Learning co-evolution information with natural language processing for protein folding problem*

Egor Zverev[1]***, Sergei Grudinin***[2]***, and Ilia Igashov***[1,2]

zverev.eo@phystech.edu; sergei.grudinin@inria.fr; igashov.is@phystech.edu

[1]Organization, address; [2]Organization, address

Co-evolution information is crucial for building protein sequence descriptors. Its computation is traditionally based on Multiple Sequence Alignment. Applications of MSA are limited, for instance, it fails for sequences with shallow alignment. This work investigates pre-trained language models as potential alternatives to MSA and focuses on fold classification problem. The authors examine several state-of-the-art methods and modify them by replacing MSA-based feature-generation parts with pre-trained LMs. The performance of the models is evaludated using TOP1-accuracy score.

Keywords: *protein fold classification; co-evolution, feature generation; transformers; BERT;*

DOI: 10.21469/22233792

1 Introduction

Protein properties are determined by its shape, which is defined by its amino acid sequence [13]. Therefore it is important to learn how to analyze this sequence. The first step is constructing protein sequence descriptors. Ideally, the descriptors contain co-evolution information [9, 11]. This information is obtained by searching for homologues of the protein in large databases and computing multiple sequence alignment (MSA) on it. This model requires significant computational effort. Alignment databases are finite. If for a given sequence there are no matches in the database, MSA-based methods fail to produce reliable results. It does not guarantee precise results since it fully relies on finite databases. MSA-based methods fail for sequences with shallow alignment.

Co-evolution-based methods assume [5] that a mutation of one amino acid leads to mutations of others. Therefore, given an amino acid sequence, it is natural to look for other similar sequences. That is performed with MSA [5]. This method fails when we are dealing with the sequences that did not evolve a lot. They have shallow alignments. That means there are only a few similar sequences in the database. Therefore, information extracted from these alignments is poor.

*

The authors propose to analyse pre-trained language models (LMs) [8] as potential alternative to traditional MSA-based models. It is assumed that protein sequences are not random [4]. There is structure in amino acid sequences. Therefore, a set of all amino acids could be seen as a language with complicated inner rules. BERT [6] learns the structure of any abstract language. We assume that by learning amino acids structure, BERT will be able to implicitly learn co-evolution information.

Our main aim is to study the efficiency of pre-trained LMs in application to the fold classification problem. To start with, we study the state-of-the-art method DeepSF [9]. It solves the protein fold classification problem. By replacing its MSA-based feature-generation part with a pre-trained LM, we study how NLP approach helps to learn fold-related information. The whole framework is schematically represented in Figure 1.

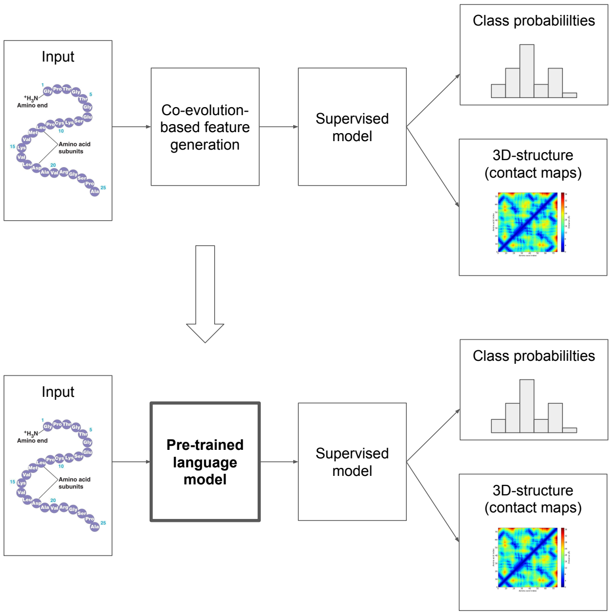

Input
+H3N
Amino end
Amino acid subunits
Co-evolution-based feature generation
Supervised model
Class probabililties
3D-structure (contact maps)
Input
Pre-trained language model
Supervised model
Class probabililties
3D-structure (contact maps)

Figure 1 The main framework

2 Problem statement

Proteins that share similarities in their structure are divided into classes – folds [12]. In this work we investigate a fold classification problem.

