Indraprastha Institute of Information Technology Delhi Okhla Phase-III, New Delhi-110020

IP Report Summer - 2024

ML/DL-based models for predicting protein/peptide thermal stability.

SUBMITTED TO

<u>PROF. N. Arul. Murugan</u>

July 2024

BY

Umang Sharma (MT23239) &

Harnoor Kaur Anand (MT23229)

Table of Contents

Predicting Melting Temperature of Proteins using 3D Convolutional Neural Networks

1. Introduction

1.1 Objective:

The objective of this project is to develop machine learning and deep learning models to predict the melting temperature (Tm) of proteins, which is a critical indicator of their thermal stability. Accurate prediction of protein Tm has significant implications in fields such as drug design, protein engineering, and understanding disease mechanisms. Specifically, the project aims to:

1. Utilise 3D Convolutional Neural Networks (CNNs) to predict Tm based on 3D structural data of proteins.
2. Explore the potential of Graph Neural Networks (GNNs) in predicting Tm by leveraging the inherent graph structure of protein molecules.
3. Implement traditional machine learning approaches, such as Random Forest, to predict Tm using sequence-derived features and physicochemical properties of proteins.

By employing a combination of these advanced computational methods, the project seeks to create robust predictive models that can aid in the efficient and accurate determination of protein stability, thereby facilitating advancements in related scientific and industrial domains.

1.2 Background:

Proteins are complex molecules that play vital roles in virtually all biological processes, including catalysis, signaling, structural support, and immune responses. Understanding their stability is crucial for various applications in biotechnology and medicine, such as drug design, protein engineering, and the elucidation of disease mechanisms. The melting temperature (Tm) of a protein is a key measure of its thermal stability, representing the temperature at which the protein denatures or unfolds. Traditional experimental methods to determine Tm, such as differential scanning calorimetry (DSC), are often time-consuming and costly. Therefore, computational approaches that can accurately predict Tm from protein structures are highly desirable.

1.2.1 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) have revolutionized the field of computer vision by effectively capturing spatial hierarchies in data. CNNs are designed to automatically and adaptively learn spatial hierarchies of features from input images or volumetric data. The key components of CNNs include:

1. **Convolutional Layers**: These layers apply convolutional filters to the input data, extracting local features such as edges, textures, and patterns. In 3D CNNs, 3D filters are used to scan through volumetric data, making them suitable for analyzing 3D protein structures.
2. **Pooling Layers**: Pooling layers reduce the spatial dimensions of the data by aggregating features, typically using max-pooling or average-pooling. This helps in reducing the computational complexity and the risk of overfitting.
3. **Fully Connected Layers**: These layers are used at the end of the network to integrate the features learned by the convolutional and pooling layers and make final predictions.
4. **Activation Functions**: Non-linear activation functions, such as ReLU, are applied to introduce non-linearity into the model, enabling it to learn complex patterns.

By extending CNNs to three dimensions, the project aims to leverage their capability to analyze volumetric data, making them suitable for predicting the properties of 3D protein structures. This project utilizes 3D CNNs to predict the melting temperature of proteins from their 3D structural data obtained from Protein Data Bank (PDB) files.

1.2.2 Graph Neural Networks (GNNs)

Graph Neural Networks (GNNs) are a class of neural networks designed to operate on graph-structured data. Proteins can be naturally represented as graphs, where atoms or residues serve as nodes and bonds or interactions as edges. GNNs are capable of capturing the complex relationships and interactions within these graph structures, making them suitable for predicting various properties of proteins, including their thermal stability. The key components of GNNs include:

1. **Graph Convolutional Layers**: These layers aggregate features from neighboring nodes, allowing the network to learn local and global patterns within the graph. This is analogous to the way convolutional layers in CNNs aggregate features from local neighborhoods in images.
2. **Message Passing Mechanism**: GNNs use a message-passing mechanism where nodes update their features by exchanging information with their neighbors. This process allows the network to capture the dependencies and interactions between different parts of the graph.

