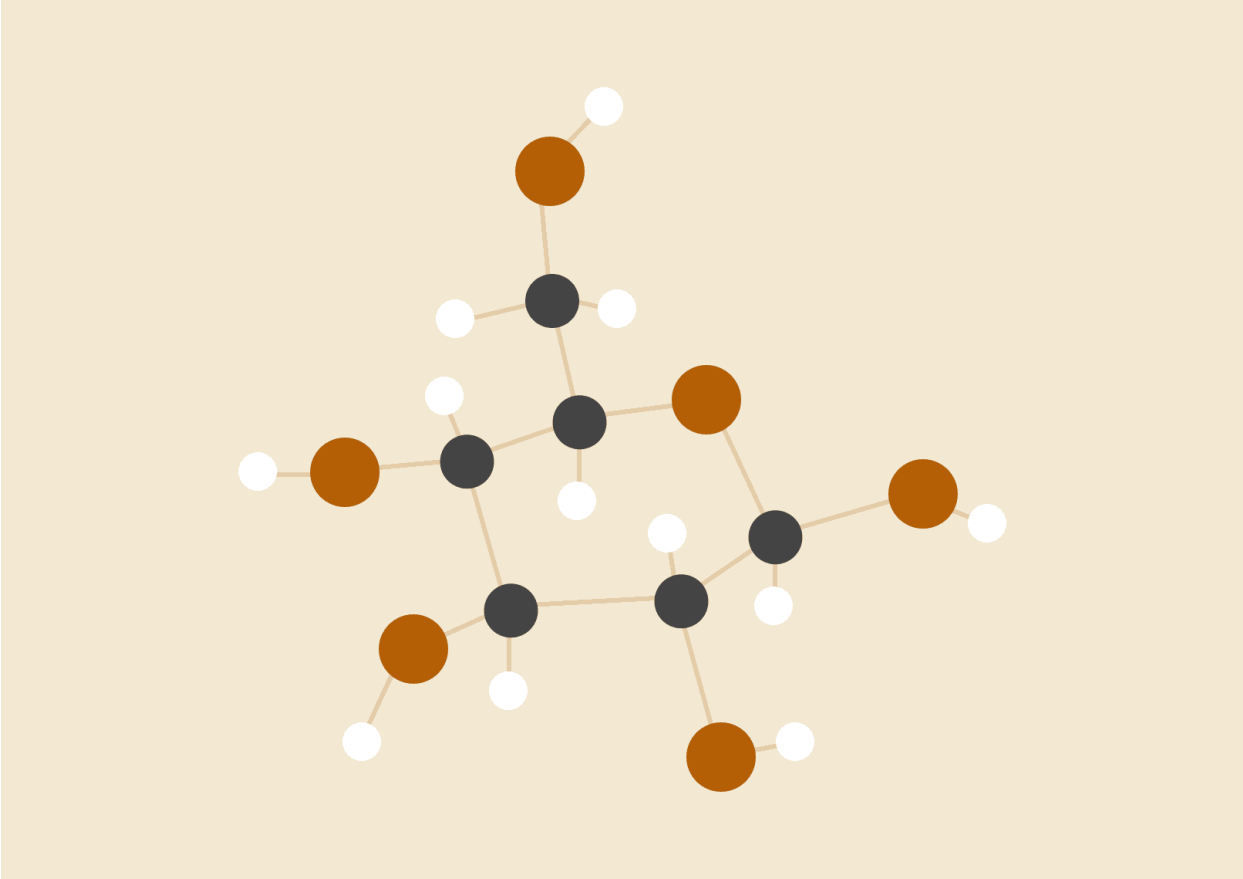
Report for Contacts Classification Project

Structural Bioinformatics 22/23



Elena Stefanovska (id: 2085310)

Angela Kralevska (id: 2072071)

June 2023

Supervisor: phD Damiano Piovesan

INTRODUCTION

In this project we have been tasked with developing software that can predict the RING classification of a contact based on statistical or supervised methods. We have explored various models and architectures that, given two residues and corresponding properties as features, predict the different contact types defined by the RING software. We also calculated additional features using some Biopython modules. Since we are dealing with a multiclass and multilabel dataset that is highly unbalanced, we also employed an oversampling strategy.

DATASET

The dataset contains 6 different types of contacts (Hydrogen bonds (HBOND), Van der Waals interactions (VDW), Disulfide bridges (SBOND), Salt bridges (IONIC), π-π stacking (PIPISTACK), π-cation (PICATION) and one additional category of unclassified contacts.

