of Length | 1054 |

**Train_sequences.fasta :** Then we used this file to get an idea about the length of protein sequences present in the dataset.




**Train_terms.tsv :** This file was used to find out the distribution of GO terms per protein and distribution of Protein per GO term



Through this EDA we were able to see the relationships between different GO terms as well as how much protein length can we safely clip or pad if we are to use transformer models. We were also able to see the dependence of GO terms on protein and vice versa.

## BLAST: Link

Due to the huge size of Sequence query it was not possible to use BLAST normally using **Biopython** package. Hence, made use of another python library called **Diamond**, which is known for handling large amount of data

For sequence matching queries, we made use of sequences provided by **testsuperset.fasta** in CAFA5 PFP dataset.

After performing sequence matching with BLAST, we store the matches in a pandas dataframe.

| | qseqid | sseqid | pident | length | mismatch | gapopen | qstart | qend | sstart | send | evalue | bitscore |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Q9CQV8 | Q9CQV8 | 100.0 | 246 | 0 | 0 | 1 | 246 | 1 | 246 | 1.100000e-167 | 464.0 |
| 1 | Q9CQV8 | P35213 | 98.8 | 246 | 3 | 0 | 1 | 246 | 1 | 246 | 1.050000e-165 | 459.0 |
| 2 | Q9CQV8 | P31946 | 98.8 | 246 | 3 | 0 | 1 | 246 | 1 | 246 | 2.120000e-165 | 458.0 |
| 3 | Q9CQV8 | V9HWD6 | 98.8 | 246 | 3 | 0 | 1 | 246 | 1 | 246 | 2.120000e-165 | 458.0 |
| 4 | Q9CQV8 | Q5PRD0 | 91.0 | 244 | 22 | 0 | 3 | 246 | 1 | 244 | 1.200000e-150 | 421.0 |
| 5 | Q9CQV8 | P63104 | 87.2 | 242 | 31 | 0 | 3 | 244 | 1 | 242 | 1.040000e-142 | 401.0 |
| 6 | Q9CQV8 | Q5ZKC9 | 86.8 | 242 | 32 | 0 | 3 | 244 | 1 | 242 | 2.970000e-142 | 400.0 |
| 7 | Q9CQV8 | P63101 | 86.8 | 242 | 32 | 0 | 3 | 244 | 1 | 242 | 5.990000e-142 | 399.0 |
| 8 | Q9CQV8 | P63102 | 86.8 | 242 | 32 | 0 | 3 | 244 | 1 | 242 | 5.990000e-142 | 399.0 |
| 9 | Q9CQV8 | P68254 | 81.0 | 242 | 46 | 0 | 3 | 244 | 1 | 242 | 4.080000e-133 | 377.0 |
| 10 | Q9CQV8 | P68255 | 81.0 | 242 | 46 | 0 | 3 | 244 | 1 | 242 | 4.080000e-133 | 377.0 |
| 11 | Q9CQV8 | P27348 | 81.0 | 242 | 46 | 0 | 3 | 244 | 1 | 242 | 4.080000e-133 | 377.0 |
| 12 | Q9CQV8 | P29310 | 78.2 | 243 | 53 | 0 | 2 | 244 | 3 | 245 | 6.820000e-128 | 363.0 |
| 13 | Q9CQV8 | Q20655 | 78.4 | 245 | 53 | 0 | 1 | 245 | 1 | 245 | 3.930000e-127 | 362.0 |
| 14 | Q9CQV8 | P41932 | 81.5 | 227 | 41 | 1 | 7 | 233 | 7 | 232 | 2.920000e-122 | 349.0 |
| 15 | Q9CQV8 | P61982 | 75.8 | 248 | 54 | 3 | 3 | 246 | 2 | 247 | 5.400000e-120 | 343.0 |
| 16 | P62259 | P62258 | 100.0 | 255 | 0 | 0 | 1 | 255 | 1 | 255 | 1.340000e-177 | 490.0 |
| 17 | P62259 | P62261 | 100.0 | 255 | 0 | 0 | 1 | 255 | 1 | 255 | 1.340000e-177 | 490.0 |
| 18 | P62259 | P62259 | 100.0 | 255 | 0 | 0 | 1 | 255 | 1 | 255 | 1.340000e-177 | 490.0 |
| 19 | P62259 | P62260 | 100.0 | 255 | 0 | 0 | 1 | 255 | 1 | 255 | 1.340000e-177 | 490.0 |