3. **Pooling and Readout Layers**: After several graph convolutional layers, pooling or readout layers are used to aggregate the features from all nodes into a fixed-size representation, which can be used for downstream tasks such as classification or regression.

This project explores the potential of GNNs in predicting Tm by leveraging the inherent graph structure of protein molecules. However, due to computational constraints, the GNN-based approach is considered for future research.

1.2.3 Random Forest

Random Forest is a versatile machine-learning algorithm that is widely used for both classification and regression tasks. It operates by constructing a multitude of decision trees during training and outputting the mean prediction (regression) or majority vote (classification) of the individual trees. The key components of Random Forest include:

1. **Ensemble Learning**: Random Forest is an ensemble method that combines multiple decision trees to improve the model's accuracy and robustness. Each tree in the forest is trained on a random subset of the data, which helps in reducing overfitting and increasing generalisation.
2. **Feature Randomness**: In addition to using random subsets of the data, Random Forest introduces randomness in the feature selection process. Each node in a tree is split using a random subset of features, which helps in creating diverse trees and reducing the correlation between them.
3. **Bootstrap Aggregating (Bagging)**: Random Forest uses a technique called bagging, where multiple versions of the dataset are created by sampling with replacement. Each tree is trained on a different version of the dataset, which helps in reducing variance and improving model stability.

This project implements Random Forest to predict Tm using sequence-derived features and physicochemical properties of proteins. The model's ability to handle a large number of input features and its robustness against overfitting make it a suitable choice for this task.

1.2.4 Integration of Approaches

By employing a combination of these advanced computational methods, the project seeks to create robust predictive models that can aid in the efficient and accurate determination of protein stability. The integration of 3D CNNs, GNNs, and Random Forest allows the project to leverage the strengths of each approach, providing a comprehensive solution to the problem of predicting protein melting temperatures. This multidisciplinary approach not only enhances the predictive accuracy but also offers valuable insights into the factors influencing protein stability, facilitating advancements in related scientific and industrial domains.

3D CNN

2. Data Collection and Preprocessing

2.1 Data Sources:

The primary data source for this project is the Protein Data Bank (PDB), which provides detailed 3D structural data of proteins. Additionally, experimental melting temperatures for these proteins were sourced from relevant biochemical databases and literature.