Our dataset is composed of 1807 PDB files, such that the contacts determined within these files are characterized with 12 different features for each the source and the target residue: DSSP, rsa, half sphere exposure up and down, phi and psi angles, SS 3 states (from angles) and 5 Atchley features. First 4 features serve as residue identifiers, those are the chain, index, insertion code and name of the corresponding residue. These features were not used during the training process.

This data was extracted using the calc_features.py script, such that a .tsv file with the calculations was generated for each protein separately. All these files were merged into one dataframe, with around 700K entries in total, that was used for data analysis and model training.

Adding additional features

In order to explore the effect of adding additional features to our dataset, we have calculated three more features for each residue: hydrophobicity, isoelectric point and aromaticity as boolean feature. We performed this using the Biopython ProteinAnalysis module that requires protein sequence or single amino acid as argument and calculates the desired properties. Another feature we calculated is the distance in Angstrom between the C-alpha atoms (with respect to their coordinates) within each pair of source

and target residues present in the sequence. We found that aromaticity may aid in determining the π-π stacking (PIPISTACK) and π-cation classes, the isoelectric point may represent in a way the charge of the residues and help in determining the ionic bonds. The distance between C-alpha atoms may also be a useful feature for contact classification since residues are placed in different distance ranges for different bonds. However, for the purposes of adding these features, we had to extract all pdb_id's in our dataset. Unfortunately, some file names are not read correctly in python, so we had to exclude these 4 proteins from the final dataset necessary for calculating our newly added features.

EXPLORATORY DATA ANALYSIS

Missing values, duplicates, basic statistics

What we noticed first when loading the dataset is that it contains missing values, both for the features and class labels. Since the dataset contains a lot of entries, we have initially decided to drop these values and continue with the remaining clean 400K instances. We also checked if the dataset contains duplicates which was not the case. Initially, some basic statistics have been calculated for the features in the dataset.

Class distribution

Another observation is the high class imbalance in the dataset. Having multi-label classification with high class imbalance is a challenging problem and should be addressed carefully. The following Figure 1 displays the class distribution. One thing we also noticed is that most of the labels around 270K are assigned one class only, around 50K have 2 classes and the remaining 2.5K with 3 labels at most.

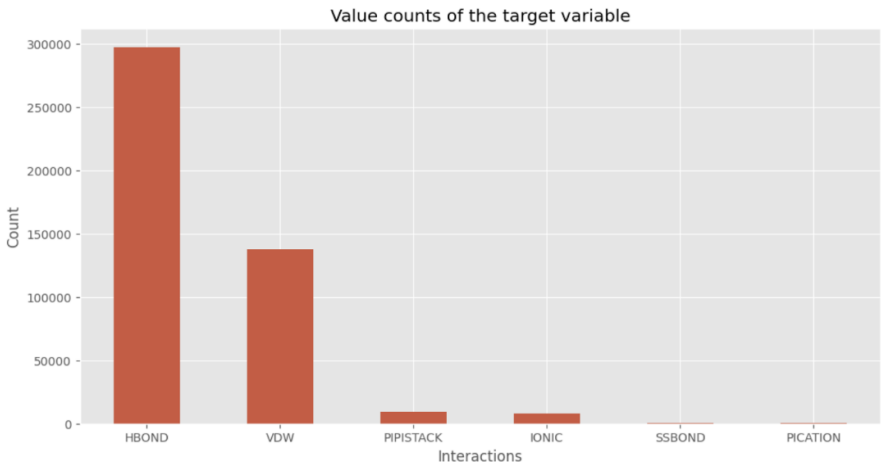


Fig. 1 Class distribution

Pairplots and Correlation matrices

By using the seaborn library, we have calculated scatterplots for each pair of numerical features in the dataset as well as correlation matrix given in Figure 2 for all features. From these correlation plots we may observe some correlations between features s_a1 and s_a5, as well as negative correlation between s_up and s_rsa with a value of around -0.8 as confirmed by the correlation matrices. The conclusion would be residues with greater solvent accessibility are more likely to have less exposure to the upper hemisphere. It may also be observed that the psi angle feature has bimodal distribution.

Another interesting scenario is the positive correlation of around 0.83 between Atchley features 3 (steric parameters - info about spatial requirements and interactions of atoms or groups within a molecule) and 5 (secondary structure likelihood). The conclusion would be that amino acids with larger steric properties tend to have a higher probability of adopting a specific secondary structure.

