# Protein Family Prediction using Classical Machine Learning and Deep Learning

Mattia Perrone[1]

***Abstract*— Proteins are among the most important biological macromolecules for living beings, performing essential functions from accelerating chemical reactions to providing structure and support for cells. Protein family classification allows to understand which family does a protein belong to, and therefore its function. Knowing the function of a protein is a crucial point, since it permits delineating which application may be more suitable for that specific protein. Artificial intelligence is a powerful tool that can be used when working with large databases, like in this scenario. This work includes the implementation of both classical machine learning (k-nearest neighbors, decision trees, bagging, XGBoost) and deep learning (LSTM network) classification models for protein family prediction. Classical machine learning models were trained on a dataset including protein biophysical properties from the Protein Data Bank, while the deep learning model was trained using protein amino acids sequences, always from the same dataset. Predictions of three protein families, which are hydrolase, transferase, and oxidoreductase, were carried out in this study. Macro F1-score has been used as metric, since the task to address is a multi-class classification problem with imbalanced classes. The LSTM model performed better than the best machine learning classification model, which is XGBoost (F1-score respectively of 75% and 66%). Possible improvements in terms of the deep learning model may include the use of a bi-directional LSTM architecture and the implementation of CNN-LSTM networks.**

## I. Introduction

Detection of protein family in large databases is one of the primary research goals in structural and functional genomics [1]. A protein family is a group of related proteins that may share a common function, such as hydrolase, transferase, or oxidoreductase [2]. Protein families classification can significantly help delineate functional diversity of homologous proteins, predict function based on domain architecture and provide valuable evolutionary insights. Proteins are considered essential to life because they perform several fundamental functions, such as DNA replication, accelerating metabolic reactions, and a variety of other important tasks carried out within an organism. It is thus becoming increasingly important to predict protein families classification in order to understand which applications may be more appropriate for a specific protein. For example, hydrolase is a class of proteins able to break chemical bonds using water, transferase are proteins able to catalyse the transfer of specific functional groups from one molecule to another, and oxidoreductase are proteins that allow the of electrons from one molecule to another.

[1]Mattia Perrone is a Master student in Biomedical Engineering, University of Illinois at Chicago, Email: mperro3@uic.edu

Artificial intelligence is a powerful tool that can be used to predict protein family. The two main approaches used in the literature for this specific aim are machine learning and deep learning. In the following paper, the expression "machine learning" refers just to "classical machine learning" and not to the whole subset of artificial intelligence including also deep learning. Machine learning can be employed when predicting the family memberships of a given unknown protein starting from its common features with known families. Deep learning, which is a subset of machine learning, is instead used when the model predicts protein families based on proteins primary structures rather than common features between proteins.

### A. Machine learning in protein family prediction

Various biological experimental approaches have been proposed in recent decades with the goal of predicting which families do proteins belong to. Hugh et al. identified different protein families based on the electrostatic potential and structural units of single proteins [3]. However, these experiments frequently require significant time and expensive materials. Computational methods have advantages over experimental methods for completing this task from massive amounts of data [4]. Many computational approaches for proteins family prediction have been developed, with machine learning and deep learning methods being among the most effective [4].

Machine learning can be applied to dataset having as features the parameters of biological experiments, such as X-ray crystallography and NMR spectroscopy. Varshavsky et al. conducted a study using physico-chemical amino acid properties to construct their input feature vector [5]. Examples of the dataset features are the number and kind of amino acids for each protein, their molecular weights and hydrophobicity indexes. Models such as support vector machine (SVM), random forest and other classification algorithms have been used for protein family prediction [4]. Another approach that appears in the literature is to combine classifiers or models by using ensemble techniques [2].

Computational approaches have advantages when it comes to processing sequential data, and there is growing evidence that predicting protein families solely from primary sequences is more effective than experimental methods [6]. With these proteins belonging to different families, the question of whether it is possible to determine a protein's family type based on its sequence arises. Although machine learning has been used for this purpose [7] [8], deep learning has proven to be more appropriate [9] [10] [11].