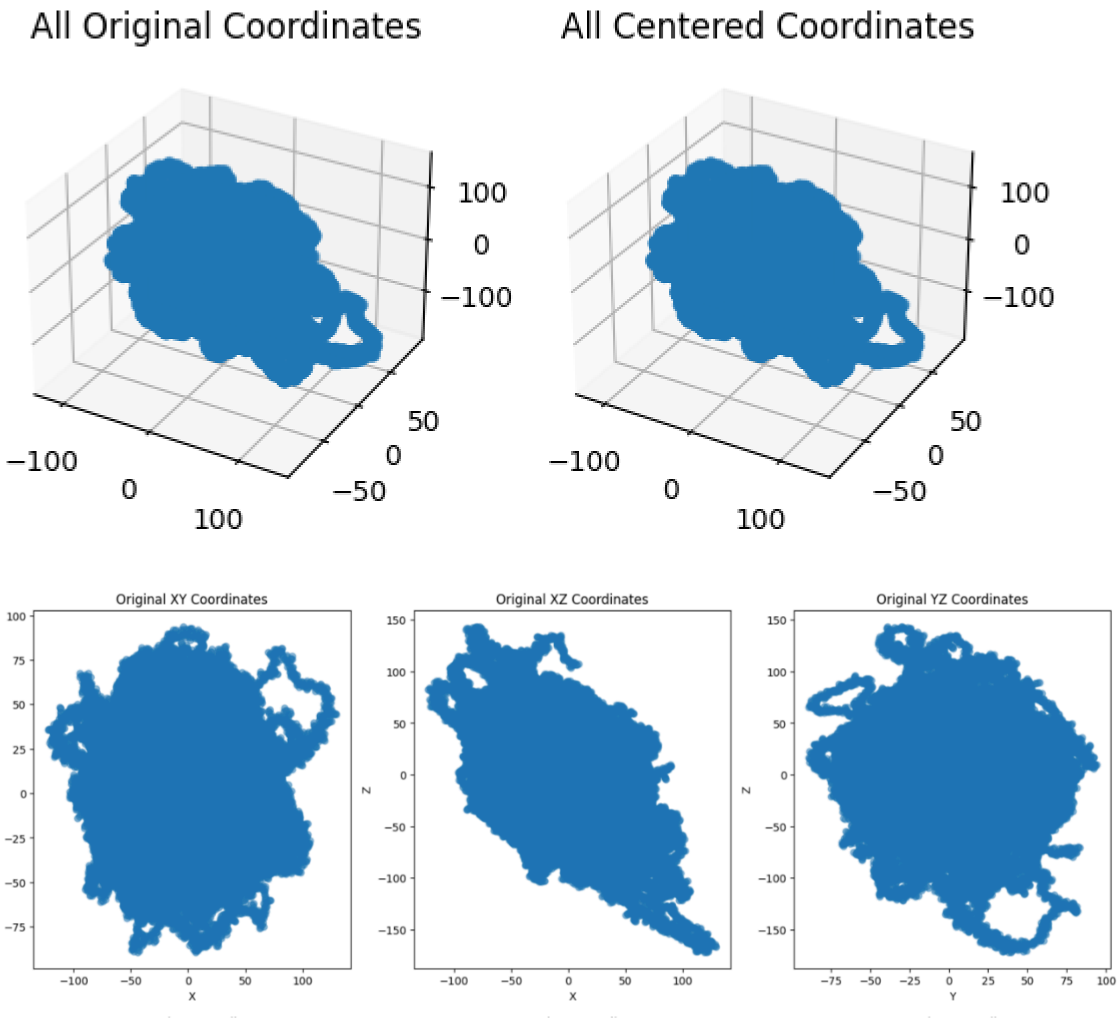
2.2 Preprocessing Steps:

2.2.1 Reading PDB Files:

PDB files, which contain atomic coordinates of proteins, were read using Biopython and Biopandas libraries. These coordinates represent the 3D structure of the protein.

2.2.2 Centering Coordinates:

The protein coordinates were centered based on their center of mass to ensure a consistent reference point for all proteins. This step involved calculating the geometric center of the protein and adjusting all coordinates accordingly.



Description: This figure illustrates the coordinates of a protein structure before and after centering. The original coordinates are shown on the left, while the centred coordinates are depicted on the right.

2.2.3. Grid Representation:

The centred coordinates were then mapped onto a 3D grid to create a fixed-size representation suitable for input into the 3D CNN. Each voxel in the grid represents the presence or absence of atoms within a specific volume of space.

2.2.4. Feature Extraction:

Additional features such as atomic number, residue type, and secondary structure information were extracted and encoded to enrich the input data.

Challenges in Preprocessing:

Handling the diversity in protein sizes and shapes required careful consideration to standardize the input representations. Moreover, ensuring that the grid resolution was sufficiently fine to capture detailed structural information without resulting in excessively large input tensors was a key challenge.

3. Model Development

Model Architecture:

The 3D CNN architecture developed for this project consists of multiple convolutional layers interspersed with pooling layers to reduce the spatial dimensions and extract hierarchical features progressively. The architecture is summarized as follows:

Input Layer: Takes the 3D grid representation of the protein structure as input.

Convolutional Layers: Multiple Conv3D layers with ReLU activation to capture spatial features. These layers use 3x3x3 kernels to scan through the 3D volume.

Pooling Layers: MaxPooling3D layers to downsample the feature maps and reduce dimensionality, thus preventing overfitting and improving computational efficiency.

