PROTENF : PROtein To Name and Function

1 Introduction

The process of genome sequencing, which involves deciphering the DNA code that encodes proteins in living organisms is one of the most fundamental problems in biology [17]. An interesting and useful question that arises is: given an amino acid sequence of a protein, can we provide a short description of its function or provide an informative name that contains useful information for biologists? This task can be formulated as taking a "foreign language" text (an amino acid sequence represented as text, consisting of ≈ 20 letters) and outputting readable English text containing a description of the protein's function. This task is known as Gene Annotation [2] and has crucial importance in biological research [3]. It is essential for a wide range of applications such as drug discovery (by identifying genes and pathways involved in disease processes, for example in cancer treatment [13]) or genome editing (for example by identifying genes accountable for crop yield and disease resistance, enabling the development of improved crop varieties [10], hugely important for CRISPR [1, 12]). Overall, gene annotation provides the foundation for understanding the function [2] and evolution of genomes and facilitates advancements in human health and agricultural productivity. The abundance of data from extensive protein databases, which are labelled using tools and verified through experiments, makes gene annotation an ideal candidate for deep learning models. Furthermore, the difficulty of the problem underscores the need for such tools, as stated in [4], where it was reported that state-of-the-art techniques cannot annotate 1/3 of the proteins. Researchers have made significant efforts in developing networks for biological applications, with successes such as AlphaFold [8]. Recent publications [4, 18] have demonstrated that deep learning achieves state-of-the-art performance in gene annotation, presenting an excellent new tool for biologists.

In this work, we began by using classification models such as convolution-based networks, LSTM models and Transformers to predict the function of the protein. Following an evaluation of these models, we focused on masked language modelling using current state-of-the-art models (BERT [5]), which allows for much easier transfer learning of protein related tasks. Finally, we attempted to replicate the work of the ProtNLM [18] study by formulating a sequence-to-sequence task for predicting the protein name given its amino acid representation.

2 Related Work

The current state-of-the-art solution for protein name prediction is ProtNLM [18]. The model is based on the T5 [14] text-to-text transformer architecture and is shown to be very accurate. Furthermore, ProtNLM [18] reported that the name outputted by their model is often preferred by life scientists in comparison to other methods. The model achieves such impressive results, that the authors of Uniprot embeded it into their database. Models achieving great performance for protein family classification tasks include simple bidirectional LSTM [4], ProtCNN [4] and ProtENN [4]. ProtCNN uses a residual CNN network architecture with a final classification layer for predicting the function class. ProtENN is an ensemble model of several ProtCNN to improve performance. ProtTNN [6] consists of a pre-trained protein BERT model and improves the performance of ProtENN. ProtTNN and ProtNLM are our main inspiration for using state-of-the-art NLP models to provide the textual description of what the protein does.

3 Methodology

3.1 Data

In our work, we utilised two specific databases - PFAM [11] and UniProt [18]. We were concerned with working with two types of data: sequence input and classification of that sequence mapping to function, and sequence input and biological name of that sequence (which provides some insight into what it does).

3.1.1 PFAM

PFAM [11] is a curated list of millions of proteins with associated functionality, which is encoded as one of $\approx$ 18000 classes. Each of these classes provides a short description of what is the function of a given sequence. The exact dataset we used, was provided by authors [4, 15], which contains 1 million sequences and $\approx$ 18000 classes. The dataset [15] provides ready-to-use train/val/test split and there were 2 main concerns about preprocessing: limiting the size of the input sequence (we restricted all sequences to the first 512 characters/aminoacids) and data imbalance (the classes follow a quite long tail distribution). In our investigations, For classification, we initially restricted the data to 3000 most common proteins ($\approx 66\%$ data), however, to make experiments faster we switched to 1000 most common proteins ($\approx 40.5\%$ data). For the task of masked language modelling, we used 50 classes to compensate for much longer training times (classification models achieved good results after a few epochs, and MLM was trained on 100 epochs).

