Classifying Protein Stability Changes upon Mutations using PRoBERTa

Anowarul Kabir
Department of Computer Science
George Mason University
akabir4@gmu.edu

I. Introduction

Proteins are vital in many biological processes in organisms, since each cell's correct functioning is determined by the certain tertiary structure of the proteins. Contrarily, a mutation, even a single amino-acid substitution, can bring about changes in protein structure, thus thermodynamic stability, quantified as $\Delta\Delta G$ (pronounced as DeltaDeltaG or ddG) value. This mutation can be stabilizing ($\Delta\Delta G \geq 0.0$ kcal/mol) or destabilizing ($\Delta\Delta G < 0.0$) [1]. Stabilizing mutation is occurred when the change in the mutated protein structure is negligible compared to it natural state, called wildtype structure. On the contrary, destabilizing mutation causes distortion to the wildtype structure, consequently prone many disorders in organisms. In this work, I study a computational approach based on neural network to classify protein mutation types from only protein sequences.

Due to the fact that mutagenesis wet-lab experiments are inadmissible for routine tasks, because of their laborious and expensive inherent nature, and the availability of datasets [2, 3] have opened up promising opportunities for applying computational approaches. Following that, predicting protein mutation types can be formulated as a classification problem. Towards this bearing, many computational methods have been developed over the years to predict the mutation types as well as estimating thermodynamic stability changes of proteins upon mutations. Available methods considered various aspect information from sequences, structures, statistical potentials, energy potentials, feature engineering and so on. In this task, only the wildtype and its mutated sequence are considered and I apply a variant of optimized RoBERTa [4] model called PRoBERTa [5] to classify protein mutation events.

Section II describes the methodology of how the classification problem is tackled from only sequence, in addition with the brief description of the pretrained PRoBERTa [5] model in Section II-A. Section III highlights the dataset used, analyze the learned embeddings of the protein sequences in Section III-B and shows the developed models performance analysis in Section III-D.

II. Methodology

Assuming, there are N training data points available such that (x_i^w, x_i^m, y_i) for $i = 1, 2, ...N$ where x_i^w, x_i^m and y_i denotes i-th wildtype protein sequence, variant protein sequence and their corresponding mutation type, respectively. Each protein (wildtype or variant) sequence is comprised of n_i amino acids.

$$x_i = r_1^{(i)} r_2^{(i)} .....r_{n_i}^{(i)} \tag{1}$$

where $r_j^{(i)}$ denotes the j-th amino acid of x_i sequence. Then, each sequence is tokenized using byte-pair encoding (BPE) [6]. The size of the vocabulary is set to 10,000 following Ananthan *et la.* [5] since the authors mentioned that the average token length does not increase much beyond $10,000$ tokens. This indicates that most of the longer tokens are detected in the first $10,000$ iterations. The tokenized sequence is a set of amino acid clusters based on how amino acids occurs together in a corpus of protein sequences, essentially reduces the dimensionality. Thereby, this process limits the protein sequence representation only to short distance relation and discard the long distance relations.

$$x_i = tok_1^{(i)} tok_2^{(i)} .....tok_{k_i}^{(i)} = BPE(r_1^{(i)} r_2^{(i)} .....r_{n_i}^{(i)}) \tag{2}$$

An embedding function, $Em(token)$, is deployed to extract features for each token. The embedding is the high dimensional representation of each amino-acid clusters. To get the final embedding of a protein sequence (wildtype or variant), all vector representation are summed up.

$$E_{x_i} = Em(tok_1^{(i)}) + Em(tok_2^{(i)}) + ... + Em(tok_{k_i}^{(i)}) \tag{3}$$

A classification module is used to predict the mutation type given a wildtype sequence embedding, $E_{x_i}^w$, and its variant sequence embedding, $E_{x_i}^m$.

$$pred_{x_i} = g(E_{x_i}^w, E_{x_i}^m | \theta) \tag{4}$$

The model is parameterized by θ, which is learned by minimizing cross entropy loss as the following.

$$\theta = \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} out_{x_i} log(pred_{x_i}) + (1 - out_{x_i}) log(1 - pred_{x_i}) \tag{5}$$

where out_{x_i} and $pred_{x_i}$ are the target and predicted class probability distribution.

A. Embedding function

The embedding function is a transformer neural network [7] which is pre-trained on task-agnostic sequence embedding problem, more specifically on masked language model (MLM) tasks, called PRoBERTa [5]. The model consists of an embedding layer, followed by 5 stacked transformer encoder and same number of stacked decoder layers. It has approximately 44 million trainable parameters. To train the model, each tokenized amino acid sequence is either truncated or padded to a fixed-length sequence of 512 tokens. In the experiments, the maximum vocabulary size was set to $10,000$. The pretrained model is taken from the [5] and extract the last layer features as the embedding of the tokenized amino acid sequence. In the experiment, if a sequence is comprised of k tokens, the feature embedding size is $(k, 768)$. Then each vector representation is summed up to get the final embedding of the protein sequence following Eq. 3. The analysis over the task-agnostic learned embeddings for both the vocabulary and protein sequences from the train set is described in Section III-B.

III. Experiments and analysis

A. Dataset

The source dataset [8] is divided into train and test set where the train and test set contains 5444 mutations of 211 proteins and 276 mutations of 37 proteins, respectively, and there are no common proteins between them (Fig. 1(a)). However, the dataset contains different $\Delta\Delta G$ values for the same mutation of the same protein. For instance, hiv-1-capsid protein (PDBID (Protein Data Bank id):1A43) contains $\Delta\Delta G$ of -2.3 and -2.8 for the same mutation C-218-S (where 218th residue is substituted from Cysteine to Serine in mutant type) at the same environment condition, such as pH and temperature. To solve this issue, the average of $\Delta\Delta G$ is taken over the same mutation of a protein following Potapov *et. al* [9].