In addition we may also observe high negative correlation between the hydrophobicity feature and the first Atchley feature, because both in opposite manner describe the hydrophobicity. Therefore we will exclude this feature in the model training later on.

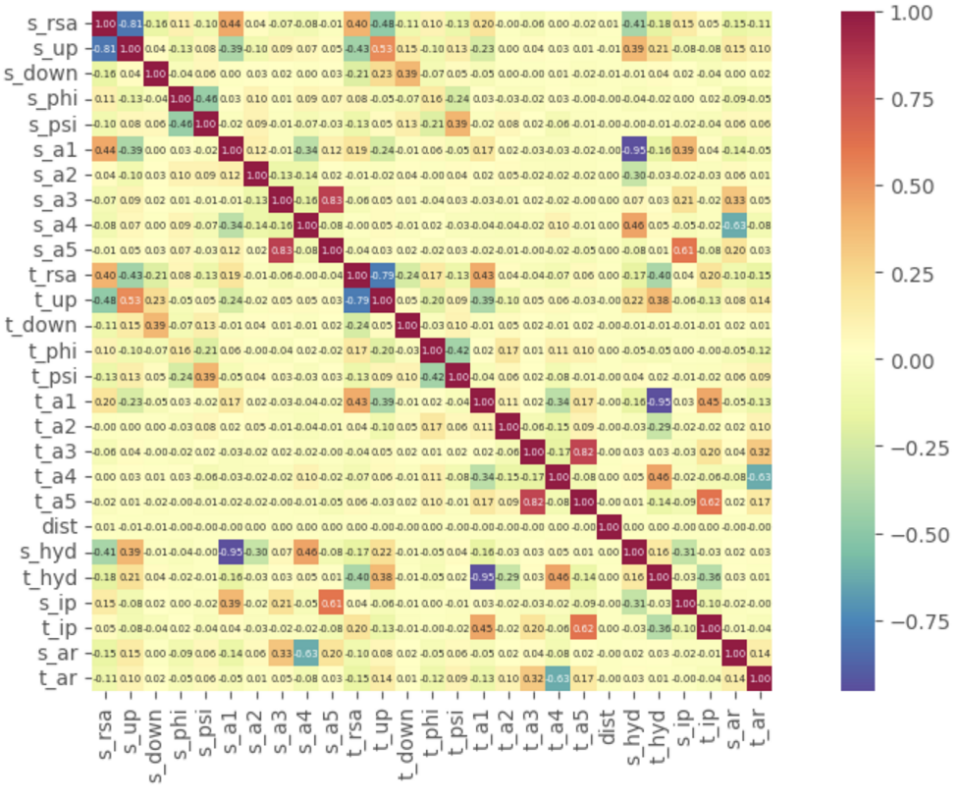


Figure 2. Correlation matrix with all features

Box Plots (Numeric features vs Target feature)

Box plots were generated for each numeric feature with respect to each class and are given in Figure 3.. In the plot for the rsa feature, we may observe various values for the median per class, such that the VDW and PIPISTACK classes also have a lot of outliers. The phi angles values are having nearly the same median across all classes, such that more outliers are spotted in HBOND and VDW. For the psi angle and the Atchley's features, different median values are spotted across classes. SSBOND class has the same value for the Atchley features and the hydrophobicity in almost all instances. The distance values we have calculated have all similar median values for each class. Regarding the isoelectric point distributions of instance values are concentrated mainly around one value, except for the IONIC and PICATION classes, which is expected since with this feature we aim to represent the charge of the residues and aid in classifying ionic bonds mainly.