3.1.2 UniProt

UniProt [18] is an extensive list of more than 200 million sequences with much more detailed information (including human-readable protein names) about their functionality compared to PFAM. We used their web REST API to collect the data. Initially, we collected all reviewed by professionals sequences of proteins with their names from human organism ($\approx$ 20 thousand entries). This small dataset was initially used in predicting the name of a protein task, however, we noticed a very low score on "predicted all tokens correctly". We hypothesised that this is because of the poor performance of the tokenizer (the inability to extract good enough tokens from the names). We collected a dataset of ≥ 1 mln pairs of sequences and names, whose annotation score (a value from 1 to 5, where 1 is associated with basic annotation and 5 is the best-annotated entry) is higher or equal to 4. This dataset was used to train the tokenizer, which increased the performance.

3.2 Predicting the protein function

Even though it is a classification task, the PFAM family provides a textual information about what exactly the protein does. This can range from a single sentence to a few sentences describing what does this sequence do. This served as a starting point in tackling later more difficult problems, but still providing useful [4] text output.

We trained multiple architectures on the task of predicting one of the 1000 most popular protein classes. More specifically, the models we trained were: a simple bidirectional LSTM network, transformer [19], protCNN [4] and BERT. Each of these models, however, was significantly rescaled, to make the training time faster on much smaller hardware. For all models, the input was 512 tokens representing amino acid sequence (padding with 0s was applied when the sequence was less than 512), and the output head of 1000 classes.

As the rest of the models are fairly standard, only a brief discussion of protCNN is provided. This model achieved exceptional performance on the PFAM ID task as reported by researchers [4]. Firstly, the network one-hot encodes the amino acids, and after that, it moves it through multiple convolutions. In particular, it uses a few residual blocks (almost the same as a residual block from Resnet [7], but uses 1D convolutions). After these convolutions (and a few tricks to permute the data), the network creates a 2D-like image, through which we slide 2D convolutions, and pass it to the classification head. The implementation of that network was not provided by paper, therefore we experimented with this implementation and some tries from kaggle notebooks [9]. The network is quite long therefore the details of the architecture can be found in the implementation submission.

3.3 Self-supervised training: Masked Language Modelling for proteins

One advantage of current NLP models is that they can be fairly easily fine-tuned. Typically, a model such as BERT is trained on a language modelling task (for example, given a few previous words, predict the next one) or masked language modelling (replace some of the words in the sentence with a masked token and try to reconstruct the sentence).

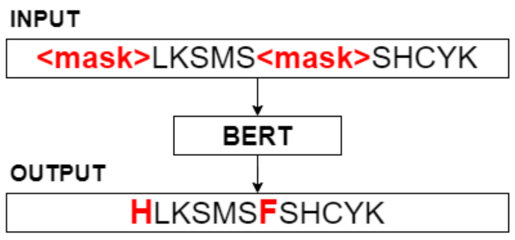

Figure 1: Figure illustrating the Masked Language Modelling

This self-supervised training should allow for this pre-trained model to be adapted to other tasks. We tested that on the BERT model: we trained it for 100 epochs on a small subset of PFAM family ids (50 classes, $\approx$ 50k proteins). The input to this model is the same as in classification (512 tokens, with padding applied when the sequence is less than 512 characters), and the output is 512 tokens. The input sequence is perturbed: each token has a small probability of being masked (we tried 1%, 5%, 10% and 15%), and the output is an unmasked input sequence. This task can be seen in figure 1.