### B. Deep learning in protein family prediction

Proteins are organic macromolecules made up of amino acids linked together by peptide bonds. The amino acids sequence is defined as the protein's primary structure. A total of 20 different amino acids exists in nature, each of which is identified by a specific alphabet letter. Proteins differ from one another based on the arrangement of amino acids in their genes' nucleotide sequence. This causes protein folding to form specific 3D structures, which determine the proteins' unique functionality. The primary structure of proteins is a key factor determining the complex 3D structure, including enough information to classify protein families and therefore their functionalities. As a result, protein families are groups of proteins that have similar structures both at the sequence and molecular levels and perform similar functions.

When dealing with sequence data, such as amino acid sequences, recurrent neural networks (RNN) are a common deep learning architecture that is employed. In RNN, learning occurs through a series of steps, with the input of a single step happens to be the output of the previous step, always considering the same hidden layer. In this way, the information acquired by the network during previous steps can influence future predictions. Long Short-Term Memory models are particular type of RNN. This architecture adds "gates" that regulate how much information from the prior step and the present phase influences decision-making. It also filters out memory regions that are less crucial for prediction-making. More flexible memory control is possible because to this method.

The aim of the current study is to use both machine learning and deep learning to perform predictions of three protein families (hydrolase, transferase and oxidoreductase) starting from a dataset of 48.953 proteins from the Protein Data Bank (PDB). Four different classification algorithms have been used for the machine learning approach, which are k-nearest neighbor (knn), decision trees, random forest and XGBoost, comparing the F1 score of each method with the others. An LSTM architecture has been implemented for the deep learning approach, performing the hyperparameter optimization using the novel approach Hyperband [12]

## II. Methods

This section illustrates the proposed workflow, starting from a description of the dataset used in the current study. Then, all the preprocessing steps that have been carried out on the dataset are presented, and a description of the models implemented, both concerning machine learning and the deep learning approach, is provided. Finally, the evaluation metric used is presented.

### A. Dataset description

The protein data set used was obtained from Research Collaboratory for Structural Bioinformatics (RCSB) Protein Data Bank (PDB).

The PDB archive is a repository of atomic coordinates and other information describing proteins and other important biological macromolecules, such as DNA and RNA. Structural biologists use techniques such as X-ray crystallography, NMR spectroscopy, and cryo-electron microscopy to determine the location of each atom relative to each other in the molecule. They then deposit this information, which is annotated and publicly released into the archive by the PDB.

The dataset included 18 variables, with the variable "classification" being the output variable. All the features, with the exception of the feature "sequence", where initially used for the machine learning part of the study. The variable "sequence", which includes the amino acids sequences of proteins, were used for the deep learning part of the study.

### B. Preprocessing

The first preprocessing step consists in the analysis of the output variable, which is the family to which a protein belongs to. These variables includes over 5050 different types of molecules, and just three of them, which are hydrolase, transferase and oxidoreductase, have been selected. All the observations that did not include these three kinds of proteins have been discarded. This led to an important dataset reduction, from over 140,000 observations to 48,953. The main reason behind this choice is that the three selected protein types are among the most used in biological experiments, being three enzymes. Moreover, they are also the three most common protein families in the dataset used for the current project.

For the machine learning part, some feature engineering steps have been carried out. These include imputation and skewness reduction. Imputation was performed to solve the problem of missing values, estimating them from the values assumed by all the other features. Skewness reduction was performed to cope with the high number of outliers of some dataset features. Logarithmic transformation was applied to reduce dataset skewness. Then, Pearson coefficients (r) between all the dataset variables were computed, to analyze correlation between them. If the value of r between two features was greater than 0.9, one of the them has been removed to avoid multicollinearity. Both the low values of r between all the features and the output variable, and the reduced number of dataset features indicates that the models that will be applied on this dataset will not likely have high performance.

For the deep learning part, an initial data cleaning step was performed to drop the observations having errors for the feature "sequence". The feature "sequence" includes a sequence of letters, each corresponding to one of the 20 different amino acids present in nature. These mistakes, which are clearly associated to errors during the data labeling process, consist in the presence of letters that do not correspond to any of the 20 amino acids. However, dropping these values did not change much the dataset size, being the observation including the wrong amino acids less than 3% of the whole dataset. Concerning character embedding, a maximum length of characters was set to 350, which is quite close to the mean value of the lengths of all sequences (334). If a protein has more amino acids than 350, then the

sequence is truncated so that the final length is 350, while if a protein has less amino acids than 350, zero padding is performed.

