Classification of contacts in protein structures via ML models

Davide Baggio
davide.baggio.1@studenti.unipd.it

Sebastiano Sanson
sebastiano.sanson@studenti.unipd.it

Abstract

Predicting the type of inter-residue contact in a protein structure is important for understanding its interaction network. In this work we frame the RING contact classification task as a supervised learning problem, using gradient-boosted trees (XGBoost) to predict each contact's RING-defined type from structural features. We train on a large dataset of residue-residue contacts (from 3,914 PDBs) labeled by RING (types like hydrogen bond, van der Waals, salt bridge, etc.) and evaluate performance via metrics suited for imbalanced multiclass data (Matthews correlation coefficient, balanced accuracy, ROC-AUC, average precision). Two approaches are compared: a single multiclass XGBoost model (softmax objective) and an ensemble of one-vs-all binary XGBoost classifiers (logistic objective). The one-vs-all ensemble attains slightly higher accuracy, while the multiclass model yields higher average precision on common contact classes. Rare contact types (e.g. disulfide bridges) are often predicted with very high ROC-AUC, while classes like "Unclassified" remain challenging. Overall, XGBoost proves to be a viable classifier for this task given the extracted structural features.

1. Introduction

Residue Interaction Networks (RINs) represent a protein's 3D structure as a graph of amino-acid contacts, capturing non-covalent interactions based on geometrical and physicochemical criteria. The RING software identifies such contacts from a PDB structure and assigns each a type (e.g. hydrogen bond, van der Waals, salt bridge, π-stacking, etc.). Traditionally, RING's classification relies on geometric rules; here we instead train a data-driven model to predict a contact's RING type from features of the interacting residues. Predicting contact types by statistical methods can help validate and complement physics-based annotations. In this study we use XGBoost (gradient-boosted trees) to classify contact types, aiming to reproduce RING's labeling from structure-derived features. We leverage the provided training data (features extracted by Biopython and 3Di encoding scripts) and compare a multiclass classification approach to a one-vs-all scheme.

2. Data Source

The dataset comprises a collection of example from 3,914 PDBs, containing one line per residue–residue contact. In total the dataset includes on the order of 3×10^6 contacts, with a highly imbalanced distribution of RING types (e.g. $\sim 1.06 \times 10^6$ hydrogen bonds, $\sim 1.09 \times 10^6$ "Unclassified" contacts, but only a few thousand disulfide bonds). Each record lists identifiers of the two residues and a variety of structural features for each "source" and "target" residue, plus the RING interaction type label (Table).

Contact	Count
HBOND	1,055,929
VDW	737,0610
PIPISTACK	38,283
SSBOND	35,391
CATIONPI	8,885
IONIC	2,100
HYDROPHOBIC	1,790
Unclassified	1,089,547

Table 1: Distribution of contact types in the dataset. The "Unclassified" type is the most common, followed by hydrogen bonds and van der Waals interactions. The other types are much rarer, with only a few thousand examples each.

2.1. Data Preprocessing

The raw feature files were merged into a single DataFrame. We first handled missing and duplicate data: any contact with a missing Interaction label was relabeled as "Unclassified". Exact duplicate rows were removed (keeping one copy) to avoid over-counting (the notebooks sorted by class frequency and dropped duplicates). For the numeric features (phi, psi angles, RSA, etc.), missing values were filled by the column mean. After this imputation

there were no remaining NaNs. Categorical features were encoded as integers: in particular, the DSSP secondary-structure state (s_ss8 and t_ss8) was label-encoded. (The 3Di alphabet letters were dropped in favor of the 3Di state index.)

2.2. Feature Engineering

For each contact (residue pair) we used structural descriptors of both the source and target residues. These include:

- **Secondary structure (DSSP)**: 8-state labels for each residue (columns s_ss8, t_ss8), later encoded as integers.
- **Solvent accessibility (RSA)**: relative solvent-accessibility values (s_rsa, t_rsa).
- **Backbone dihedral angles**: phi and psi angles for each residue (s_phi, s_psi, t_phi, t_psi).
- **Atchley factors**: five physiochemical factor scores per residue ($s\_a1 - s\_a5$, $[t\_a1, t\_a5]$) .
- **3Di structural state**: the 3Di alphabet state index for each residue (s_3di_state, t_3di_state).

Altogether this yields a feature vector capturing both the local secondary-structural context and physico-chemical properties of the residue pair.

2.3. Balancing the Dataset

Due to the highly imbalanced nature of the dataset (with some classes like "Unclassified" dominating), we applied class balancing techniques. SMOTE (Synthetic Minority Over-sampling Technique) was used to generate synthetic examples for the minority classes, ensuring that each class had a more balanced representation in the training set. This helps prevent the model from being biased towards the majority class and improves its ability to generalize across all contact types. We also used class weights in the XGBoost model to further mitigate the imbalance during training.