For finding the probability of the new predictions, We have considered matching with all the terms as multiple GO terms can be assosciated with the same function

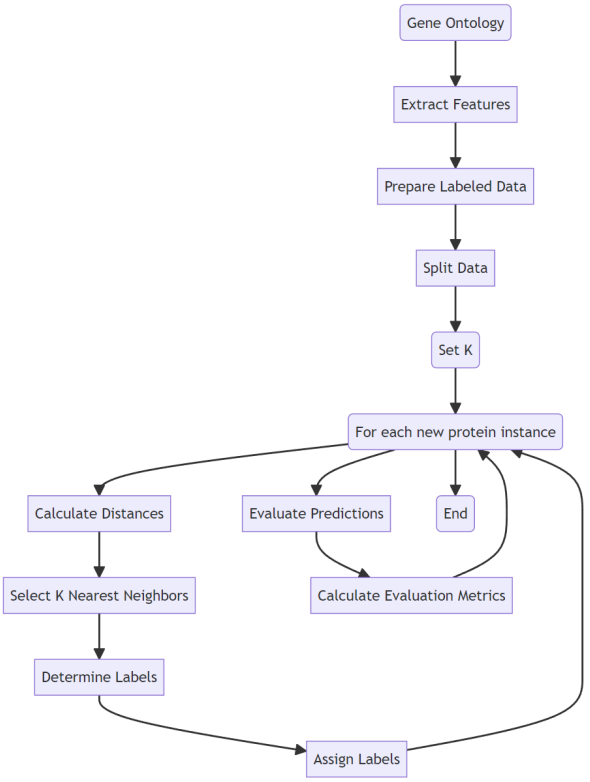
| | qseqid | terms | index | ntargets |
|---|---|---|---|---|
| 0 | A0A023PXF5 | GO:0000722 | 5019000 | 0.7 |
| 1 | A0A023PXF5 | GO:0000723 | 5019000 | 0.7 |
| 2 | A0A023PXF5 | GO:0003674 | 3584991 | 0.5 |
| 3 | A0A023PXF5 | GO:0003678 | 3584991 | 0.5 |
| 4 | A0A023PXF5 | GO:0003824 | 3584991 | 0.5 |

## KNN:

K-Nearest Neighbors (KNN) is a non-parametric machine learning algorithm used for classification and regression tasks. It predicts the label or value of a new data point by considering the K nearest neighbours from the training set. The algorithm calculates the distances between the new data point and all the data points in the training set, selects the K nearest neighbours based on the distance metric, and assigns the new data point to the most common class or averages the target values of its neighbours.

**For Multi-Label Classification,**

1. We first extracted the relevant features from the protein sequences.
2. Organised the data into feature vectors with corresponding labels.
3. We applied the KNN algorithm to train the model. During training, the KNN algorithm stores the feature vectors and labels from the training set.
4. We finally got the predictions from this model. When making predictions for a new protein, it calculates the distances between the new protein's features and the training set. The K nearest neighbours are selected, and their labels are used to assign protein function annotations to the new protein.