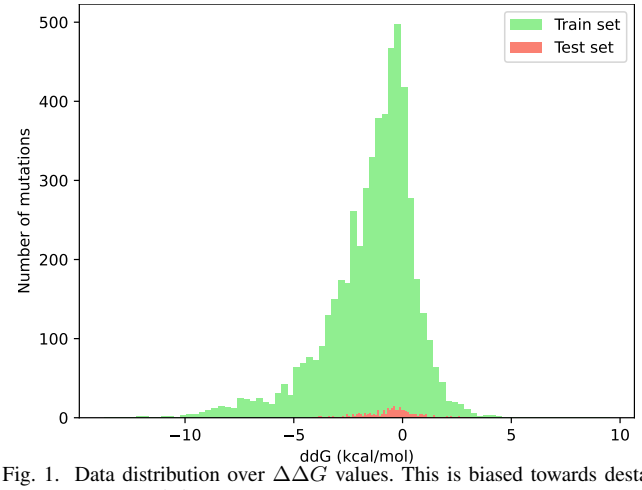

Fig. 1. Data distribution over $\Delta\Delta G$ values. This is biased towards destabilizing mutations ($\Delta\Delta G < 0$) since there are 4 times more destabilizing data points than stabilizing ones ($\Delta\Delta G \geq 0$)

After cleaning the final train and test set contains 4344 mutations of 209 proteins and 253 mutations of 37 proteins. Furthermore, a validation set is created from the training set by randomly choosing 20 proteins and their corresponding all mutations. Therefore, not overlapping among train, validation and test set holds. Table I summarizes the class distribution over train, validation and test test.

TABLE I
Class distribution over train, validation and test set. Note that, there is no overlapping proteins among these three sets.

Data	#-Stabilizing	#-Destabilizing	#-Proteins
Train	894	3119	183
Validation	53	277	20
Test	71	183	37

B. Embedding analysis

There are two types of embeddings in this study: vocabulary embedding where each vocabulary consists of a set of amino acids, and protein sequence embedding where each sequence is comprised of a set of vocabulary. In the following two subsections, both type of embedding is experimented using Principal Component Analysis (PCA) and T-distributed Stochastic Neighbor Embedding (T-SNE). To do that, I utilize the default implementation of PCA and T-SNE from scikit-learn package.

1) Vocabulary embedding: Following Ananthan *et la.* [5], I train the sentencepiece [10] tokenizer model for byte-pair encoding (BPE) [11] using a corpus of sequences. Here, the maximum vocabulary size is set to $10,000$ and the corpus of sequence is used from UniProtKB/Swiss-Prot which contains $565,928$ manually annotated and reviewed protein sequences.

Each of these vocabulary is fed to the pretrained PRoBERTa [5] model to extract features. The feature size is 768 by the construction of the model. Note that, here only the last layer features of the encoder module are considered. Then, PCA and T-SNE are applied by setting the reduced dimension size of 3. The amount of variance explained by the 3 components in PCA is 0.081, 0.067 and 0.051. On the other hand, T-SNE achieved mean sigma of 4.412 and Kullback–Leibler (KL) divergence of 3.303. Fig. 2 depicts the summary of the three principal components for both (a) PCA and (b) T-SNE. Although the amount of variance explained by the components of PCA is very small, however Fig. 2(a) plots the vocabulary into two partitions. Therefore, further experiment is run and shows that the smaller cluster mostly contains those vocab that starts with "_" or "M". These are the vocabularies that starts the protein sequence. Thus, the pretrained model embeddings partition the starting vocabularies from just the corpus of sequence. Fig. 5 shows more fine-grained analysis of Fig. 2 where each 3D plot is comprised of 500 vocabularies and annotated with the vocab itself. Same analysis is also done T-SNE and Fig. 6 summarizes the result.

2) Protein sequence embedding: Each protein sequence is tokenized using the tokenizer model and each token is embedded using the pretrained PRoBERTa [5] model. Then the token vectors are summed to compute the final vector representation of the protein sequence using Eq. 3. Since each

(a) PCA: [0.081, 0.067, 0.051] **(b) T-SNE: [Mean sigma: 4.412, KL divergence: 3.303]**

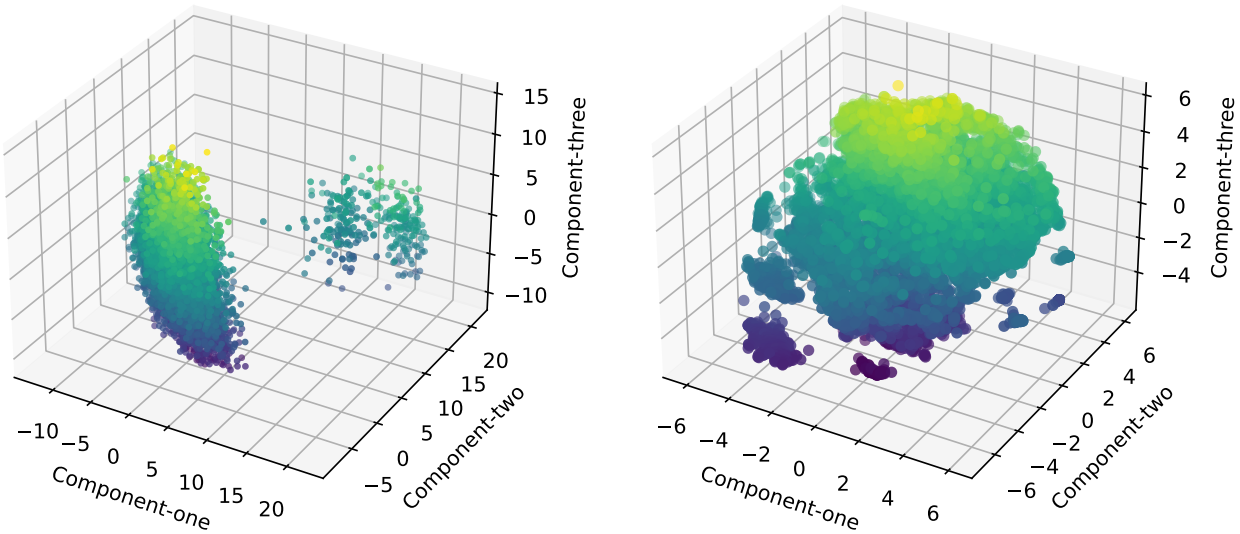

Fig. 2. This illustrates the sequence vocabulary embedding. The vocabulary size is 10,000. (a) PCA: the first three principle components captures the amount of variance of 0.081, 0.067 and 0.051. (b) T-SNE achieved mean sigma of 4.412 and Kullback–Leibler (KL) divergence of 3.303.

vocab embedding is of 768 dimensional, the protein sequence embedding is of 768 dimensional.