3.4 Protein name function prediction

The final task was a sequence-to-sequence task, of mapping an amino acid sequence to a name. This was achieved by retraining the T5 model (t5-small [16]) on a small subset of proteins collected from UniProt (manually reviewed proteins from human organism). T5 [14] is a popular encoder-decoder model which achieved great performance on many tasks. Our main concern is that although this model can handle a variety of tasks and languages, in its initial state it is probably not well suited to converting amino acids to very specific, scientific vocabulary of protein names. We replaced, therefore, this tokenizer with our own, which was trained on a much larger collection of protein names. The main concern, of course, was how to tokenize difficult names of the proteins. Due to time limitations, we did not explore much of that and focused on very simple tokenization (Hugging Face SentencePiece) which will automatically split these tokens into subwords. This is not great however further exploration of better tokenization is a goal for the future. The input to the T5 is 512 tokens (max, 510 aminoacid characters, the model also encodes explicitly the start of the sentence and end of the sentence), and outputs 32 tokens representing the name (we got that number by using the tokenizer

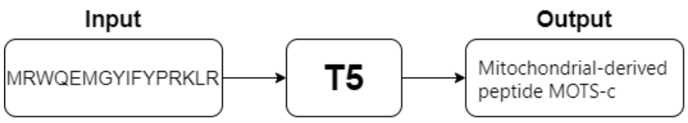

Figure 2: Figure illustrating the sequence-to-sequence task of protein sequence to protein name

for all names, and this value was closest power of 2 that bounded the size of all tokenized names). The overall diagram of the task is in figure 2.

4 Evaluation and Discussion

4.1 Classification

The table 1 contains information on all classification model results that we collected. As expected

model name	accuracy	parameter count
ProtCNN (kernel size=3)	83.66%	393k
ProtCNN (kernel size=5)	88.24%	459k
Bidirectional LSTM	98.02%	192k
pure transformer	93.87 %	213k
BERT	96.16%	3.67m

Table 1: Results for classification models

from publication [4], the bidirectional LSTM performs very well. However, it does not scale well: we believe, for 10000 classes, the reported architecture would have 37.5 mln parameters (while BERT-small would have 4.83mln). We were not able to reproduce the great performance of protCNN (it was reported that the model outperforms the LSTM on a much larger dataset). We attribute to the main factor that we do not know exactly whether this code is the same as used by authors [4]. We however noticed that on smaller subset (250 and 500 classes), these models (protCNN and LSTM) achieve extremely similar, almost 100% accuracy on the test dataset. The ProtCNN achieves that score also much faster than LSTM. We believe that it would be possible to reach $\geq 95\%$ accuracy with protCNN on a larger dataset, however, we would need to scale the models differently. Based on reported results, we believe also that the BERT model could be further fine-tuned to achieve much better, closer to LSTM performance.

4.2 Masked Language Modelling for proteins

The training of Masked Language Modelling models was much more computationally expensive (it seems to converge after 100-150 epochs). We

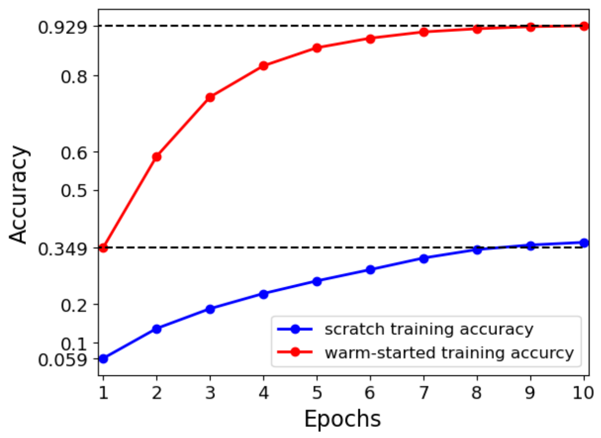

Figure 3: The benefits of warm starting. Two models with the same architecture (BERT with 8 layers and hidden size = 512) and number of parameters were trained for 10 epochs. One was warm-started (weights taken from pretraining) and one was trained from scratch. Both models were trained with the same parameters, optimizer AdamW and learning rate=1e-4

trained two models: BERT-tiny with an accuracy of 46.5% and a version of BERT with 8 layers and hidden sizes of 512 with an accuracy of 56.6%. Our MLM procedure consisted only of randomly masking an input token. A more difficult procedure was described here [6] and achieved an accuracy of $19.33\% - 26.46\%$. Although these scores might not seem impressive, they provide a tremendous benefit for training, which can be seen in figure 3. It is worth noticing, that both models can achieve very high, competitive performance of over 95%. However, the model from scratch requires much more epochs, with more aggressive training.