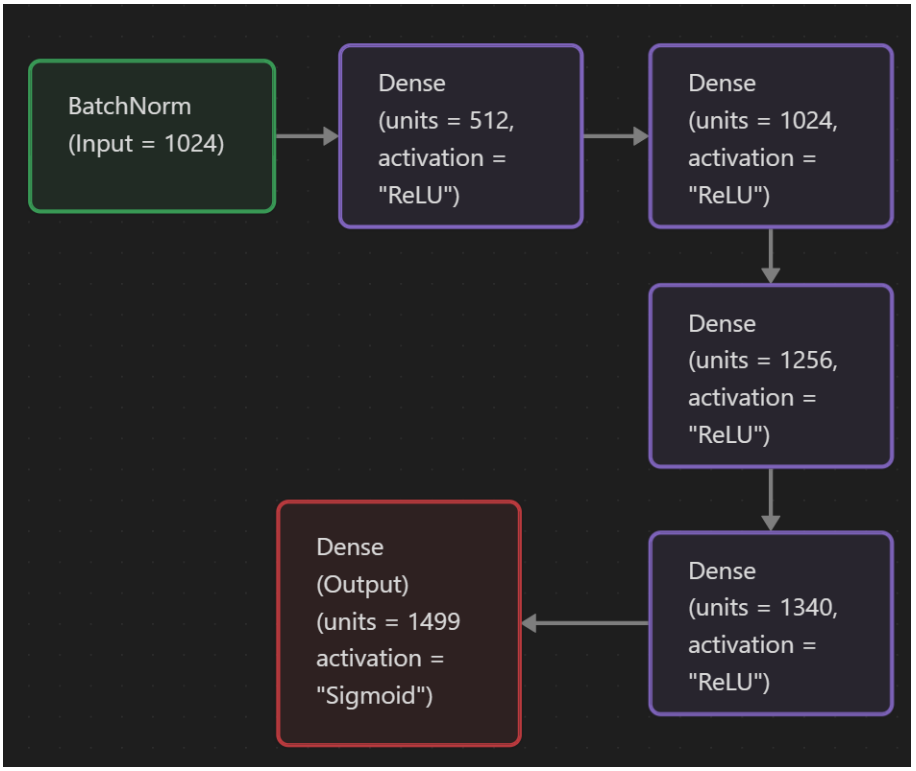


## Dimensional Reduction:

We tried Autoencoder method for dimensional reduction and then used the generated labels with KNN model

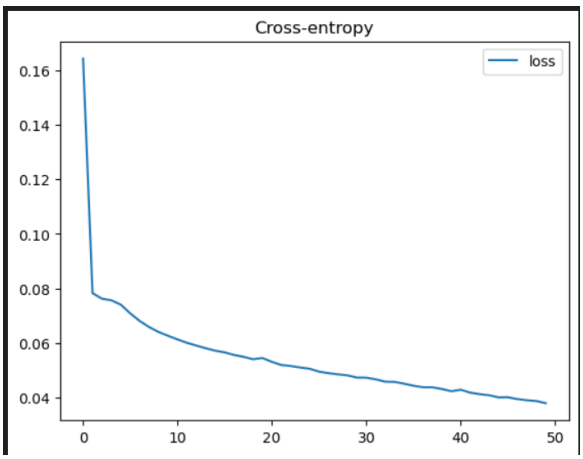
## MLP + T5 : Link

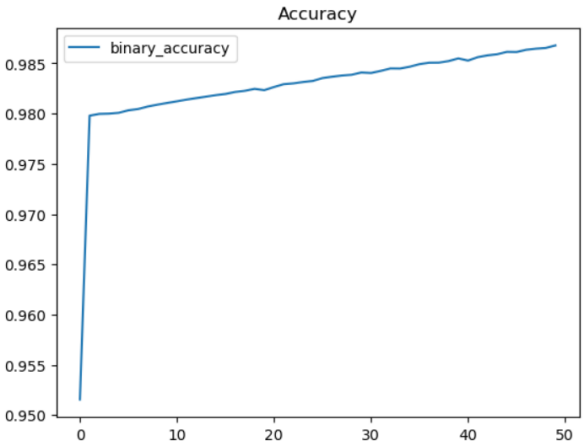
After experimenting with BLAST and standard ML models, we started Deep Learning by implementing a **Multi Layer Perceptron with T5** embeddings.




