RING Classification of Residue-Residue Contacts in Protein Structures

Anahita Soltantouyeh[a], **Mikael Poli**[b] **and Biddut Bhowmik**[c]

[a]*anahita.soltantouyeh@studenti.unipd.it*
[b]*mikael.poli@studenti.unipd.it*
[c]*biddut.bhowmik@studenti.unipd.it*

1. Introduction

Protein structures are stabilized by various residue-residue interactions, including hydrogen bonds, van der Waals contacts, disulfide bridges, salt bridges, and π-π interactions. Accurately classifying these contacts is essential for understanding protein folding, stability, and function.

Traditionally, software like RING [1] identifies and classifies residue-residue contacts in protein structures using geometric and physicochemical rules. In this project, we experiment with a machine learning-based approach to predict the type of contacts without relying directly on geometric thresholds.

2. Objective

Our goal is to develop a classification model that predicts the RING type of residue-residue contacts in a PDB structure using a supervised approach. The model is trained using features derived from both sequence and structural data, such as secondary structure, solvent accessibility, half-sphere exposure, Atchley scales, and 3Di alphabet encoding. We aim to outperform a random baseline across multiple evaluation metrics, including balanced accuracy, AUC-ROC, Matthews correlation coefficient (MCC), and average precision.

We define the "random baseline" as a model that makes uniform guesses across all classes. For a classification task with K classes, the expected performance of such a random predictor is approximately:

- *Balanced Accuracy:* $\approx \frac{1}{K}$ (for $K = 8$, this yields $\frac{1}{8} \approx 0.125$). Note: Using `sklearn`'s `balanced_accuracy_score` with `adjusted=True`, the chance level is set to 0.
- *AUC-ROC:* ≈ 0.5.
- *Matthews Correlation Coefficient (MCC):* ≈ 0.
- *Average Precision:* $\approx \frac{1}{K} = \frac{1}{8} = 0.125$.

3. Dataset

The dataset comprises examples categorized into eight distinct classes, with their distribution illustrated in Figure 2.

Contact type	Count
HBOND	1,055,929
VDW	737,061
PIPISTACK	38,283
IONIC	35,391
PICATION	8,885
SSBOND	2,100
PIHBOND	1,790
Unclassified	1,089,547

Figure 2. Distribution of RING types in the original dataset.

Each example consists of the features detailed in Table 1.

Features		
Source	**Target**	**Description**
s_ss8	t_ss8	Secondary structure 8 states (DSSP)
s_rsa	t_rsa	Relative solvent accessibility
s_phi	t_phi	Phi angle
s_psi	t_psi	Psi angle
s_a1	t_a1	Atchley feature 1
s_a2	t_a2	Atchley feature 2
s_a3	t_a3	Atchley feature 3
s_a4	t_a4	Atchley feature 4
s_a5	t_a5	Atchley feature 5
s_3di_state	t_3di_state	3Di state
s_3di_letter	t_3di_letter	3Di alphabet

Table 1. Feature names and their descriptions.

We visualized all features to get a sense of their distributions among classes. We report in this section an example of one continuous (Figure 1) and one categorical variable (Figure 3). Plots for all variables are available in the project's repository (see the Software section at the end of the report).

4. Methods

Model training was conducted in Python 3.11.