2.1 General description

We denote Y a set of all known classes, A a set of all proteins with known folds and $f : A \to Y$ a mapping between proteins and their classes.

Suppose, $x \notin A$ is a protein with unknown fold. We aim to build a model capable of classifying x as an element of one of the present classes, extending f on the set $A \cup \{x\}$.

2.2 Feature generation

In the fold classification problem x_i represents i-th protein description in D-dimensional space. However, initially each protein is described by its amino acid sequence. Let Σ be the 20-letter alphabet of acid sequences.

There is a set $A = \{a_i \in \Sigma^+\}_{i=1}^n$ of protein sequences, where Σ^+ is a set of all possible non-empty sequences. Let

$$f : \Sigma^+ \to \mathbb{R}^D$$

be an embedding of space of acid sequences into D-dimensional set of descriptors.

Protein sequence descriptors are generated using f as $x_i = f(a_i)$

2.3 Fold classification problem

Suppose there is a given set of pairs $S = \{(x_i, Y_i)\}_{i=1}^n$ where $x_i \in \mathbb{R}^D$ denotes the sequence descriptor of i-th protein and

$$Y_i \in \mathbb{Y} = \{(1, 0, ..., 0), (0, 1, 0, ..., 0)....(0, ..., 0, 1)\}$$

denotes the i-th protein known class represented as one-hot vector, $|\mathbb{Y}| = m$.
S is separated into training, validation and test set.

$$S = S_{\text{train}} \cup S_{\text{val}} \cup S_{\text{test}}$$

Let $W = \{(w_1, ..., w_m) | w_i \in \mathbb{R}^D\}$ be a space of parameters for classification models. Let g_w be a model parameterized by w. We use e_k to denote a vector with 1 on k-th position, with zeros on all other positions

$$e_k = \quad [0, \ldots, 0, \quad \underset{\substack{\uparrow \\ k}}{1}, \quad 0, \ldots, 0].$$

We work under the following assumption:

$$P_w(Y = e_k | x) = \frac{exp(x^\top w_k)}{\sum_j exp(x^\top w_j)}$$

Let $p_{w,x}(y)$ be discrete density of Y, $L_{Y,x}(w)$ be likelihood function of Y.

$$p_{w,x}(y) = \prod_{d=1}^{m} (P_w(Y = e_d | x))^{y_d}$$

,

$$L_K(w) = \prod_{i=1}^{m} p_{w,x_i}(Y_i) \text{ for } K = \{(x_i, Y_i)\}_{i=1}^{m}$$

Given S_{train}, classification problem is a maximization of likelihood on training set:

$$L_{S_{\text{train}}}(w) \to \max_{w \in W}$$

3 Computational experiment

The goal of our experiments is to verify that BERT is able to extract important features from the given amino acid sequences. For this we will use pre-trained AlBert [7] model to generate these features and then train our own neural networks to solve fold classification problem.

3.1 Data

We use SCOP2 [1,2] dataset. It contains information about proteins whose structure is already known. All the proteins in the database have classes assigned to them (folds). At present, there are 1517 known folds [2]. The number of proteins contained in each fold is less than 50 for most of the folds. Detailed histogram is represented in Figures 2, 3.