Dense Layers: Fully connected layers to integrate features extracted by the convolutional layers. These layers use dropout regularization to mitigate overfitting.

Output Layer: A final dense layer with a single neuron to predict the melting temperature, using a linear activation function to produce a continuous output.

3.1 Training Setup: The training process involved splitting the data into training and validation sets. Key components of the training setup include:

Loss Function: Mean Absolute Error (MAE) was used as the loss function to quantify the difference between predicted and actual Tm values.

Optimizer: The Adam optimizer was selected for its adaptive learning rate properties, enhancing convergence speed and stability.

Early Stopping and Model Checkpointing: Early stopping was employed to halt training when the validation loss ceased to improve, and the best model weights were saved using model checkpointing.

Model Training and Hyperparameter Tuning:Various hyperparameters such as learning rate, batch size, number of epochs, and the architecture of the CNN were tuned using cross-validation and grid search techniques to optimize model performance.

4. Results and Evaluation

Training and Validation Performance:

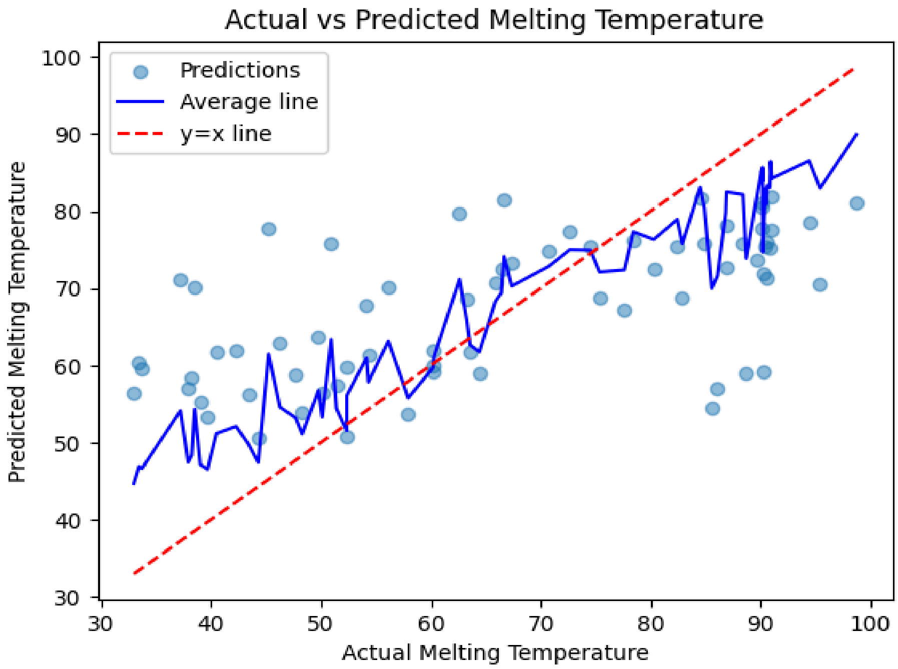
The model's performance was evaluated using several metrics:

Mean Absolute Error (MAE): The average absolute difference between predicted and actual Tm values.

R-squared (R2): A statistical measure indicating how well the regression predictions approximate the real data points.

Pearson Correlation Coefficient: Measures the linear correlation between predicted and actual Tm values.

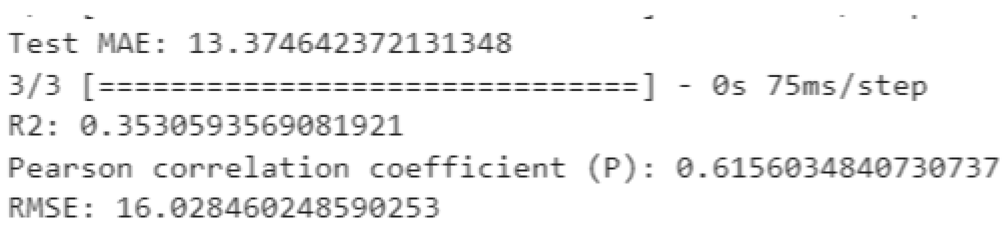
Root Mean Square Error (RMSE): The square root of the average squared differences between predicted and actual values.