PCA and T-SNE are applied to experiment the high dimensional embedding of the sequences. All protein sequences are selected from the train set and Fig. 3 summarizes the result of (a) PCA and (b) T-SNE. The amount of variance explained by the three components is 0.692, 0.062 and 0.042. The first principal component captures the most information which may cause from only considering sequences which captures local amino acid congruence relation. Fig. 7 shows fine grained analysis of 200 sequences each subplot. T-SNE achieved mean sigma of 34.793 and KL divergence of 0.807. PCA and T-SNE shows some clusters using only three main components. Further analysis where each plot is comprised of 200 protein sequences in Fig. 8 shows that each protein sequence and its variant sequence stay very close in the embedding space.

C. Training and testing

From Fig. 1 and Table I, it is evident that the class distribution is biased. Therefore, random sampling for a batch while training cannot be utilized. Instead, given a batch size, I randomly sample data points from both classes of size $batch\_size/2$ without replacement until all the data points from both classes are occurred in at least one batch. As a result, each stabilizing data point may occur in multiple batches whereas destabilizing training examples can only show up once, since the stabilizing class distribution is much smaller than the counterpart. In the training process, only the classification module is trained while keeping the PRoBERTa model freezed. Two classification modules are adopted. Model-0 contains no hidden layers, it only maps the features of size 768 to the classification layer of size 2, therefore it has only $1,536$ learnable weights. Note, the features of wildtype and variant sequence are summed up before feeding the model. The Model-1 is comprised of two hidden layers of size 512 and 64. In this case, the features of wildtype and variant sequence are aggregated, and thus the input size of this model is $1,536$. In total it has $8,19,328$ learnable weights. After each hidden unit a ReLU activation function is applied. In both models, a dropout layer with probability of 50% is applied after each layer. The model is optimized by Adam optimizer by minimizing cross-entropy loss using Eq. 5 since the final classification layer is a softmax function which outputs class distribution over two classes for each pair of wildtype and its variant protein sequence. Since the training class distribution is biased, weighted loss function is applied. The hyperparameters, such as learning rate, batch size, loss function weights are selected using a grid search approach. Table II summarizes the hyperparameter options.

TABLE II
HYPERPARAMETER SECTION.

Hyperparameter	Parameter options
Learning rate	$0.001, 0.0001$
Batch size	$32, 64$
Class weights	$[0.4, 0.6], [0.5, 0.5], [0.6, 0.4]$

Initially, there are resulting 12 number of models. Furthermore, the Model-1 is trained for learning rate 0.00001, batch size 64 and all class weights. Thus, in total I have 12 models for Model-0 and 15 models for Model-1. All these models are run 50 epochs. The best model is selected based on the performance over the validation set to reduce the overfitting. The performance is analyzed over the test set in the following section.

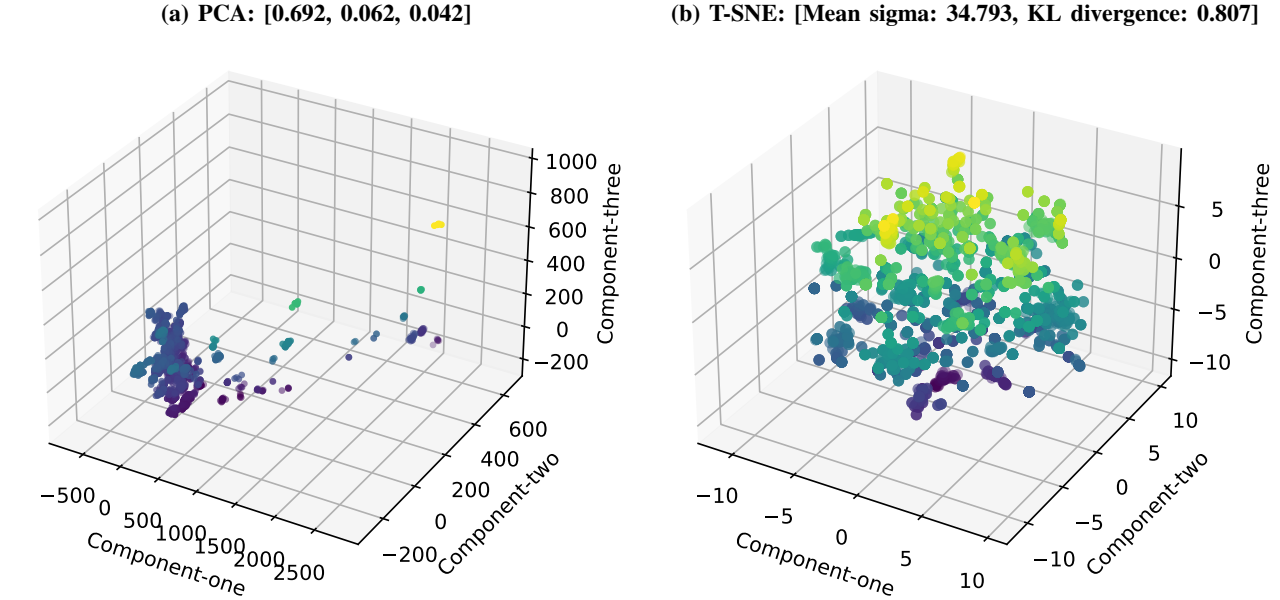

Fig. 3. This illustrates the protein sequence embedding. (a) PCA: the first three principle components captures the amount of variance of 0.692, 0.062 and 0.042. The first principle component is highly significant. (b) T-SNE achieved mean sigma of 34.793 and Kullback–Leibler (KL) divergence of 0.807.