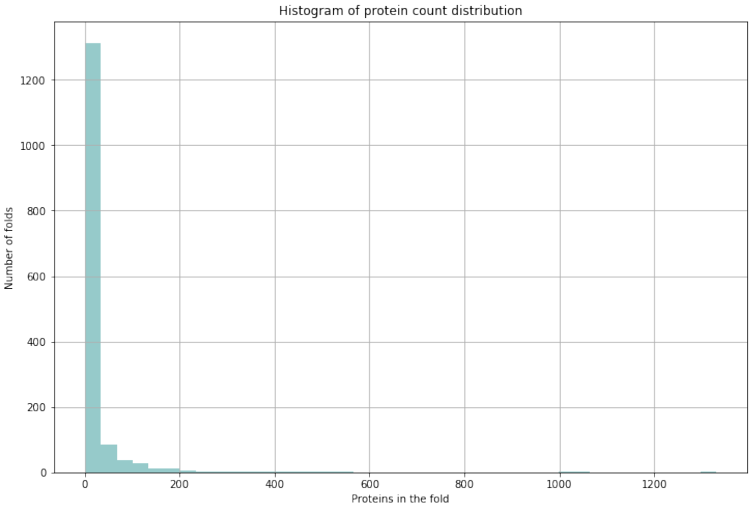


Figure 2 Protein count histogram

In SCOP2 proteins are represented by their amino acid sequences. Each sequence is stored as a string in the 20 letter alphabet and an ID. The length of the string varies mostly between 25 and 500 from protein to another. The length diagram is represented on Figure 2. Search by ID allows database users to extract information about target proteins, including their fold classes.

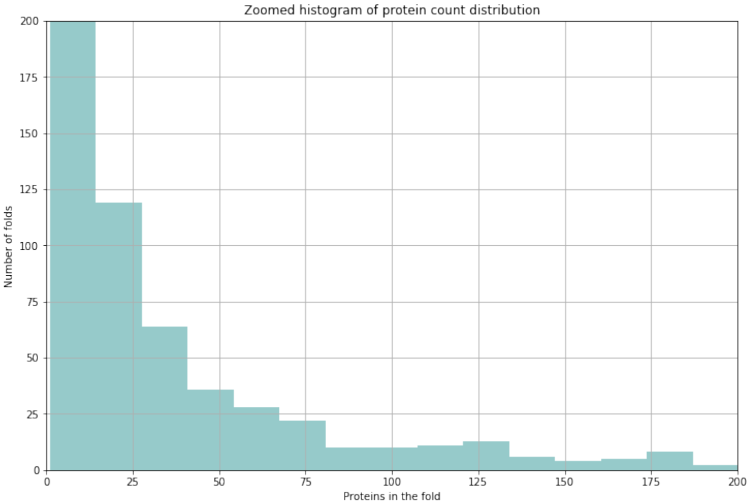


Figure 3 Protein count zoomed histogram

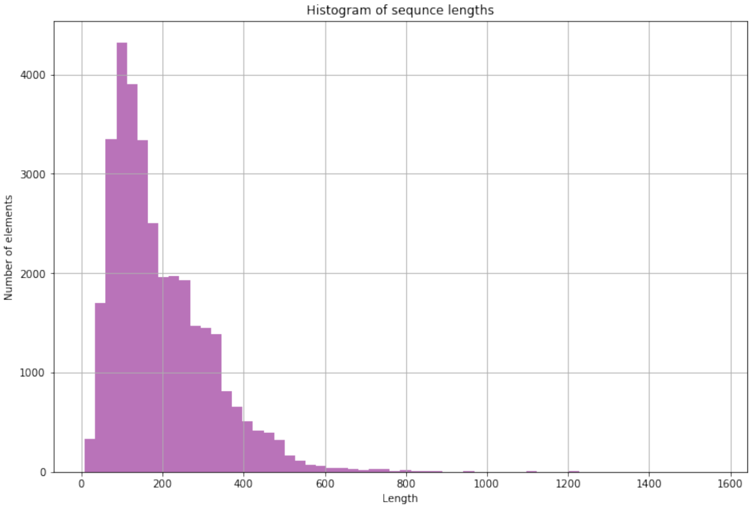


Figure 4 Sequence length histogram

3.2 Baseline solution

Our solution is a modification of the DeepSF [9] method. The designers of DeepSF use PSI-BLAST [3] to generate features from MSA as well as SCRATCH [10] to extract secondary structure (3 classes) and solvent accessibility (2 classes). In the core of this classification method PSI-BLAST is used during the first stage of prediction [10].

DeepSF authors then use these features as an input to the deep convolutional neural network. The depth of CNN used in DeepSF is 10. The output of the model is a vector of fold probabilities. Fold with the maximal probability is accepted as a target protein fold.