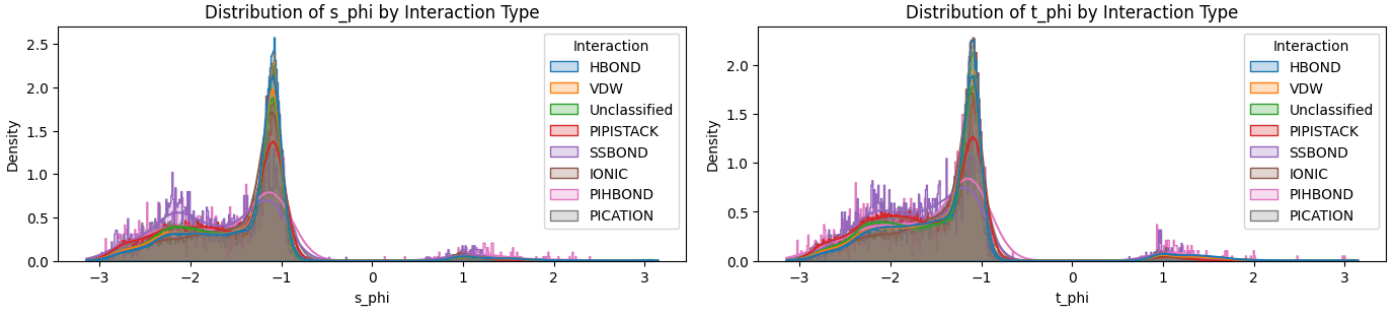

Figure 1. Phi angle distribution for the source (s_) and target (t_) residues.

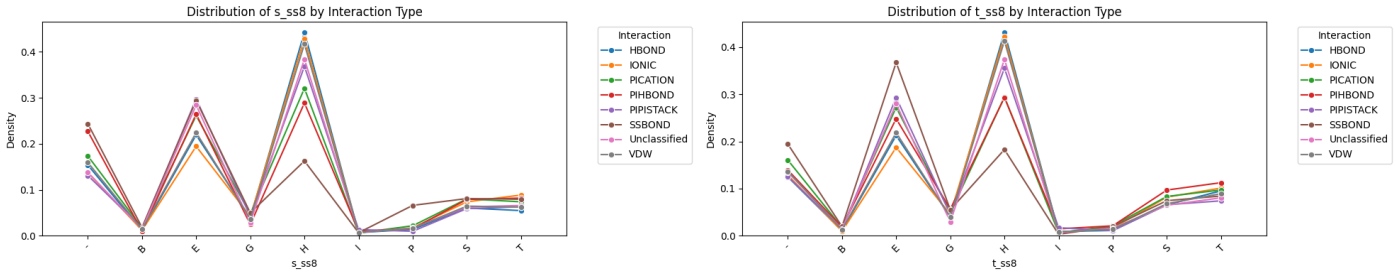

Figure 3. DSSP state distribution for the source (s_) and target (t_) residues.

4.1. Data Preprocessing

We began by merging the provided .tsv files, which included precomputed structural and functional features as well as contact RING labels, into a single dataframe. Any instances labeled as "NA" were assigned to a new category labeled "Unclassified". We excluded all rows containing missing feature values from the dataset. We used all 22 features for training.

Next, we split the data into three subsets. First, we set aside 10% of the data as the test set. The remaining 90% was used for training and validation, which we further divided into 80% for training and 20% for validation.

Due to significant class imbalance, we applied a resampling strategy to the training set, oversampling the minority classes and undersampling the majority classes. The final sizes of the train, validation, and test sets are summarized in Table 2.

Interaction	Train (O.)	Train (R.)	Val.	Test
Unclassified	764,634	100,000	191,159	106,199
HBOND	739,818	100,000	184,954	102,753
VDW	516,198	100,000	129,050	71,694
PIPISTACK	26,888	30,000	6,722	3,734
IONIC	24,700	30,000	6,175	3,431
PICATION	6,174	30,000	1,544	858
SSBOND	1,453	30,000	363	202
PIHBOND	1,242	30,000	310	172
Tot.	2,081,107	450,000	520,277	289,043

Table 2. Dataset composition by interaction type across training (O. = original, R. = resampled) training, validation, and test sets.

4.2. Model

We trained eight binary XGBoost [2] classifiers in a One-vs-Rest (OvR) setup, with one model dedicated to each RING interaction type, using `xgboost.train()`. Each model was trained with the following parameters:

```
"model_parameters": {
    "device": "cuda",
    "objective": "binary:logistic",
    "eval_metric": "logloss",
    "max_depth": 10,
    "learning_rate": 0.2
},
"early_stopping_rounds": 15,
"num_boost_round": 2000,
```

To predict the interaction type for each test example, we use all eight models to compute class probabilities and assign the label corresponding to the highest predicted probability.

We chose XGBoost because of the balance it offers between scalability and accuracy, especially for structured data with mixed continuous and categorical features.

5. Results

Training took in total ≈ 20 seconds (wall time) using Google Colab's T4 GPU.