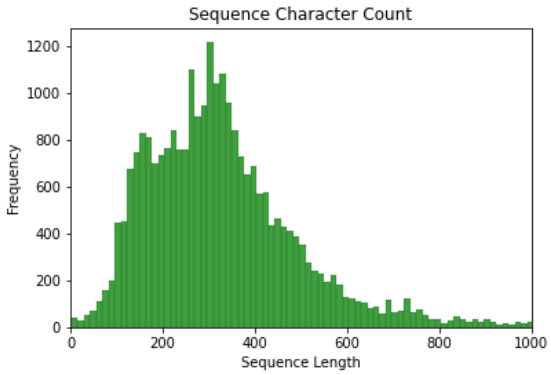


**Fig. 1:** Lengths of protein sequences

### C. Model Selection

Four classification models were used to predict protein families starting from the dataset features. These are knn, decision trees, bagging and XGBoost. Knn was chosen as first approach, since this is one of the simplest models that can be used for classification purposes. Ensemble models were also implemented, in order to reduce the bias obtained with knn. However, before implementing ensemble models, decision trees were chosen as second approach, since they are the basic structures for the two more complex models later implemented. These two ensemble models are bagging and XGBoost. XGBoost was the model that performed better, as it has already proven in other context [13]. All the hyperparameters that were considered for the hyperparameter tuning are reported in Table I, and the graph for XGBoost are reported in Figure 2 and Figure 3.

| | Min Value | Max Value | Step | **Best Value** |
|---|---|---|---|---|
| Knn<br># of neighbors | 1 | 25 | 1 | **1** |
| Decision Trees<br>Max depth | 1 | 40 | 1 | **33** |
| Bagging<br># of estimators | 5 | 40 | 5 | **40** |
| Bagging<br>Max depth estimator | 5 | 40 | 5 | **35** |
| XGBoost<br># of estimators | 5 | 120 | 20 | **105** |
| XGBoost<br>Max depth estimator | 5 | 25 | 5 | **20** |

**TABLE I:** Hyperparameter tuning for machine learning models

An LSTM model was implemented for predicting protein families starting from amino acids sequences. RNN is the most straightforward architecture to use when dealing with sequence models. RNN handles indeed sequence data efficiently, both as it acts on arbitrary length sequence (which is 350 in this study) and as it exploits information during previous steps for future predictions, because of weight sharing across time steps. However, if long sequences are given in input, vanishing gradient problem may arise. An LSTM model was used in the current study, exploiting the concept of memory cells to solve the vanishing gradient issue. The network architecture was taken from Vazhayil et al., since they were able to achieve good results using this architecture for protein family prediction [10]. This consists in an input layer, an embedding layer, a recurrent layer, a dense layer and an activation layer (Table II).

| **Layers** | **Type** | **Output shape** | **Parameters** |
|---|---|---|---|
| 1 | Input | (None, 350) | 0 |
| 2 | Embedding | (None, 350, 128) | 2,688 |
| 3 | LSTM | (None, 350, 128) | 131,584 |
| 4 | Flatten | (None, 44,800) | 0 |
| 5 | Dropout | (None, 44,800) | 0 |
| 6 | Dense | (None, 1800) | 80,641,800 |
| 7 | Dense (activation) | (None, 3) | 5,403 |

**TABLE II:** Architecture of the deep learning model. The total number of trainable parameters is 80,781,475

The embedding layer maps each of the 350-length numerical sequence to a 128 length real value vector, which is given in input to the recurrent layer (LSTM). This layer is followed by a dropout layer, ensuring regularization, and two fully connected layers. The activation functions for these layers are respectively sigmoid, for the first one, and softmax, for the second one.

Categorical cross-entropy was used as activation function to predict the deep learning model's losses. To minimize the loss of categorical-cross entropy, Adam optimization algorithm has been used. Hyperparameter tuning was performed using Hyperband [12], considering the number of LSTM units, the dropout rate, the number of units in the first densely-connected (DC) layer, and the learning rate as hyperparameters. Table III includes the minimum value, maximum value and the steps for each hyperparameter, as long as the best value according to Hyperband. All the experiments were run on GPU using Keras.

| | Min value | Max value | Step | **Best Value** |
|---|---|---|---|---|
| LSTM units | 32 | 128 | 32 | **128** |
| DC layer units | 300 | 3000 | 300 | **1800** |
| Dropout rate | 0.2 | 0.6 | 0.2 | **0.4** |
| Learning rate | 0.0001 | 0.001 | \ | **0.001** |