3.3 Proposed solution

DeepSF feature generation is entirely based on MSA. The main point of this work is to suggest another approach to feature generation which is independent of MSA.

We apply BERT to the language of amino acids. We take the representation of acid sequences BERT creates as a new set of features. Then we train a model similar to DeepSF on these features.

It shall be noted that we do not reject co-evolution information in our research. Instead of directly applying MSA we let BERT learn information about amino acid language. We suppose that the model learns evolution information implicitly.

3.4 Evaluation

We use TOP-1 accuracy score on $S_{\text{test}} == \{(x_i, Y_i)\}_{i=1}^{l}$ to evaluate the performance of the model. Let g be the prediction mode. The score as computed as

$$acc(g) = \sum_{i=0}^{l} I_{g(x_i)=Y_i}$$

4 Experiment details

Though the data is available for 1517 classes, during the experiments we approached the problem slowly. First, we solved a classification problem for 2 classes, then for 10 classes.

4.1 First experiment

In this experiment we took two balanced classes. They contain 66 and 80 elements respectively. The distributions of their sequence length are similar. They are displayed on figure 5. We used a model that consists of two essential layers - 30-max-pooling followed by a linear layer. Using this architecture we achieved 93% score on validation data. The architecture and training curves are represented on figures 6 and 7.

The architecture presented here is the simplest possible one - it contains max-pooling layer to transform input to fixed size and FC layer for classification. Motivation for this experiment is to verify whether features extracted by BERT are meaningful.

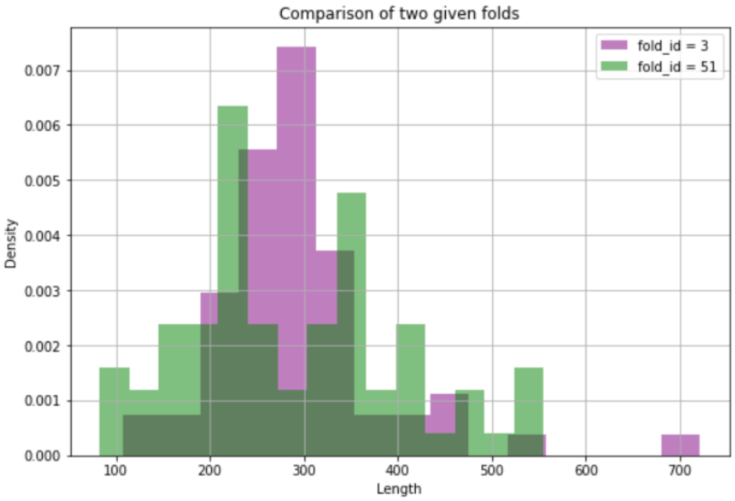
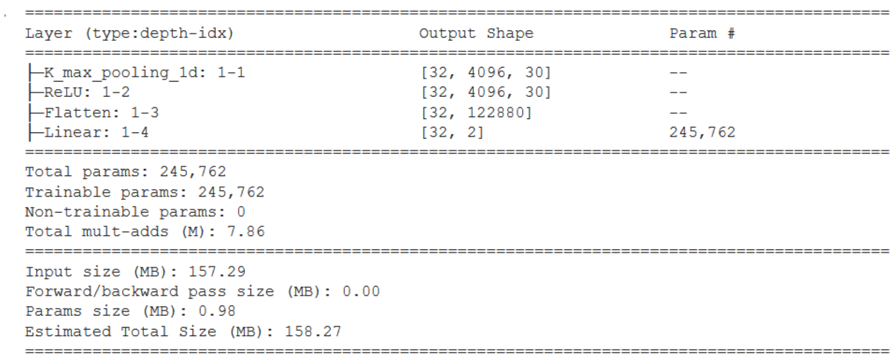


Figure 5 Sequence lengths of two folds



```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
├─K_max_pooling_1d: 1-1                  [32, 4096, 30]            --
├─ReLU: 1-2                              [32, 4096, 30]            --
├─Flatten: 1-3                           [32, 122880]              --
├─Linear: 1-4                            [32, 2]                   245,762
==========================================================================================
Total params: 245,762
Trainable params: 245,762
Non-trainable params: 0
Total mult-adds (M): 7.86
==========================================================================================
Input size (MB): 157.29
Forward/backward pass size (MB): 0.00
Params size (MB): 0.98
Estimated Total Size (MB): 158.27
==========================================================================================
```