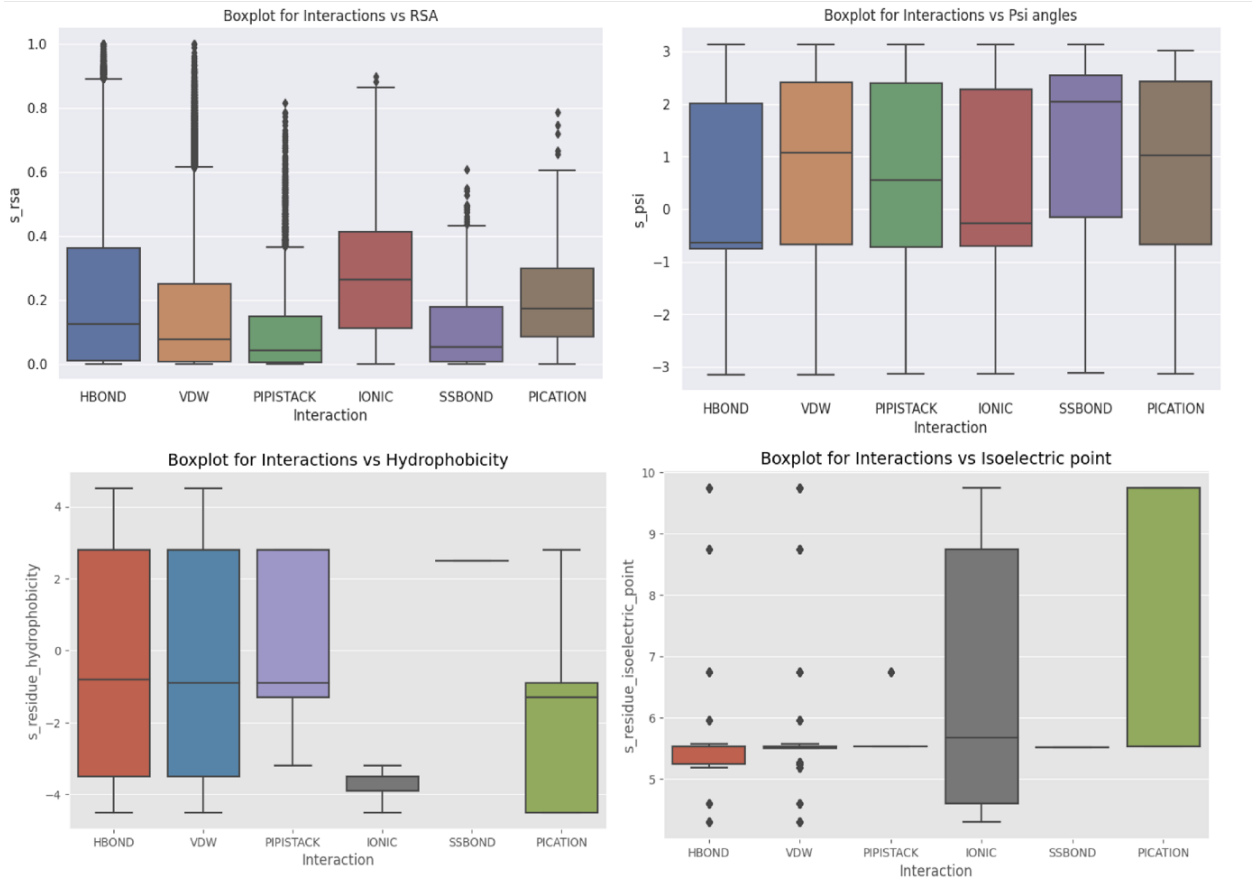


Figure 3. Boxplots

MODELS AND OVERSAMPLING STRATEGY

Deep Neural Network

Before inputting the data into a DNN model, we used LabelEncoder from sklearn to encode the categorical values for the DSSP and SS 3 features. Afterwards, in order to transform the dataset into multi labeled, we used one-hot encoding by representing the labels for each class with binary vectors. (ex. [100010 - indicate contact belongs to class HBOND and SSBOND). For this purpose, the dataset was grouped by the identification features for the source and target residues. We used relu activation functions for each hidden layer, binary-cross entropy loss and adam optimizer. Last layer activation is sigmoid. The sigmoid activation function produces outputs in the range of [0, 1] for each class, representing the probability of that class being present. The binary cross-entropy loss compares these probabilities with the true labels for each class independently. Moreover, accuracy, precision and recall are monitored during model training and early

stopping mechanism has been set.

Oversampling - MLSMOTE

For oversampling of multi-label dataset we used the so-called MLSMOTE algorithm (https://github.com/niteshsukhwani/MLSMOTE) for synthetic oversampling of instances belonging to multiple classes. In simple words, this algorithm calculates so-called tail-labels, that are the labels such that the imbalance ratio per label is greater than the mean imbalance ratio: IRPL(l) > MIR. The set of all the instances of the data which contain that label is considered as minority instance data. In our case it contains 6 tail-labels and 1293 samples in the minority instance. By oversampling, new instances of the minority dataset are synthetically generated, such that the ratio within classes in minority instances is preserved. The number of synthetized instances is 10% of the training dataset length.