We used **Adam** optimizer along with **binary_crossentropy** loss as we are dealing with multilabel classification.

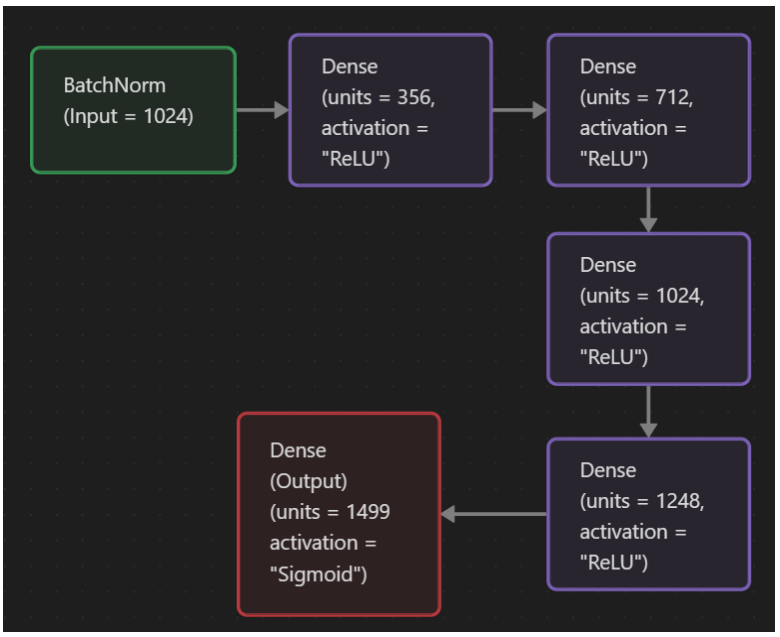
We also used **AUC** score as well as **F1-score** to evaluate the models performance on a test set made from splitting train data into train and validation.



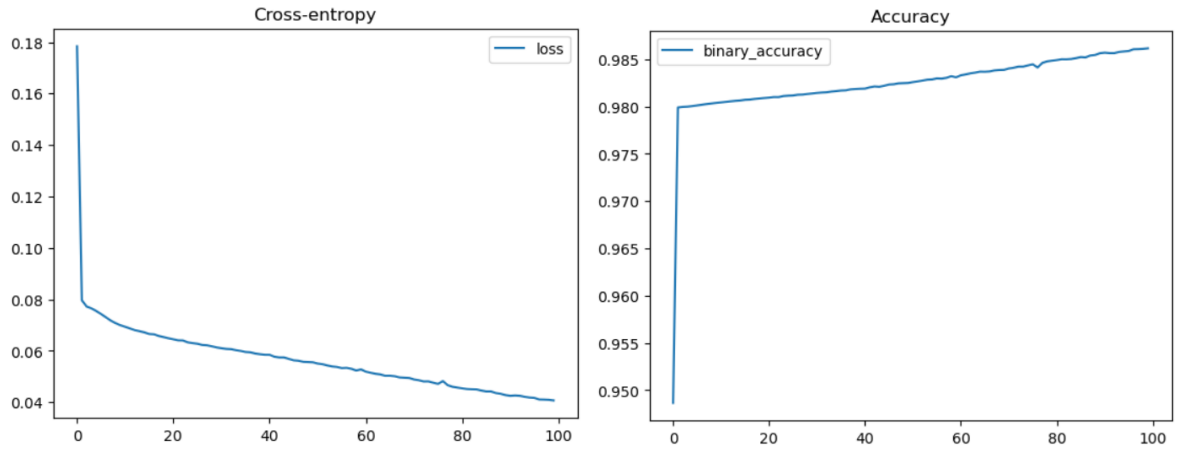




After doing hyperparameter tuning and submitting to the competition for 1 lakh labels we got an score of **0.40** and a hemming loss of **0.018**.

