University of Padua - Department of Mathematics
Course: Structural Bioinformatics
Academic year: 2021-22.
Professor: D. Piovesan

Emma Roveroni - 2058618 - emma.roveroni@studenti.unipd.it
Marco Tesser - 2058223 - marco.tesser.1@studenti.unipd.it

Protein contacts' classification

1 Data retrieval

The first phase of the project consisted in retrieving the data and putting it into a suitable data structure to be handled subsequently. The data itself consisted in a dataframe available for each protein in which each row represented a contact in the protein and each column represented a feature about that contact. The last column is the target label, which is the interaction type. These dataframes were then joined together in a single dataframe.

2 Data preprocessing

The first step in the preprocessing phase consisted in removing all the samples in which the label was not available. Secondly the missing values of each feature were replaced by the most common value for that feature (the mode). Only numerical features were kept. Three more preprocessing steps were also performed. Firstly a best subset selection was fulfilled in order to determine which features were the most meaningful ones and which, on the other hand, were not adding any information. The subset selection was performed using a default scikitlearn method that kept only features above a certain threshold of importance and removed the other ones. Secondly data scaling was done. Features were scaled between 0 and 1, in order to have the same compensation from each feature. Finally since the data was heavily unbalanced also oversampling/undersampling were performed. For undersampling an InstanceHardnessThreshold undersampler was used, which fits an estimator on the data and removes the most difficult data points to classify afterwards. As for the oversampler, SMOTE was used, which uses an interpolation between samples technique to create new artificial data points. It must be denoted that this undersampling/oversampling step decreased slightly the overall performance of the model (around 2%) but in exchange it increases the average precision/recall result of all the classes.

3 Model

The model used for the contacts classification was a neural network for multiclass classification. In order to perform multiclass classification in neural networks the first thing that has to be carried out is the encoding of the label of each sample into an identity vector. This is done to then set in the output layer of the net a number of neurons equal to the number of classes in our dataset so that we can train each of these output neurons to determine the probability $\mathrm{P}(C_i \mid \text{data})$, with sum $\mathrm{P}(C_i \mid \text{data}) = 1$. This means that the output layer will effectively represent the probability distribution of a contact of being of a certain type. The input layer, on the other hand, will have size equal to the number of features in the dataset. Beside these mandatory features of the architecture, the other hyperparameters were determined through a grid search approach in which various versions of the net were trained to ascertain the best combination. The metric that was used to establish which model was the best was the ROC_AUC score. This is due to the fact that accuracy is more prone to bias in an unbalanced dataset. The hyperparameters that were iterated through were: the number of layers of the net and the number of neurons of each layer. Others parameters instead were kept fixed at a certain value:

- Batch size = 16000
- Weight initialization: Glorot Normal
- Loss: categorical crossentropy
- Optimizer: Adam
- Activation function (hidden layers): Relu
- Activation function (output layer): Softmax

Other than that an early stopping mechanism was implemented to not let the training run uselessly once the loss has stopped decreasing. Once the best model was determined, a new net was set up with the best models' hyperparameters. To evaluate in the best way the model a 10-fold cross validation was computed. Afterwards the final ROC_AUC score was computed and it was established that the predictor had an ROC_AUC rate equal to 0.92. A training-validation curve was also plotted in order to assess if the model was in any extent underfitting or overfitting. The results are the following:

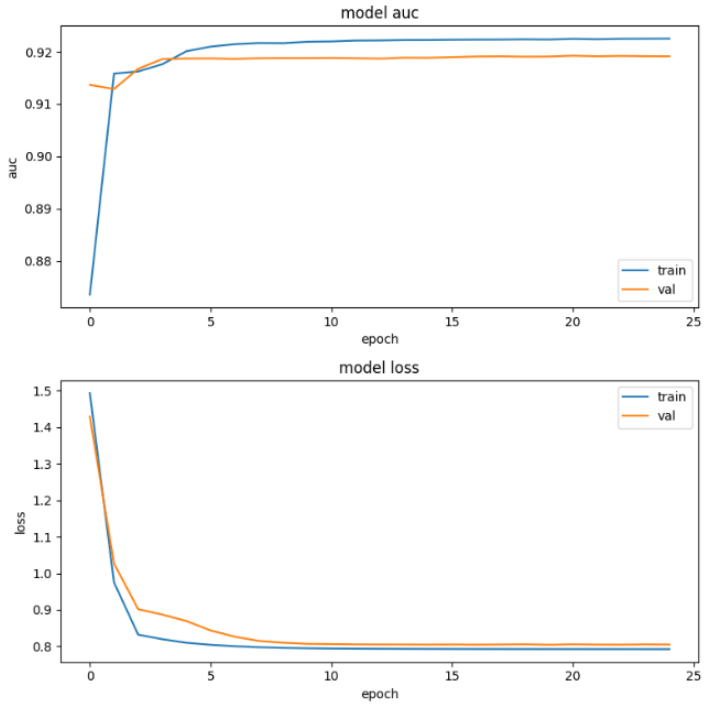

Figure 1: Training-Validation curve

Since the curves coincide, it is established that the model does not show any sign of underfitting or overfitting. Once the final model was obtained, to assess its goodness of fit, a precision-recall histogram was plotted.

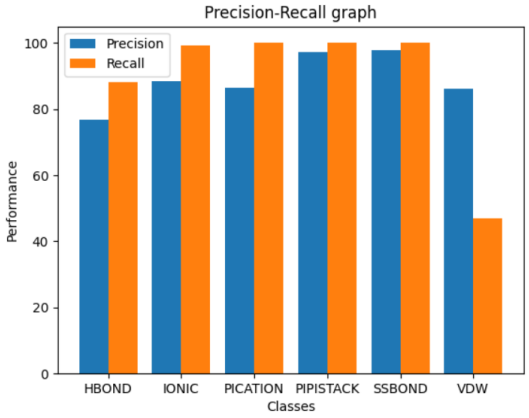

Figure 2: Precision-Recall plot

4 Protein contacts' evaluation

Now that the fully trained model has been computed, it's possible to use it to predict a new protein's contacts. The software will now ask the user to input the 4 letter code of a new protein. Once this is done, it will automatically download the protein from the PDB in mmCIF format, compute its features using a separate script and finally preprocess its data to make it usable as an input for the neural net. Subsequently it will use the trained model to predict the protein's contacts. Once this is finished it will merge the dataframe containing the protein's features with the one containing the probability distribution of each contact of being a certain type of contact. Finally it will download the dataframe as a CSV file.