OneVsRest models

We have also tried to decompose the multi-label problem into multiple independent binary classification problems (one per category). For that purpose we applied the "OneVSRest" strategy, just to see how well six different independent classifiers would learn from the features that we have. One type of classifiers that we exploited is Multilayer perceptron (MLP). The obtained results from each of the six MLP classifiers is presented in the following table.

Performance metrics	Matthew Correlation coefficient	Balanced Accuracy	Average Precision Score	AUC ROC
HBOND MLP Classifier	**0.46**	**0.71**	**0.80**	**0.71**
IONIC MLP Classifier	**0.56**	**0.73**	**0.34**	**0.73**
PICATION MLP Classifier	**-0.00**	**0.49**	**0.00**	**0.49**
PIPISTACK MLP Classifier	**0.80**	**0.94**	**0.65**	**0.94**
SSBOND MLP Classifier	**0.88**	**0.99**	**0.78**	**0.99**

VDW MLP Classifier	**0.43**	**0.71**	**0.58**	**0.71**

As we can notice from the results, the features that we have are somewhat good for teaching a model how to predict the PIPISTACK and SSBOND classes. But, the results are very bad for predicting the PICATION class, i.e. the trained model for that class always returns the same output.

Random Forest

When training a ML model it is always useful to have some understanding of what happens "behind the scenes". But most of the models are black-box. So, we decided to make use of a Random Forest classifier to see which of the features are most important. As we can see in Figure 4, the psi, phi angles and the Atchley feature 1 are the most

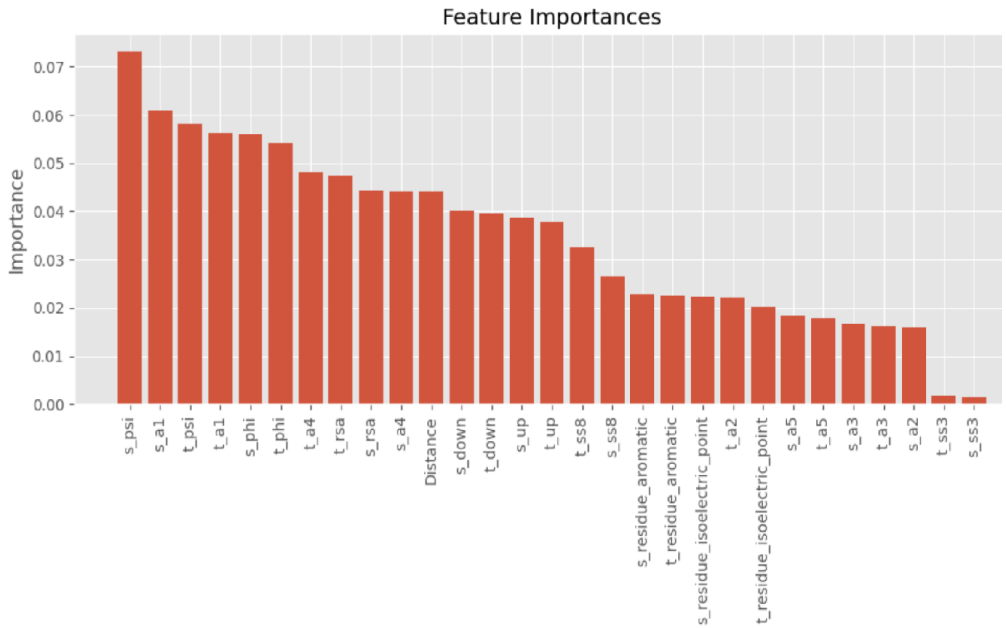


Figure 4. Feature importances from RF model

informative ones. On the other hand, secondary structure 3 states (ss3) features are the least important ones.

RESULTS OF FINAL MODEL

Since we decided to continue with the DNN model as the most successful one, at first, we tried with one hidden layer with 20 neurons in the model, trained for 30 epochs and 20% validation split. The model had nearly the same performance with and without the

additional features in the dataset. Performance has improved after introducing oversampling to our dataset, more layers with more neurons in our model and more epochs for training (up to 100 epochs). Model evaluation is carried out using 10-fold cross-validation, such that oversampling is performed for each train set split separately, so that the test set remains with the original class distribution. Metrics results for data with the additional features is slightly better than the one without them, as it may be observed from the results table below.