## MLP + ProtBert :



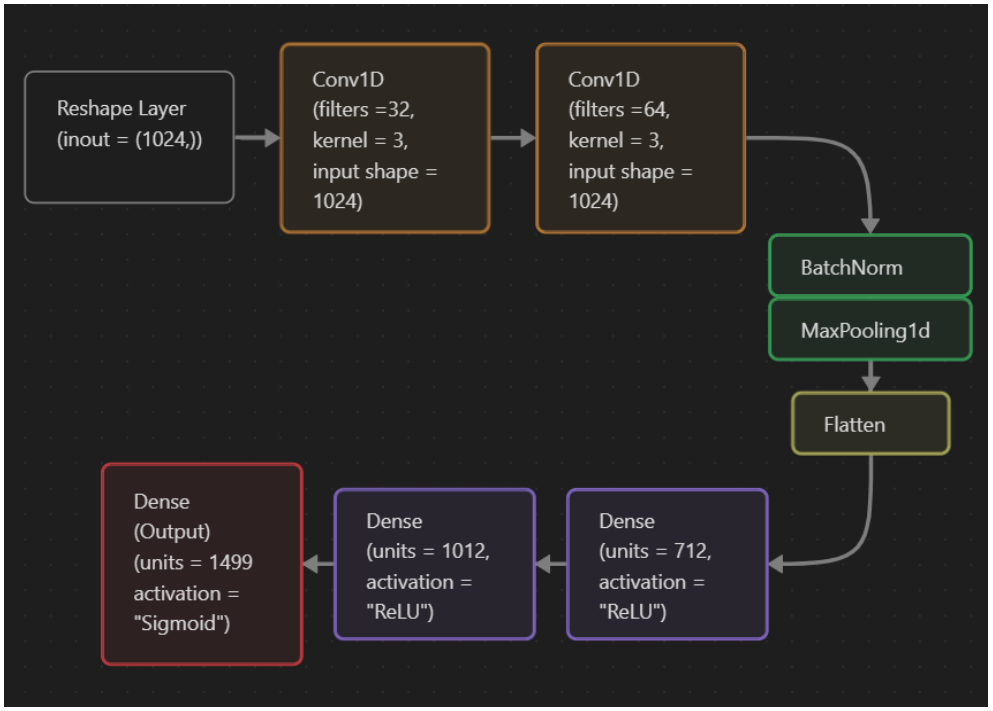

We used **Adam** optimizer and **Binary crossentropy loss** with a learning rate of 0.001.




We were able to get a F1 score of **0.36 with** hemming loss of **0.019** on **validation** dataset.

## MLP + Conv1D with ProtBert: [Link]

Model Architecture:

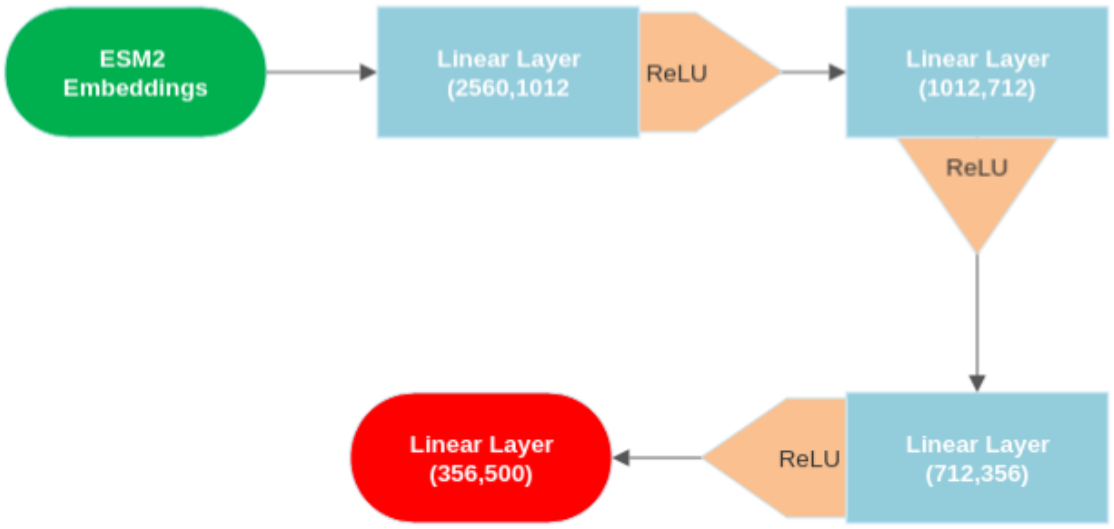


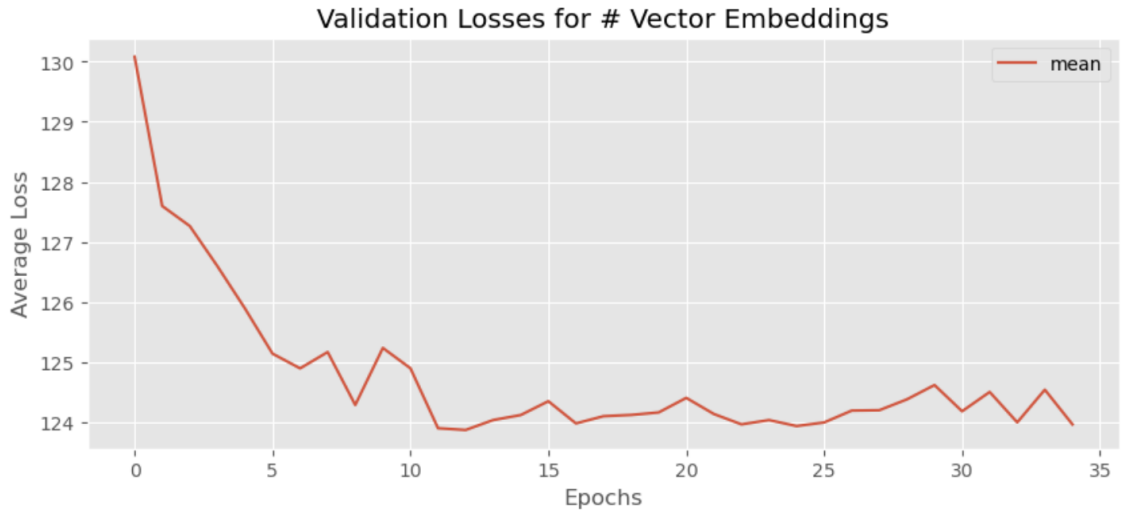

We used **Adam** optimizer and **Binary crossentropy loss** with a learning rate of 0.0015.

We were able to get a **F1 score** of 0.37 and Hemming loss of **0.02** on **validation** dataset.

## MLP with ESM2 (Transformer Model) : Link

Since, ProtBERT and T5 embeddings were not giving satisfactory F1 scores, we tried using the ESM2 embeddings and fine-tuned only the top 500 labels using a MLP model.




We used the Adam optimizer and Cross Entropy loss with a learning rate of 0.001 to get the best F1 score of 0.50 and an average loss of 123.96 on the validation dataset.

## Conclusion:

Table below summarises the F1 score obtained with different Model and model + embeddings combinations:

| Model | F1-Score |
|---|---|
| KNN | 0.059 |
| MLP + T5 | 0.40 |
| MLP + ProtBert | 0.36 |
| MLP+Conv1D with ProtBert | 0.37 |
| MLP with ESM2 | 0.503 |
| MLP + Conv1d with ESM2 | 0.501 |

## References:


- https://link.springer.com/article/10.1186/1471-2105-14-S3-S14
- https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-14-S3-S8