The ensemble's accuracy scores and confusion matrix are reported in Table 3 and Figure 4 respectively.

Metric	Value
Balanced Accuracy	0.622
AUC-ROC	0.923
Matthews Correlation Coefficient	0.231
Average Precision	0.355

Table 3. Performance metrics for the XGBoost OvR ensemble.

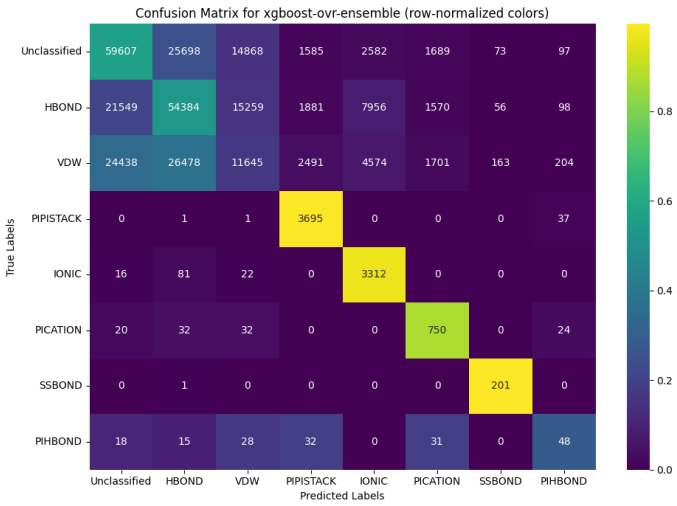

Figure 4. Confusion matrix for the XGBoost OvR ensemble.

All metrics were above the desired baseline.

6. Discussion

While all accuracy scores met our performance goals, the ensemble model particularly struggled to distinguish between the majority classes, i.e., HBOND, VDW, and Unclassified interactions. Among the minority classes, PIPISTACK interactions posed the greatest challenge, although the model still correctly classified most of them overall. This difficulty may be due to the similarity in feature distributions across classes, especially among continuous variables. This is one of the reasons why we chose a model capable of natively handling datasets with a mix of continuous and categorical features. Additionally, the resampling strategy likely influenced performance.

We tested various resampling approaches. For instance, increasing the number of VDW samples improved VDW classification but led to poorer performance on the other majority classes, which were

frequently misclassified as VDW. Ultimately, we selected the resampling strategy that yielded the best overall results, likely because it maintained a relatively balanced 1:3 ratio between minority and majority classes. Further oversampling — especially of the SSBOND and PIHBOND classes — tended to distort the class distributions and degrade model performance.

Additionally, while using a relatively larger validation set can lead to more stable performance estimates, it may also reduce model accuracy. This trade-off is another consequence of the resampling strategy we employed.

We also experimented with different learning rates and early stopping thresholds before selecting the final configuration used in the ensemble.

In addition to the ensemble, we trained a single multiclass XGBoost model using the same hyperparameters, except for the `objective` and `eval_metric`, which were set to `multi:softprob` and `mlogloss`, respectively. This model showed the same general classification trends as the ensemble but performed slightly worse, with a balanced accuracy of 0.605, AUC-ROC of 0.919, MCC of 0.219, and average precision of 0.336.

Overall, the main challenges this classification task posed are the non-linear separability of the features and the extreme class imbalance. The models' difficulty in distinguishing hydrogen bonds from van der Waals interactions appears to reflect the physical and chemical similarities between these two types of electrostatic contacts.

7. Software

The code, data, figures, and documentation for the project can be found on GitHub.

References

[1] A. Del Conte, G. F. Camagni, D. Clementel, *et al.*, "RING 4.0: faster residue interaction networks with novel interaction types across over 35,000 different chemical structures", *Nucleic Acids Research*, vol. 52, no. W1, W306–W312, Apr. 2024. DOI: 10.1093/nar/gkae337. [Online]. Available: https://doi.org/10.1093/nar/gkae337.

[2] 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*, ser. KDD '16, San Francisco, California, USA: Association for Computing Machinery, 2016, pp. 785–794, ISBN: 9781450342322. DOI: 10.1145/2939672.2939785. [Online]. Available: https://doi.org/10.1145/2939672.2939785.