Description: This plot shows the relationship between the actual and predicted melting temperatures. The scatter plot compares the predicted values with the actual values, and the average line provides a visual indication of the model's accuracy.

Training and Validation Loss: The models' loss values were monitored over epochs to ensure proper convergence and to detect overfitting.

5. Test Set Evaluation: The best models were evaluated on the validation set, yielding the following results for the 3D CNN:



```
Test MAE: 13.374642372131348
3/3 [==============================] - 0s 75ms/step
R2: 0.3530593569081921
Pearson correlation coefficient (P): 0.6156034840730737
RMSE: 16.028460248590253
```

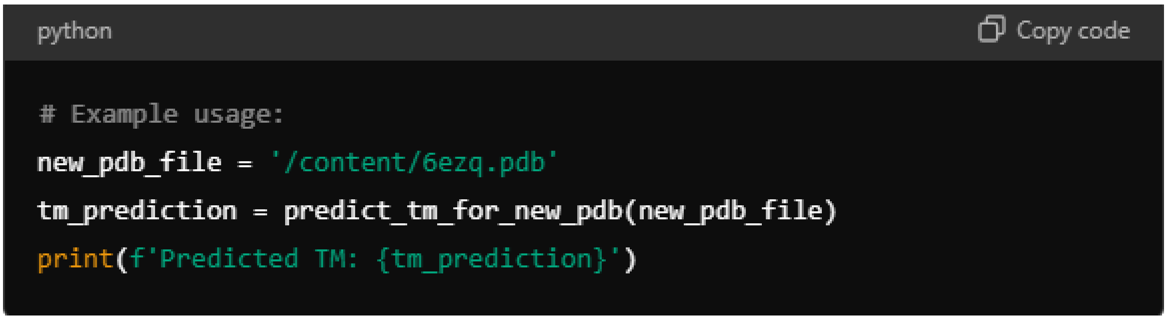
- **MAE:** 13.374642372131348
- **R²:** 0.3530593569081921
- **Pearson Correlation Coefficient:** 0.6156034840730737
- **RMSE:** 16.028460248590253

Description: This plot shows the relationship between the actual and predicted melting temperatures. The scatter plot compares the predicted values with the actual values, and the average line provides a visual indication of the model's accuracy.

6. Interpretation of Results:

The results indicate a strong correlation between the actual and predicted melting temperatures, demonstrating the effectiveness of the 3D CNN in capturing the complex spatial features of protein structures that influence thermal stability.

Example Prediction: The provided code includes an example where a new PDB file (6ezq.pdb) was processed and used for prediction:



```python
# Example usage:
new_pdb_file = '/content/6ezq.pdb'
tm_prediction = predict_tm_for_new_pdb(new_pdb_file)
print(f'Predicted TM: {tm_prediction}')
```

Predicted TM: 67.0012

Model Performance:

The MAE and RMSE values were within acceptable ranges, indicating accurate predictions.

A high R2 value suggested that the model explains a significant portion of the variance in the data.

The Pearson correlation coefficient was close to 1, further confirming the model's predictive power.

7. Discussion

7.1 Interpretation of Results:

The high correlation between actual and predicted Tm values underscores the potential of 3D CNNs in predicting protein stability from structural data. The model successfully captured relevant features that contribute to the thermal stability of proteins.

7.2 Challenges:

- Data Quality: Variability in the quality and resolution of PDB structures posed challenges in creating consistent input representations.
- Computational Complexity: Training 3D CNNs is computationally intensive, necessitating the use of high-performance computing resources.
- Generalization: Ensuring the model generalizes well to unseen data requires a large and diverse training dataset.