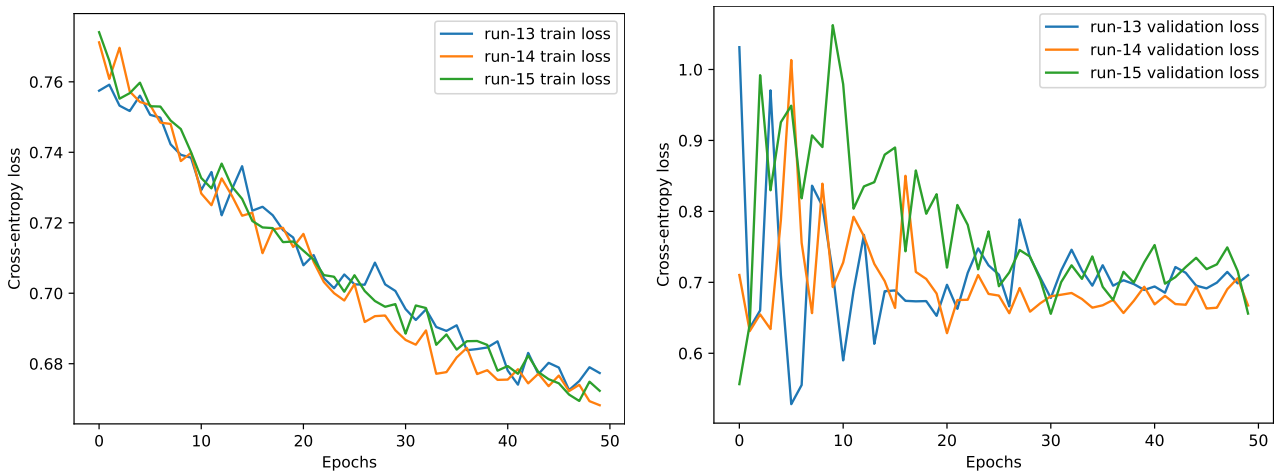

Fig. 4. Train (left) and validation (right) loss of run-13, 14, 15 for Model-1.

D. Performance analysis

Fig. 4 shows the best loss function minimization with respect to cross-entropy of run-13, 14 and 15 for Model-1 versus the number of epochs over the train set (left) and the validation set (right). Fig. 9 and Fig. 10 shows the train and validation loss for all 12 runs of Model-0 and Model-1 where the mode of the loss function is collapsed into two values: ≈ 0.78 or ≈ 0.47, except run-12 of Model-1. The rationale behind this is that the model may not be leaning both class distribution, therefore it is predicting only one class, either stabilizing or destabilizing.

Table III shows the analysis of the class prediction using precision, recall, accuracy, f1-score and area under the curve (AUC) values. All models predict a single class except run-12, 13, 14, and 15 of Model-1. The rationale behind the mode collapse can be multi-modal and crucial for further study. First, the model is trained on the learned features based on only sequence which may not have enough information to solve the problem. Second, a wildtype and its variant sequence differs in only one position in the amino acid sequence in this study. Therefore, the problem, predicting the change-type, stabilizing or destabilizing, based on very tiny distortion compared to the dynamic sequence length, poses a difficult problem. Third, the number of features corresponding to variant amino acids compared with other amino acids is same. Therefore the model may not able to compute the parameters with respect to the mutated position. In addition, the best performing model, run-15 of Model-1, is biased towards predicting destabilizing class. Moreover, I study the learning-based methods that consider only sequence based features such as PROTS-RF [12] and PON-tstab [13]. Commonly they used sequence similarity features, conservation features, amino acid features and neighborhood features. The PCA and T-SNE analysis

TABLE III
SUMMARY OF PERFORMANCE COMPARISON OF ALL MODELS WITH THEIR HYPERPARAMETER SETTINGS.

	Hyperparameters				Performance analysis					
Run no	Learning rate	Batch-size	Epochs	Class weights	Confusion matrix[c]	Accuracy	Precision	Recall	F1-score	AUC
Model-0										
1	0.001	32	50	0.6, 0.4	0, 182, 0, 71	0.280	0.280	1.0	0.438	0.5
2	0.0001	32	50	0.6, 0.4	182, 0, 71, 0	0.719	0	0	0	0.5
3	0.001	32	50	0.5, 0.5	182, 0, 71, 0	0.719	0	0	0	0.5
4	0.0001	32	50	0.5, 0.5	182, 0, 71, 0	0.719	0	0	0	0.5
5	0.001	32	50	0.4, 0.6	0, 182, 0, 71	0.280	0.280	1.0	0.438	0.5
6	0.0001	32	50	0.4, 0.6	0, 182, 0, 71	0.280	0.280	1.0	0.438	0.5
7	0.001	64	50	0.6, 0.4	0, 182, 0, 71	0.280	0.280	1.0	0.438	0.5
8	0.0001	64	50	0.6, 0.4	0, 182, 0, 71	0.280	0.280	1.0	0.438	0.5
9	0.001	64	50	0.5, 0.5	182, 0, 71, 0	0.719	0	0	0	0.5
10	0.0001	64	50	0.5, 0.5	0, 182, 0, 71	0.280	0.280	1.0	0.438	0.5
11	0.001	64	50	0.4, 0.6	182, 0, 71, 0	0.719	0	0	0	0.5
12	0.0001	64	50	0.4, 0.6	0, 182, 0, 71	0.280	0.280	1.0	0.438	0.5
Model-1										
1	0.001	32	50	0.6, 0.4	0, 182, 0, 71	0.280	0.280	1.0	0.438	0.5
2	0.0001	32	50	0.6, 0.4	0, 182, 0, 71	0.280	0.280	1.0	0.438	0.5
3	0.001	32	50	0.5, 0.5	0, 182, 0, 71	0.280	0.280	1.0	0.438	0.5
4	0.0001	32	50	0.5, 0.5	182, 0, 71, 0	0.719	0	0	0	0.5
5	0.001	32	50	0.4, 0.6	182, 0, 71, 0	0.719	0	0	0	0.5
6	0.0001	32	50	0.4, 0.6	182, 0, 71, 0	0.719	0	0	0	0.5
7	0.001	64	50	0.6, 0.4	182, 0, 71, 0	0.719	0	0	0	0.5
8	0.0001	64	50	0.6, 0.4	182, 0, 71, 0	0.719	0	0	0	0.5
9	0.001	64	50	0.5, 0.5	0, 182, 0, 71	0.280	0.280	1.0	0.438	0.5
10	0.0001	64	50	0.5, 0.5	182, 0, 71, 0	0.719	0	0	0	0.5
11	0.001	64	50	0.4, 0.6	182, 0, 71, 0	0.719	0	0	0	0.5
12	0.0001	64	50	0.4, 0.6	15, 167, 4, 67	0.324	0.439	0.943	0.439	0.513
13	0.00001	64	50	0.6, 0.4	29, 153, 19, 52	0.320	0.253	0.732	0.376	0.445
14	0.00001	64	50	0.5, 0.5	57, 125, 24, 47	0.411	0.273	0.661	0.386	0.487
15	0.00001	64	50	0.4, 0.6	53, 129, 14, 57	0.434	0.306	0.802	0.443	0.547