3. Approaches

We implemented two strategies using XGBoost:

Multiclass XGBoost. A single XGBoost model was trained with the objective `multi:softprob` for 8-way classification (7 contact classes plus "Unclassified"). We used class-balanced training by providing sample weights (`compute_sample_weight('balanced')`) to the DMatrix (to mitigate class imbalance). Key hyperparameters (selected via some manual tuning) included a maximum tree depth of 8, learning rate 0.03, subsample 0.85, and regularization terms ($\gamma = 0.2$, $\alpha = 0.5$, $\lambda = 1.5$, etc). Early stopping on validation log-loss was applied to prevent overfitting. Predictions are the class with maximum predicted probability.

One-vs-All Ensemble. In parallel, we trained eight binary XGBoost classifiers, one per class. For class i we labeled all training examples of type i as positive (1) and all others as negative (0). Each binary model used the `binary:logistic` objective. We adjusted for the imbalance by setting the `scale_pos_weight = (negatives/positives)` for each class. All binary models shared most hyperparameters (max_depth, learning rate, etc.), with evaluation metrics set to AUC and average precision. At prediction time, the eight binary models output one probability each; the final predicted class is taken as the class with highest probability.

Comparison and Tuning. Both methods used a fixed train/validation split (or cross-validation) from the provided data. We did not perform an extensive automated search for hyperparameters, but rather adjusted a few parameters (tree depth, regularization) to optimize validation MCC. Both approaches were implemented in the provided notebooks, and their parameters and training procedures were as shown in the code .

4. Model Evaluation Metrics

We evaluated performance using metrics that account for class imbalance and multi-class scoring. As specified in the project requirements, we computed:

- **Balanced accuracy**: the average of recall (true positive rate) over all classes.
- **Matthews Correlation Coefficient (MCC)**: a correlation coefficient between true and predicted labels (a balanced measure even for imbalanced data).
- **ROC-AUC**: area under the Receiver Operating Characteristic curve, extended to the multiclass setting (one-vs-rest average).
- **Average Precision (AP)**: area under the precision-recall curve, indicative of performance on positive class detection.

These metrics were explicitly noted as evaluation criteria in the project specification and are standard for multiclass classification. Higher values (up to 1.0) indicate better performance; a random predictor would score near 0.5 (AUC) or 0 (MCC/AP) in expectation.

5. Testing and results analysis

After training, we evaluated both models on held-out test data (from the provided set). The multiclass XGBoost model achieved an overall **Balanced Accuracy** of about 0.721 and **Matthews correlation** 0.207 on test data. (Overall accuracy was ∼0.426.) The one-vs-all ensemble performed similarly: Balanced Accuracy 0.718 and MCC0.203 . Thus both methods substantially outperformed random chance, but the numeric values indicate only moderate predictive power. The one-vs-all approach yielded a slightly higher raw accuracy (0.484 vs 0.426) but similar balanced accuracy, suggesting it was better at predicting majority classes while still matching multiclass recall performance. Conversely, the multiclass model produced a higher **macro-average AP** (0.415) than the ensemble (0.325), indicating better precision on the more common classes.

Examining per-class performance (OVA ensemble): some contact types were predicted extremely well in terms of ranking. For example, the rare *SSBOND* (disulfide) and *PIPISTACK* classes each had ROC-AUC ≈ 0.99 and AP around 0.40 (0.4034 for SSBOND) . Common classes like *HBOND* and *VDW* had lower AUC (0.66 and 0.63, respectively) and moderate AP (0.69 and 0.55) . The "Unclassified" class was effectively never predicted (AP0), as the models favored assigning one of the specific types. In summary, the models learned to distinguish well-defined interaction types (especially those with distinctive structural signatures) but struggled to identify the generic/unclassified interactions. The confusion matrices (not shown) confirmed that many contacts were misassigned among the major classes, consistent with the modest MCC values.

6. Conclusions

In conclusion, gradient-boosted trees (XGBoost) can learn to predict residue–residue contact types from structural features with performance well above random, confirming that the features (DSSP, RSA, angles, Atchley, 3Di) carry information about interaction class. Both a single multiclass model and a one-vs-all ensemble achieved similar results (Balanced Acc ≈ 0.72, MCC ≈ 0.20) in this dataset. The one-vs-all method had slightly higher accuracy, while the multiclass model achieved higher average precision on the main classes. Overall, XGBoost shows effectiveness for this task, though predicting the abundant "Unclassified" contacts remains difficult. Future work could explore additional features or model architectures (e.g. graph neural networks) to improve classification of borderline contacts.