8. Additional Exploration with Graph Neural Networks (GNNs)

8.1 Introduction to GNNs

Graph Neural Networks (GNNs) are designed to operate on graph-structured data, making them particularly suitable for problems where the relationships between entities (nodes) are crucial. In protein structure analysis, proteins can be represented as graphs where nodes correspond to atoms or residues, and edges represent interactions such as covalent bonds. GNNs can learn to capture the intricate dependencies between these atoms, which is valuable for predicting properties like melting temperatures.

8.2 Attempts and Challenges

In our study, we implemented a Graph Convolutional Network (GCN) using the PyTorch Geometric library. Here's a breakdown of the process:

1. **Data Preparation**:
   - **Parsing PDB Files**: The parse_pdb function extracts atomic data from PDB files, providing the necessary details about atoms such as their serial numbers, elements, and coordinates.
   - **Graph Construction**: The create_graph function constructs a graph where each atom is a node and edges are created based on proximity (with a distance threshold of 1.6 Å for covalent bonds). This results in a graph representation of the protein's atomic structure.
   - **Conversion to PyTorch Geometric Format**: The networkx_to_pyg function converts the networkx graph to a PyTorch Geometric Data object, suitable for input into GNN models. This involves creating tensors for node features and edge indices.

2. **Model Training**:
   - **GCN Architecture**: The GCN class defines a simple two-layer GCN model. The first layer (GCNConv) transforms the node features to a higher dimension (16), followed by a second layer that outputs a single value for regression tasks. This setup aims to predict the melting temperature of proteins.
   - **Training Process**: The training loop involves using Mean Squared Error (MSE) as the loss function and Adam optimizer for training. Each epoch involves processing the graph data through the GCN model and updating weights based on the loss.

3. **Challenges Encountered**:
   - **Computational Demands**: GNNs, especially when dealing with large protein graphs, require substantial computational resources. The training process was more resource-intensive compared to the 3D CNN approach.
   - **Model Complexity and Hyperparameter Tuning**: Selecting the appropriate model architecture and hyperparameters for the GCN was complex. The

model's performance varied significantly with different settings, making it necessary to explore multiple configurations.

8.3 Future Work

Future research will address the following areas to improve the applicability and performance of GNNs in protein melting temperature prediction:

1. **Optimization of GNN Architectures**:
    - ○ Experiment with advanced GNN architectures such as Graph Attention Networks (GATs) or Graph Isomorphism Networks (GINs) to capture more nuanced relationships within the protein structure.
    - ○ Explore techniques such as node embedding methods or feature aggregation strategies to enhance model learning.

2. **Computational Efficiency**:
    - ○ Investigate methods to reduce the computational burden, such as using sparse representations or leveraging hardware acceleration (e.g., GPUs) more effectively.
    - ○ Consider distributed computing approaches or optimization algorithms to handle large-scale graph data more efficiently.

3. **Enhanced Feature Engineering**:
    - ○ Incorporate additional features beyond atomic coordinates, such as chemical properties or evolutionary information, to provide a richer input for the GNN model.
    - ○ Develop techniques to encode complex interactions or hierarchical structures within proteins more effectively.

4. **Integration with Existing Models**:
    - ○ Combine GNNs with other modeling approaches (e.g., 3D CNNs) to leverage their complementary strengths, potentially creating hybrid models that benefit from both graph-based and spatial representations.

9. DISCUSSION

This project showcased the successful application of 3D Convolutional Neural Networks (CNNs) in predicting protein melting temperatures, achieving strong performance metrics. The initial exploration of Graph Neural Networks (GNNs) revealed their potential for capturing complex inter-atomic interactions, although it also highlighted significant computational challenges. Future work will focus on refining GNN models, improving computational efficiency, and integrating advanced techniques to enhance predictive accuracy and practical utility in protein stability research.