[c] True positive, false positive, false negative and true negative.

over the learned embeddings of the proteins sequences have shown potentials to use them with other features. Combining them together may improve the performance for the mutation stability prediction tasks. For the constraint of time, this study could not cover the combination of features which is one of the future directions.

IV. SOURCE CODE

The code and train/validation/test data are available in https://github.com/akabiraka/mutation_analysis_by_PRoBERTa. It also contains the setting up instructions, features computation, training the model and evaluation process. The code is run partially in GMU Argo cluster [14].

REFERENCES

[1] Fabrizio Pucci, Katrien V Bernaerts, Jean Marc Kwasigroch, and Marianne Rooman. Quantification of biases in predictions of protein stability changes upon mutations. *Bioinformatics*, 34(21):3659–3665, April 2018.

[2] Rahul Nikam, A Kulandaisamy, K Harini, Divya Sharma, and M Michael Gromiha. ProThermDB: thermodynamic database for proteins and mutants revisited after 15 years. *Nucleic Acids Research*, 49(D1):D420–D424, November 2020.

[3] A Kulandaisamy, R Sakthivel, and M Michael Gromiha. MPTherm: database for membrane protein thermodynamics for understanding folding and stability. *Briefings in Bioinformatics*, 22(2):2119–2125, April 2020.

[4] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[5] Ananthan Nambiar, Maeve Heflin, Simon Liu, Sergei Maslov, Mark Hopkins, and Anna Ritz. Transforming the language of life. In *Proceedings of the 11th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*. ACM, September 2020.

[6] Philip Gage. A new algorithm for data compression. *C Users J.*, 12(2):23–38, feb 1994.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[8] Huali Cao, Jingxue Wang, Liping He, Yifei Qi, and John Z. Zhang. Deepddg: Predicting the stability change of protein point mutations using neural networks. *Journal of Chemical Information and Modeling*, 59(4):1508–1514, Apr 2019.

[9] V. Potapov, M. Cohen, and G. Schreiber. Assessing computational methods for predicting protein stability upon mutation: good on average but not in the details. 22(9):553–560, June 2009.

[10] Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *CoRR*, abs/1808.06226, 2018.

[11] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909, 2015.

[12] Yunqi Li and Jianwen Fang. PROTS-RF: A robust model for predicting mutation-induced protein stability changes. *PLoS ONE*, 7(10):e47247, October 2012.

[13] Yang Yang, Siddhaling Urolagin, Abhishek Niroula, Xuesong Ding, Bairong Shen, and Mauno Vihinen. PON-tstab: Protein

variant stability predictor. importance of training data quality. *International Journal of Molecular Sciences*, 19(4):1009, March 2018.

[14] Argo wiki. Argo Cluster. http://wiki.orc.gmu.edu/index.php/Main_Page, 2021.