4.3 Protein Name prediction

The model used for protein name prediction was T5 [14]. The main difficulty of this task is due to the fact, that checking whether two names represent the same information requires extensive biological knowledge. We used two metrics: "tokens matching" which is the accuracy of all non-padding tokens predicted (for example for prediction [1, 2, 3, 4] and truth [1, 2, 3, 5], the score would be 0.75) correctly and "exact match" where all tokens need to be the same (the previous prediction/truth it would be 0).
Initially, it was trained on a human protein dataset ($\approx 9k$ points for training, $\approx 2k$ points for evaluation). The model, however, performed very poorly on this dataset, with 55% "tokens matching" and $.156\%$ of exact matches. We were not able to explain the terrible exact match performance, however, we believe that it is due to the small dataset (huge overfitting) and the way that the tokenizer was trained. Further investigation of this is a plan for the future. We moved on to a larger dataset (all proteins from UniProt with annotation score = 5). The results there were much more promising: 83.2% on "tokens matching" and $\approx 60\%$ of exact matches. We believe that, given the complexity of the problem and the smaller size of the model, this is a very good score. The reported exact match for protNLM was 57% [18]. A thing to point out is that protNLM was trained for tens of millions of proteins (with very different annotation scores) and we used a small dataset of 100k of what is described as "best-annotated entries in UniProt".

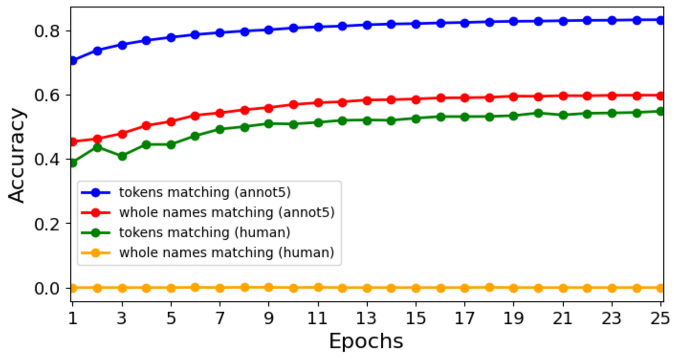

Figure 4: Two metrics ("tokens matching": percentage of tokens matching and "whole names matching": percentage of predicted points that match perfectly) on two datasets (human proteins dataset and annotation score 5 datasets)

5 Conclusion

We believe that obtained results are good enough. The models we trained achieved very strong performance on classification, decent score on MLM (however obtained results are very good when warm-starting model) and good for protein name. We were not able to reproduce very strong protCNN performance. As mentioned in the proposal, there is difficulty in evaluating properly the protein name prediction. However, we believe that given enough time and computational resources, it would be possible to rescale the developed to larger datasets and models, that match closely reported results from publication.

References

[1] *A Programmable Dual-RNA–Guided DNA Endonuclease in Adaptive Bacterial Immunity*. Science, 2023.

URL: https://www.science.org/doi/full/10.1126/science.1225829 (visited on 05/05/2023).

[2] Josep F. Abril and Sergi Castellano. “Genome Annotation”. In: *Encyclopedia of Bioinformatics and Computational Biology* (2019), pp. 195–209. DOI: 10.1016/b978-0-12-809633-8.20226-4. URL: https://www.sciencedirect.com/science/article/pii/B9780128096338202264 (visited on 05/04/2023).