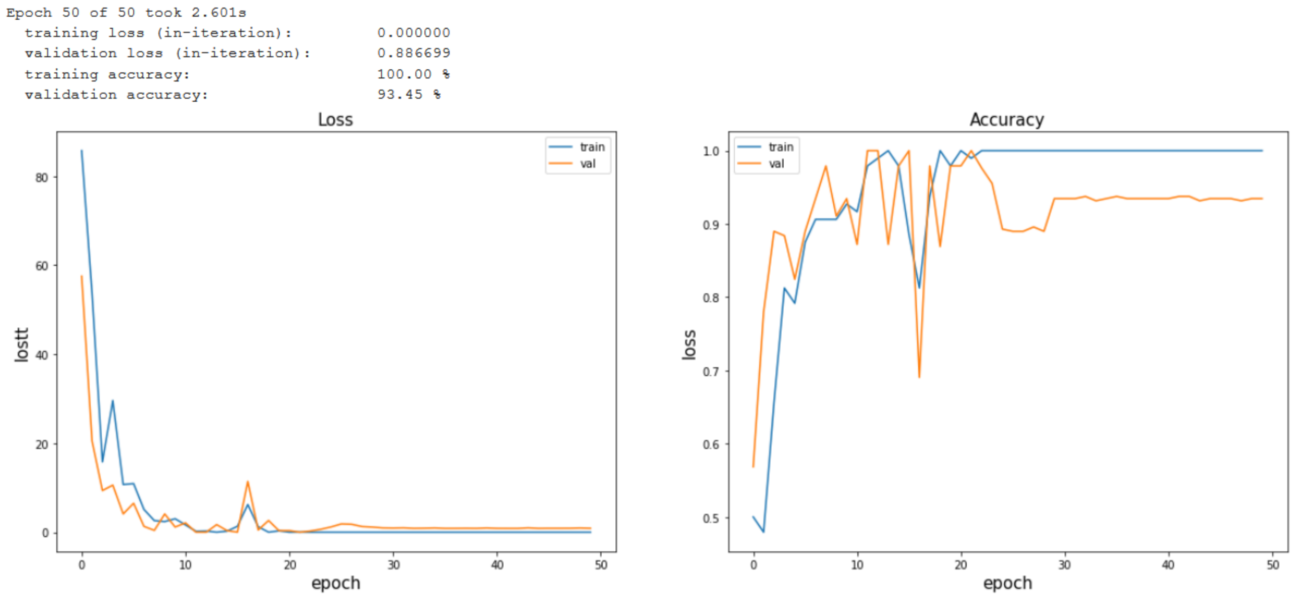
Figure 6 Experiment 1 architecture

4.2 Second experiment

We decided to use exactly the same architecture for 10-fold classification problem. Our experiments show that increasing the k hyper-parameter in k-max-pooling layer improves the quality of the model. We chose k = 60. It allowed us to reach 95% accuracy on validation and 93% accuracy on test samples. The training curve is represented on figure 8.

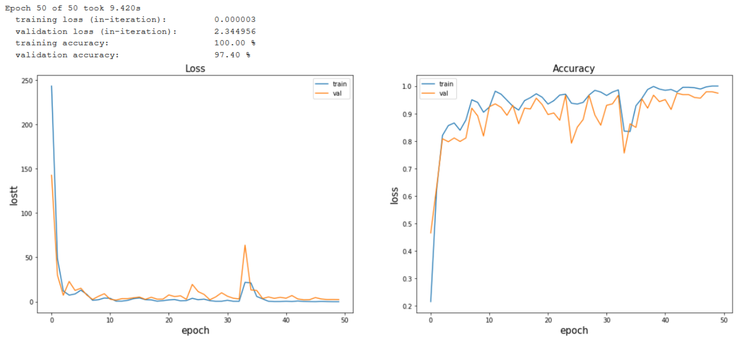
The second experiment is an extension of the first one. Its goal is to validate that simple 2-layer architecture could deal with more complicated classification problem.