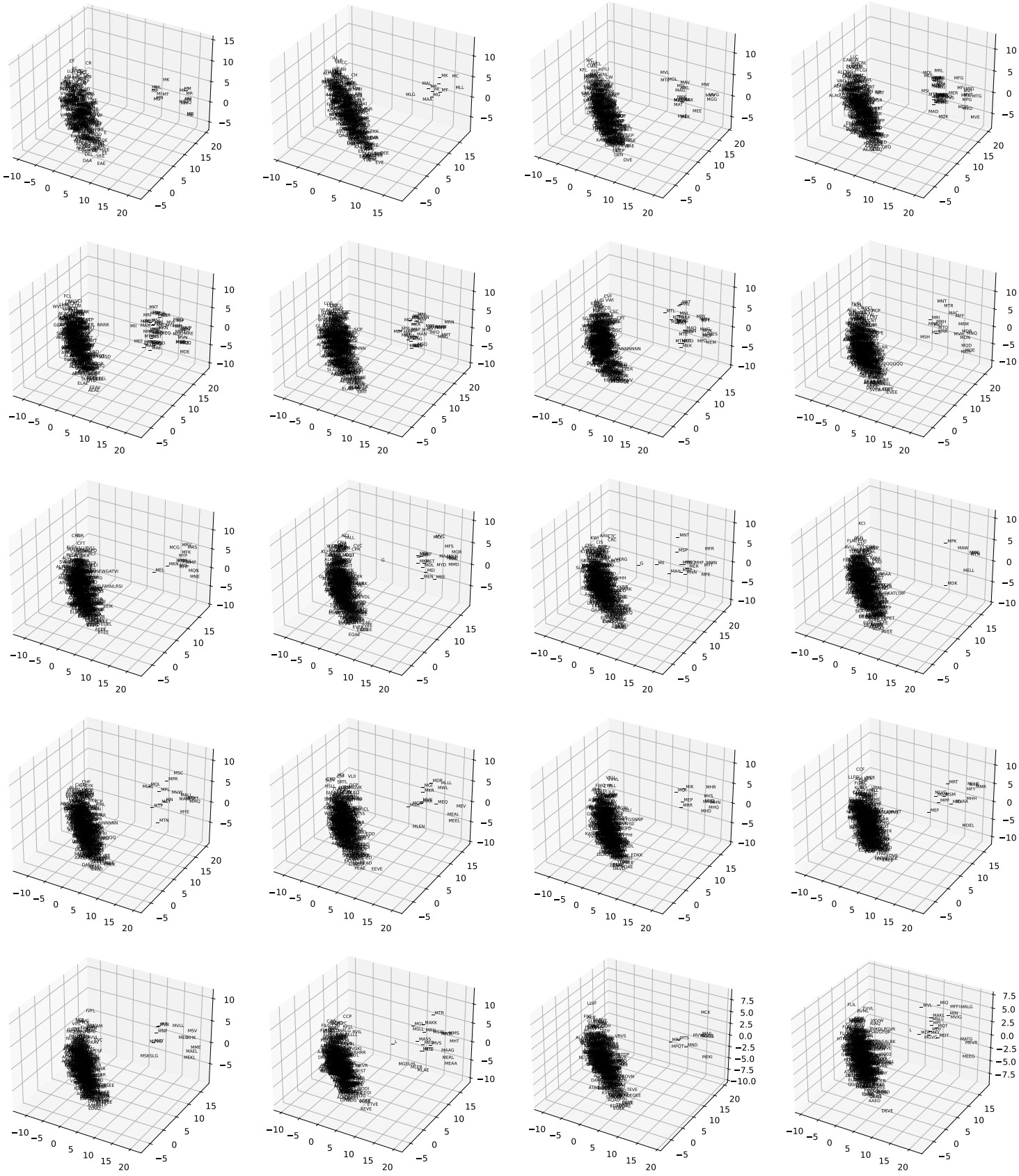

Fig. 5. All 10,000 vocabulary embedding by PCA. Each 3D plot is comprised of 500 vocabularies.

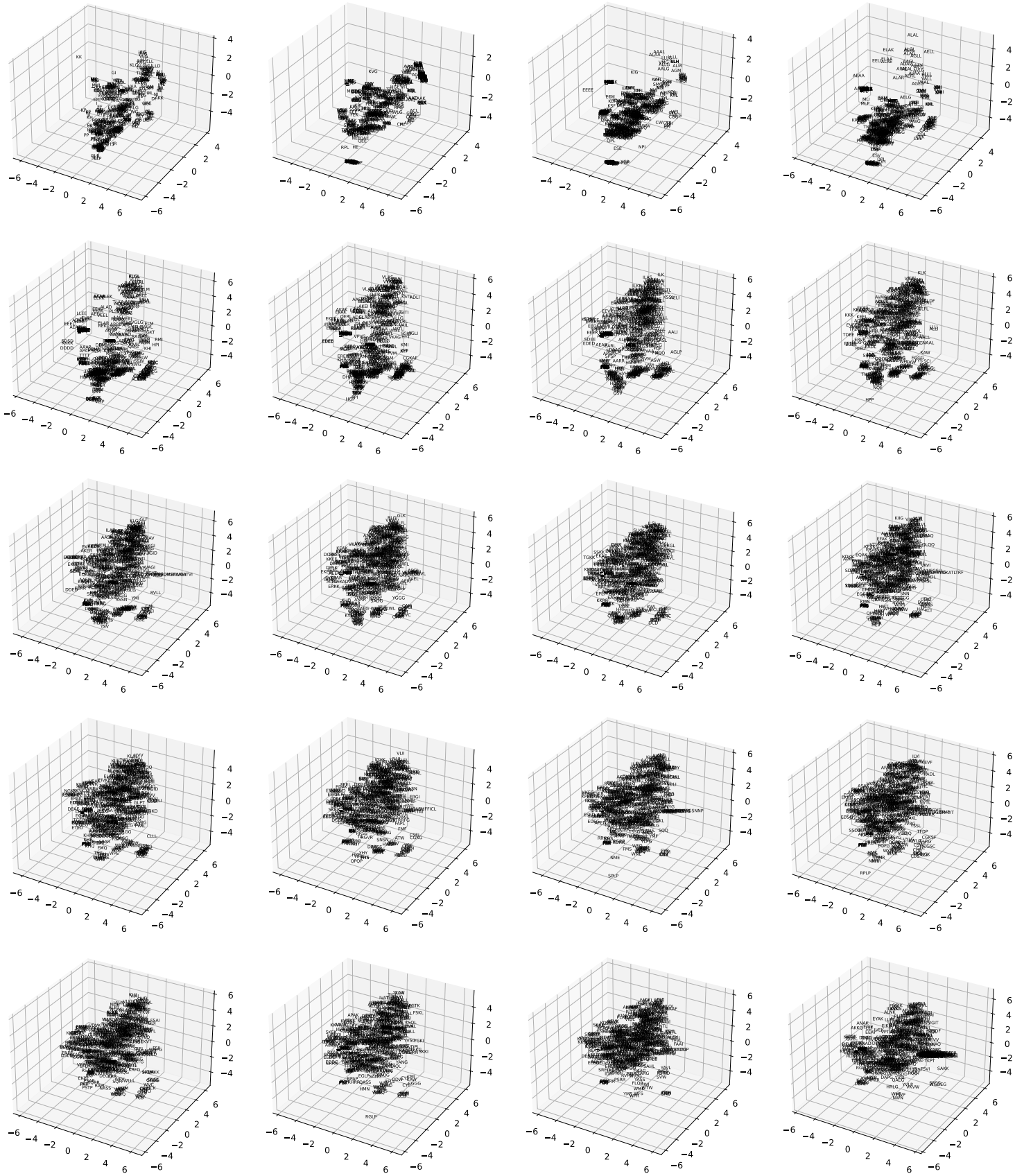

Fig. 6. All 10,000 vocabulary embedding by TSNE. Each 3D plot is comprised of 500 vocabularies.