[3] Jennifer L. Ashurst and John E. Collins. “Gene Annotation: Prediction and Testing”. In: *Annual Review of Genomics and Human Genetics* 4 (Sept. 2003), pp. 69–88. DOI: 10.1146/annurev.genom.4.070802.110300. URL: https://pubmed.ncbi.nlm.nih.gov/14527297/ (visited on 05/05/2023).

[4] Maxwell L Bileschi et al. “Using deep learning to annotate the protein universe”. In: *Nature Biotechnology* 40 (Feb. 2022), pp. 932–937. DOI: 10.1038/s41587-021-01179-w. URL: https://www.nature.com/articles/s41587-021-01179-w (visited on 05/01/2023).

[5] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. May 2019. URL: https://arxiv.org/pdf/1810.04805.pdf.

[6] David Dohan et al. “Improving Protein Function Annotation via Unsupervised Pre-training: Robustness, Efficiency, and Insights”. In: *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery Data Mining* (Aug. 2021). DOI: 10.1145/3447548.3467163.

[7] Kaiming He et al. *Deep Residual Learning for Image Recognition*. arXiv.org, 2015. URL: https://arxiv.org/abs/1512.03385 (visited on 05/05/2023).

[8] John Jumper et al. “Highly accurate protein structure prediction with AlphaFold”. In: *Nature* 596 (July 2021), pp. 583–589. DOI: 10.1038/s41586-021-03819-2. URL: https://www.nature.com/articles/s41586-021-03819-2 (visited on 05/05/2023).

[9] kaledhoshme. *Deep Learning to Annotate the Protein Universe*. Kaggle.com, Nov. 2022. URL: https://www.kaggle.com/code/kaledhoshme/deep-learning-to-annotate-the-protein-universe (visited on 05/04/2023).

[10] Colin Kern et al. “Functional annotations of three domestic animal genomes provide vital resources for comparative and agricultural research”. In: *Nature Communications* 12 (Mar. 2021). DOI: 10.1038/s41467-021-22100-8. URL: https://www.nature.com/articles/s41467-021-22100-8 (visited on 05/04/2023).

[11] Jaina Mistry et al. “Pfam: The protein families database in 2021”. In: *Nucleic Acids Research* 49 (Oct. 2020), pp. D412–D419. DOI: 10.1093/nar/gkaa913. URL: https://academic.oup.com/nar/article/49/D1/D412/5943818?guestAccessKey=18212916-1e97-4c18-8f4d-b0ba26beeaa8 (visited on 05/04/2023).

[12] *Multiplex Genome Engineering Using CRISPR/Cas Systems*. Science, 2013. URL: https://www.science.org/doi/10.1126/science.1231143 (visited on 05/04/2023).

[13] Patrick Kwok-Shing Ng et al. “Systematic Functional Annotation of Somatic Mutations in Cancer”. In: *Cancer Cell* 33 (Mar. 2018), 450–462.e10. DOI: 10.1016/j.ccell.2018.01.021. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5926201/ (visited on 05/04/2023).

[14] Colin Raffel et al. “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer”. In: *Journal of Machine Learning Research* 21 (2020), pp. 1–67. URL: https://jmlr.org/papers/volume21/20-074/20-074.pdf.

[15] Google Research. *Pfam seed random split*. Kaggle.com, 2019. URL: https://www.kaggle.com/datasets/googleai/pfam-seed-random-split (visited on 05/04/2023).

[16] *t5-small · Hugging Face*. Huggingface.co, Nov. 2022. URL: https://huggingface.co/t5-small (visited on 05/04/2023).

[17] *The complete sequence of a human genome*. Science, 2022. URL: https://www.science.org/doi/10.1126/science.abj6987 (visited on 05/05/2023).

[18] *UniProt*. Uniprot.org, 2023. URL: https://www.uniprot.org/help/ProtNLM (visited on 05/01/2023).

[19] Ashish Vaswani et al. *Attention Is All You Need*. arXiv.org, 2017. URL: https://arxiv.org/abs/1706.03762 (visited on 05/05/2023).