5 Results



```
Epoch 50 of 50 took 2.601s
  training loss (in-iteration):          0.000000
  validation loss (in-iteration):        0.886699
  training accuracy:                     100.00 %
  validation accuracy:                   93.45 %
```

Figure 7 Experiment 1 training curve



```
Epoch 50 of 50 took 9.420s
  training loss (in-iteration):          0.000003
  validation loss (in-iteration):        2.344956
  training accuracy:                     100.00 %
  validation accuracy:                   97.40 %
```

Figure 8 Experiment 2 training curve

Conducted experiments show that BERT successfully captured the structure of the protein sequence language. We managed to solve 10-fold classification problem with 93% accuracy.

Therefore, BERT is able to extract crucial information from the sequences of amino acids.

6 Feature work

In this paper we investigated only classification problem for at most classes. In the future we would like to extend our experiments for 1517-fold classification problem and compare the results.

Here we used BERT as a "black box". We generated features with BERT and used them as an input for another model. The desired improvement here would be to learn how to leverage attention mechanisms from BERT directly, possibly gaining more co-evolution information in the process.

References

[1] Antonina Andreeva, Dave Howorth, Cyrus Chothia, Eugene Kulesha, and Alexey G. Murzin. SCOP2 prototype: a new approach to protein structure mining. *Nucleic Acids Research*, 42(D1):D310–D314, 11 2013.

[2] Antonina Andreeva, Eugene Kulesha, Julian Gough, and Alexey G Murzin. The SCOP database in 2020: expanded classification of representative family and superfamily domains of known protein structures. *Nucleic Acids Research*, 48(D1):D376–D382, 11 2019.

[3] Medha Bhagwat and L. Aravind. Psi-blast tutorial. *Methods in molecular biology (Clifton, N.J.)*, 395:177–186, 2007. 17993673[pmid].

[4] Davide De Lucrezia, Debora Slanzi, Irene Poli, Fabio Polticelli, and Giovanni Minervini. Do natural proteins differ from random sequences polypeptides? natural vs. random proteins classification using an evolutionary neural network. *PLOS ONE*, 7(5):1–10, 05 2012.

[5] S. de Oliveira and C. Deane. Co-evolution techniques are reshaping the way we do structural bioinformatics. *F1000Res*, 6:1224, 2017.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[7] Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, DEBSINDHU BHOWMIK, and Burkhard Rost. Prottrans: Towards cracking the language of life's code through self-supervised deep learning and high performance computing. *bioRxiv*, 2020.

[8] Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rihawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Debsindhu Bhowmik, and Burkhard Rost. Prot-

trans: Towards cracking the language of life's code through self-supervised deep learning and high performance computing. *bioRxiv*, 2020.

[9] Jie Hou, Badri Adhikari, and Jianlin Cheng. DeepSF: deep convolutional neural network for mapping protein sequences to folds. *Bioinformatics*, 34(8):1295–1303, 12 2017.

[10] C. N. Magnan and P. Baldi. SSpro/ACCpro 5: almost perfect prediction of protein secondary structure and relative solvent accessibility using profiles, machine learning and structural similarity. *Bioinformatics*, 30(18):2592–2597, Sep 2014.

[11] Roshan Rao, Jason Liu, Robert Verkuil, Joshua Meier, John F. Canny, Pieter Abbeel, Tom Sercu, and Alexander Rives. Msa transformer. *bioRxiv*, 2021.

[12] R. Dustin Schaeffer and Valerie Daggett. Protein folds and protein folding. *Protein engineering, design & selection : PEDS*, 24(1-2):11–19, Jan 2011. 21051320[pmid].

[13] Akif Uzman. Molecular biology of the cell (4th ed.): Alberts, b., johnson, a., lewis, j., raff, m., roberts, k., and walter, p. *Biochemistry and Molecular Biology Education*, 31(4):212–214, 2003.

Received January 01, 2017