**TABLE III:** Hyperparameter tuning for the LSTM model

### D. Evaluation metrics

The performances of both the machine learning and the deep learning models were evaluated using F1-score, being the three classes of the output variable imbalanced

$$F1score = 2\frac{Precision * Recall}{Precision + Recall}. \quad (1)$$

Being a multi-class classification problem, macro F1-score has been used as evaluation metrics

$$Macro\ F1score = \sum_{i=1}^{n\ classes} (\frac{F1score_i}{n\ classes}) \quad (2)$$

## III. Results

This section presents an evaluation of the F1-score obtained both by the machine leaning and the deep learning models. Cross-validation (10 splits) has been used to select the best hyperparameters for each of the machine learning model according to the F1-score reached by these models, while holdout validation was used for the deep learning model.

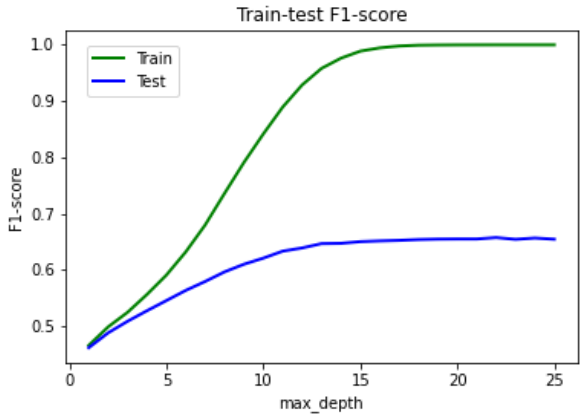


**Fig. 2:** Tuning of the hyperparameter max_depth (maximum depth of base estimators)


**Fig. 3:** Tuning of the hyperparameter n_estimators (number of base estimators)

Concerning the LSTM model, Figure 4 and Figure 5 provide information about how do the loss function and F1-score vary according to the number of epochs the model has been trained on. Finally, Table IV summarizes the F1-score values of each model for the test set.

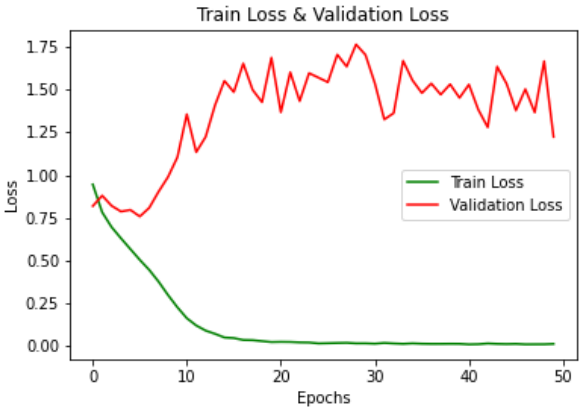


**Fig. 4:** Train and validation loss during model training

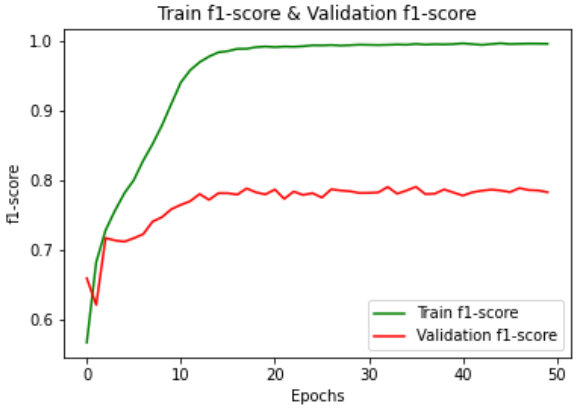


**Fig. 5:** Train and validation F1-score during model training

| | **F1-score - Test set** |
|---|---|
| K nearest neighbors | 0.57 |
| Decision Tree | 0.59 |
| Bagging | 0.66 |
| XGBoost | 0.66 |
| **LSTM model** | **0.75** |

**TABLE IV:** F1-score of all the classification models

## IV. Discussion

The deep learning model performed better than the machine learning ones as expected [10]. In general, this is mostly related to the fact that deep learning models are able to learn high-level features from data, while this is not possible for machine learning models.