Performance metrics	DNN Model without additional features	DNN Model with all features
Matthew Correlation coefficient	**0.715 (0.004)**	**0.721 (0.003)**
Balanced Accuracy	**0.863 (0.002)**	**0.864 (0.003)**
Average Precision Score	**0.637 (0.004)**	**0.643 (0.004)**
AUC ROC	**0.863 (0.002)**	**0.864 (0.003)**
Precision	**0.758 (0.002)**	**0.766 (0.005)**
Recall	**0.786 (0.004)**	**0.785 (0.007)**

Plots for the metrics during model training are given in Figure 4, and we may see they follow the usual trend, meaning loss decreasing and accuracy precision and recall increasing up to values of around 0.7 and 0.8.

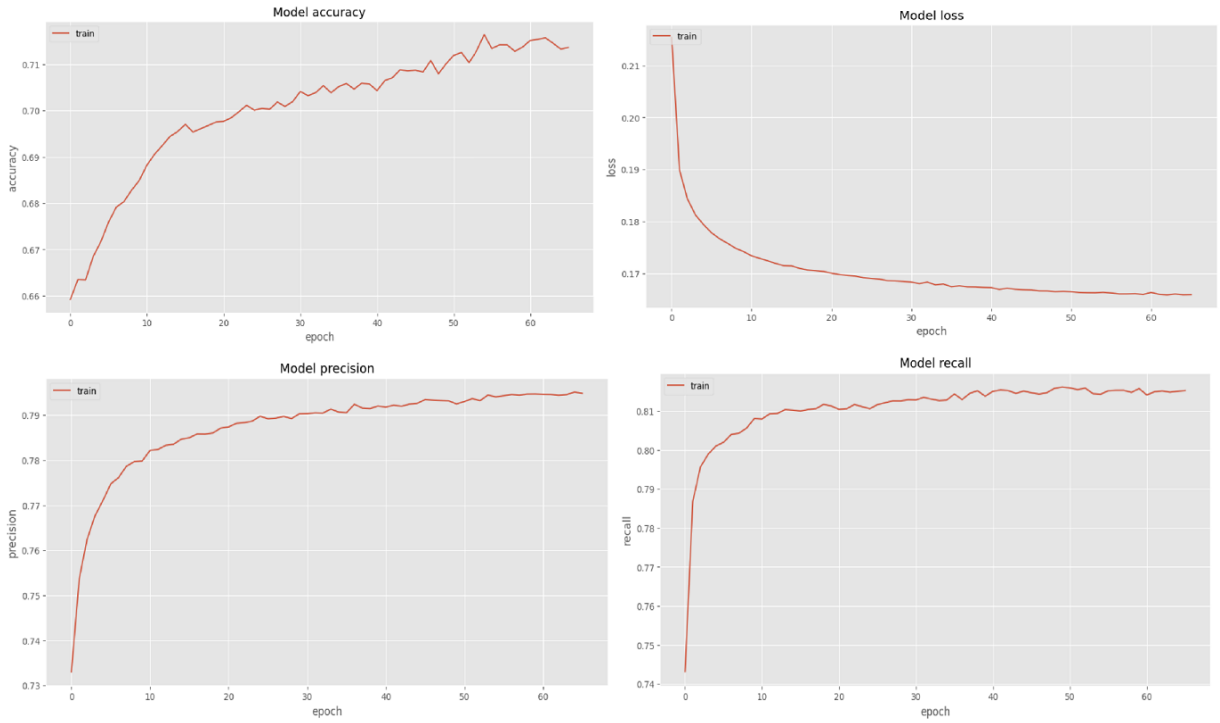


Figure 5. Training Metrics

ILLUSTRATING THE MODEL CORRECTNESS

We also wanted to see how well the model performs when given a whole pdb file that it has never seen before, and to compare the results with the ones obtained from the RING software. For that purpose we chose the Spike protein with PDB ID “2GHV”. Results for the number of each type of contacts obtained from our model VS the RING software are represented in the table below.

Contact type	Number of contacts predicted by our model	Number of contacts predicted by RING
HBOND	**277**	**182**
IONIC	**6**	**7**
PICATION	**2**	**0**

PIPISTACK	**28**	**32**
SSBOND	**7**	**4**
VDW	**217**	**306**

We can see that for some of the classes the model is very close to predicting the same number of contacts as the RING software.

CONCLUSION

In this project, we developed software to predict the RING classification of protein contacts using supervised methods for multi-labeled classification. By exploring different models and architectures, extensive data analysis, utilizing additional features and addressing class imbalance by oversampling, we achieved promising results in terms of MCC, balanced accuracy and average precision. This project showcased the effectiveness of incorporating diverse features and models to classify protein contacts while handling challenges of data imbalance.