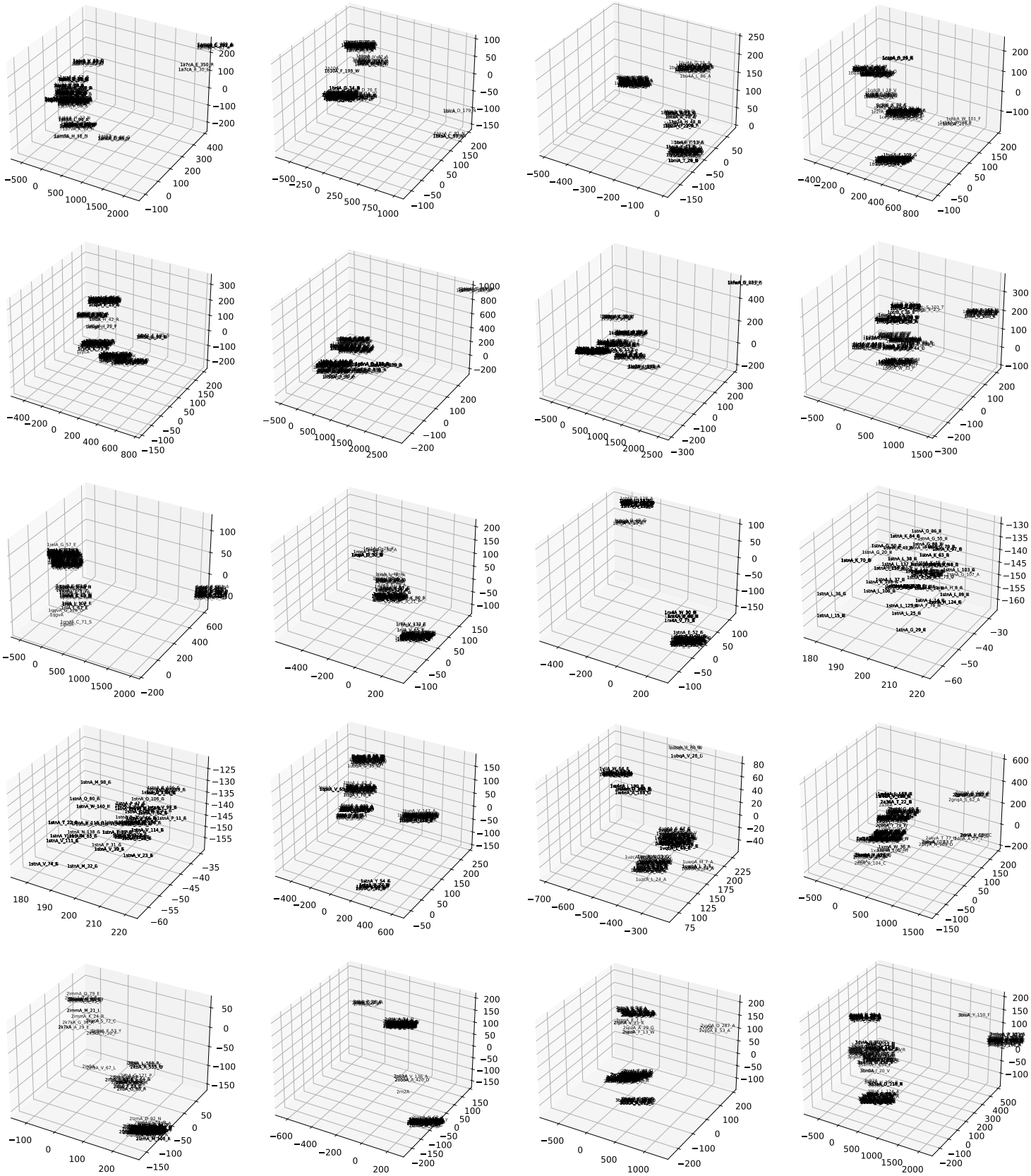

Fig. 7. 4,000 protein sequence embedding by PCA from train set. Each 3D plot is comprised of 200 sequences.