In this particularly study, XGBoost is the machine learning model performing better, with F1-score = 0.66 on the test set. The main reason for such a low value is likely related to the low number of features (only 9 after feature engineering) and low values of r between all the features and the output variable. A possible way to address these problems, other than changing the dataset, may be including polynomial features. Another possible reason why these models performed poorly is that the number of hyperparameters tuned for each model was maximum two. This choice was made since machine learning models were

expected to perform worse than deep learning ones, thus less importance to the hyperparameter tuning of these models has been given. LSTM model reached F1-score = 0.75 on the test set. Although this is an acceptable value, there is room for improvement. From Figure 4, the increase in the validation loss with the number of epochs clearly indicates overfitting, as long as higher values of F1-score on the training set with respect to the test set.

This finds a confirmation in the high number of the trainable parameters (over 80.000.000), that makes the model too complex. A valid approach to address this problem would be changing the model architecture. This may allow to keep the same performance, if not increasing it, and will result in a less computationally demanding model training.

## V. Conclusions

This project consists in the implementation of machine learning (knn, decision trees, bagging, XGBoost) and deep learning (LSTM network) classification models to predict which families different proteins belong to, starting from features available in the Protein Data Bank (PDB).

Machine learning models were trained on a dataset including protein biophysical properties, while the deep learning model was trained using protein amino acids sequences.

The LSTM model performed better than the best machine learning classification model (XGBoost), with an increase of F1-score on the test set of 12% IV.

Possible improvements in terms of the deep learning model architecture may include the reduction of the number of trainable parameters, the use of bi-directionals LSTM, to make prediction also based on future information, and the implementation of CNN-LSTM networks, as already proposed in the literature [11].

## VI. Source Code

The source code of this project is available at https://github.com/MattiaPerrone123/Protein_Family_Prediction.git.

## References


[1] A. J. Enright, S. Van Dongen, and C. A. Ouzounis, "An efficient algorithm for large-scale detection of protein families," *Nucleic acids research*, vol. 30, no. 7, pp. 1575–1584, 2002.

[2] J. S Bernardes, "A review of protein function prediction under machine learning perspective," *Recent patents on biotechnology*, vol. 7, no. 2, pp. 122–141, 2013.

[3] H. P. Shanahan, M. A. Garcia, S. Jones, and J. M. Thornton, "Identifying dna-binding proteins using structural motifs and the electrostatic potential," *Nucleic Acids Research*, vol. 32, no. 16, pp. 4732–4741, 2004.

[4] S. Hu, R. Ma, and H. Wang, "An improved deep learning method for predicting dna-binding proteins based on contextual features in amino acid sequences," *PLoS one*, vol. 14, no. 11, p. e0225317, 2019.

[5] R. Varshavsky, M. Fromer, A. Man, and M. Linial, "When less is more: improving classification of protein families with a minimal set of global features," in *International Workshop on Algorithms in Bioinformatics*, pp. 12–24, Springer, 2007.

[6] J. Brown and T. Akutsu, "Identification of novel dna repair proteins via primary sequence, secondary structure, and homology," *BMC bioinformatics*, vol. 10, no. 1, pp. 1–22, 2009.

[7] P. J. Ballester and J. B. Mitchell, "A machine learning approach to predicting protein–ligand binding affinity with applications to molecular docking," *Bioinformatics*, vol. 26, no. 9, pp. 1169–1175, 2010.

[8] Y. H. Li, J. Y. Xu, L. Tao, X. F. Li, S. Li, X. Zeng, S. Y. Chen, P. Zhang, C. Qin, C. Zhang, *et al.*, "Svm-prot 2016: a web-server for machine learning prediction of protein functional families from sequence irrespective of similarity," *PloS one*, vol. 11, no. 8, p. e0155290, 2016.

[9] T. K. Lee and T. Nguyen, "Protein family classification with neural networks," *Accessed: Dec*, vol. 10, p. 2018, 2016.

[10] A. Vazhayil, S. KP, *et al.*, "Deepproteomics: Protein family classification using shallow and deep networks," *arXiv preprint arXiv:1809.04461*, 2018.

[11] Z. Shi, "Graph neural networks and attention-based cnn-lstm for protein classification," *arXiv preprint arXiv:2204.09486*, 2022.

[12] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6765–6816, 2017.

[13] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.