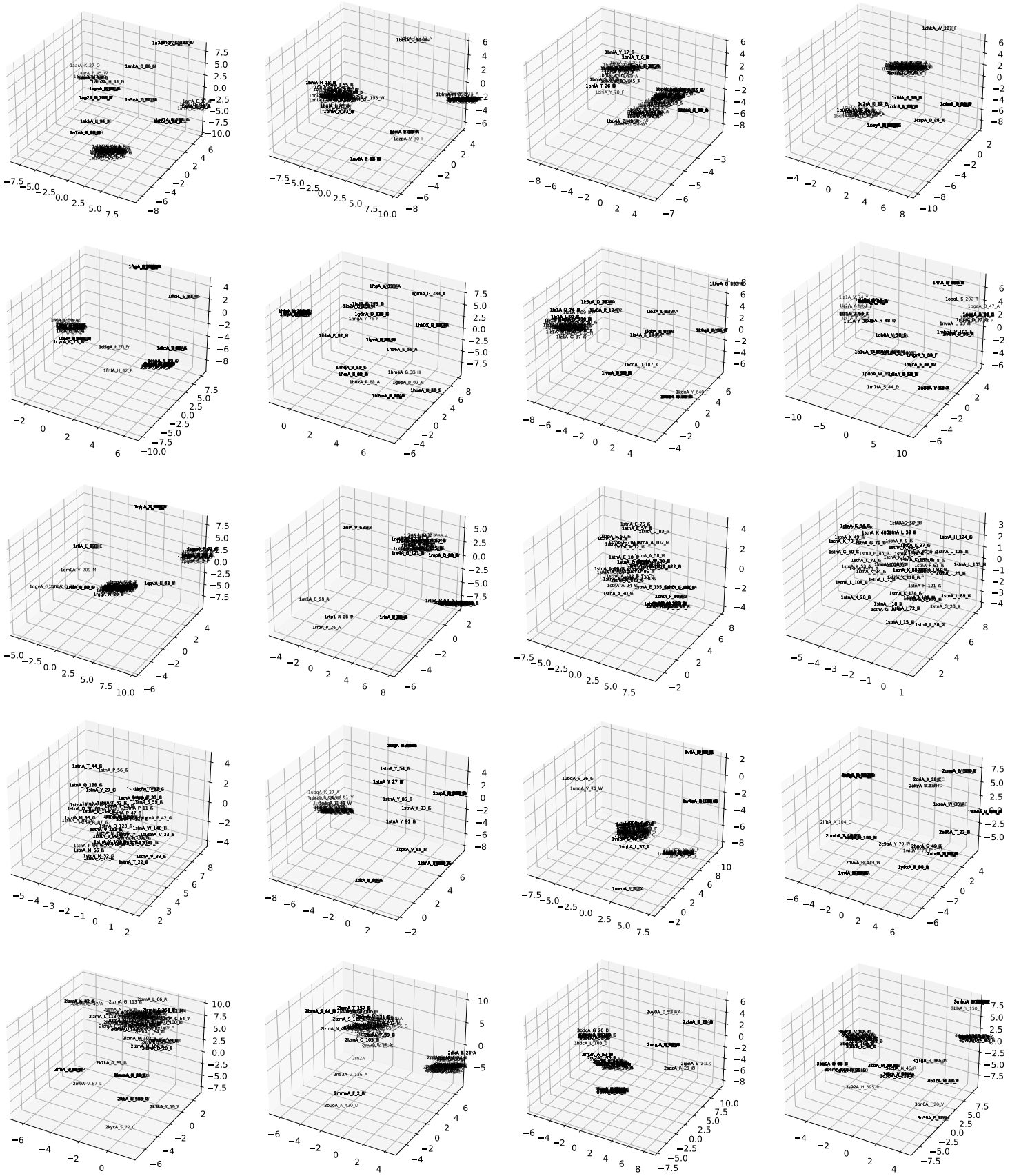

Fig. 8. 4,000 protein sequence embedding by TSNE from train set. Each 3D plot is comprised of 200 sequences.

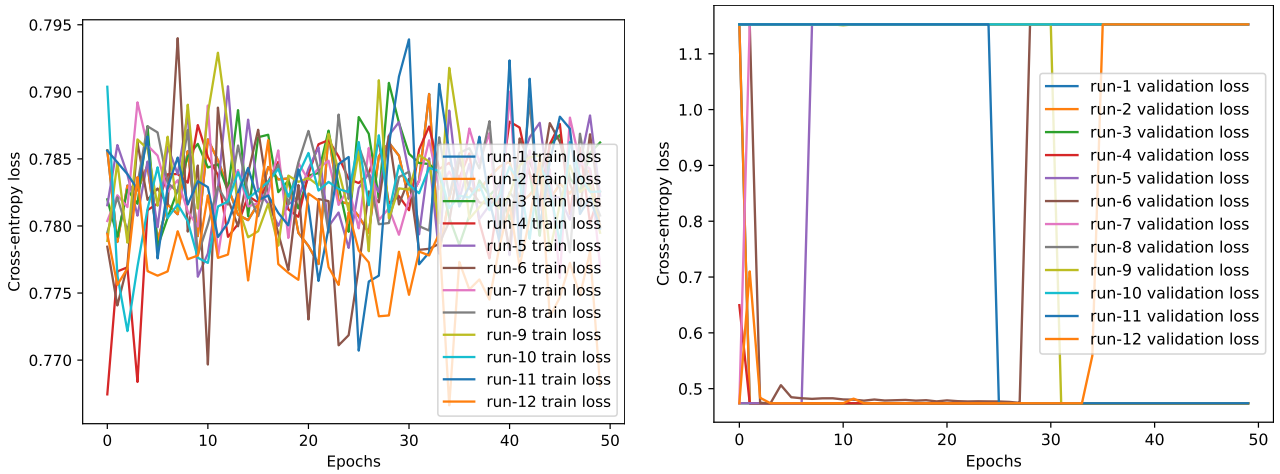

Fig. 9. Train (left) and validation (right) loss of 12 runs with different parameter settings of Model-0. It shows no convergence at all.

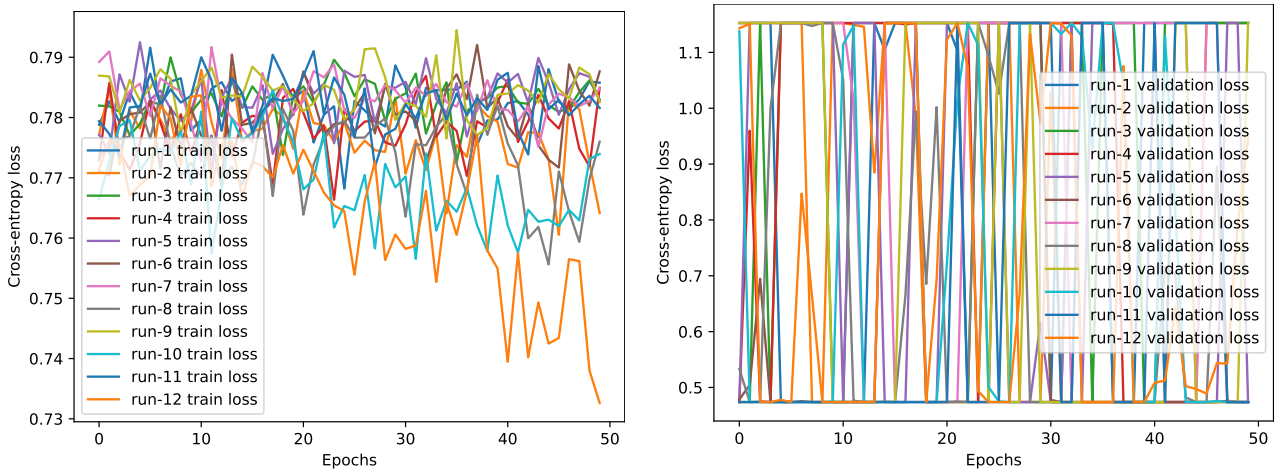

Fig. 10. Train (left) and validation (right) loss of 12 runs with different parameter settings of Model-1. It shows run-12 converged at training bug oscillated at validation.