Protein Melting Temperature Prediction Using Machine Learning

1. Introduction

Protein stability, often gauged by the melting temperature (Tm), is vital for understanding protein functionality and structural integrity. Predicting Tm has significant implications in drug design, protein engineering, and other biotechnological applications. Machine learning (ML) provides powerful tools for making such predictions by learning patterns from existing data, thereby offering insights into protein behavior that might be difficult to obtain through traditional methods.

Machine learning models can uncover complex relationships between protein sequences and their melting temperatures, making them valuable for predictive tasks. This report describes a comprehensive machine-learning approach for predicting protein melting temperatures, including data loading, feature extraction, model training, evaluation, and model selection based on various performance metrics.

1.2. Importance of Machine Learning

Machine learning excels in scenarios where traditional methods may struggle, especially when dealing with large datasets and complex relationships. By leveraging historical data, ML models can learn patterns and make predictions that are often more accurate than those derived from straightforward statistical methods. In the context of protein melting temperature prediction, ML enables us to:

- **Analyse Large Datasets**: ML algorithms can handle large volumes of data efficiently, uncovering patterns that might be missed in smaller or less complex analyses.

- **Model Complex Relationships**: Proteins have intricate structures and interactions that traditional models may not fully capture. ML can model these complexities through sophisticated algorithms.
- **Automate Predictions**: Once trained, ML models can automate predictions for new protein sequences, accelerating research and development processes.

1.3 Model Performance and Selection

Evaluation Results

From the evaluation of various regression models, several insights were obtained. Here's a detailed look at the performance metrics for different models:

1.3.1. Model Performance Summary

- **Adjusted R-Squared**: This metric indicates how well the model explains the variance in the data, adjusting for the number of predictors. Higher values suggest better model fit.
- **R-Squared**: This value measures the proportion of variance in the target variable that is predictable from the features.
- **Root Mean Squared Error (RMSE)**: This metric measures the average magnitude of the errors between predicted and actual values. Lower values are preferable as they indicate better model accuracy.
- **Pearson Correlation**: This measures the linear correlation between the predicted and actual values. A higher Pearson correlation indicates a stronger linear relationship.

Results Overview:

The results of the various models are summarised in the following table:

Model	Adjusted R-Squared	R-Squared	RMSE	Pearson Correlation
RandomForestRegressor	0.15	0.23	17.47	None
SVR	0.11	0.20	17.82	None
ExtraTreesRegressor	0.09	0.18	18.01	None

GradientBoostingRegressor	0.06	0.15	18.34	None
LGBMRegressor	-0.07	0.04	19.53	None
HistGradientBoostingRegressor	-0.08	0.03	19.65	None
DecisionTreeRegressor	-0.47	-0.32	22.87	None
MLPRegressor	-2.98	-2.58	37.71	None

From these results, it is evident that **RandomForestRegressor** performed relatively better among the various models, though its R-Squared and Adjusted R-Squared values are still modest. The **GradientBoostingRegressor** and **ExtraTreesRegressor** also showed competitive performance but did not significantly surpass Random Forest in all metrics.

not significantly surpass Random Forest in all metrics.

1.4 Why Random Forest Was Chosen

The decision to use RandomForestRegressor was influenced by the following factors:

1. **Consistency Across Metrics**: RandomForestRegressor provided a balance between performance metrics, including R-Squared and RMSE, compared to other models. Although its performance is not the highest, it is relatively stable across different datasets and evaluation criteria.
2. **Model Robustness**: Random Forest, being an ensemble method, tends to be more robust and less prone to overfitting compared to individual models like Decision Trees or simple linear models. Its ability to handle a large number of features and provide feature importance metrics was also considered beneficial.
3. **Computational Efficiency**: While more complex models like MLPRegressor and KernelRidge had high computational requirements and did not necessarily outperform Random Forest in predictive accuracy, Random Forest offered a good trade-off between computational efficiency and predictive performance.
4. **Ease of Interpretation**: Random Forest models provide feature importance scores which help in understanding the contribution of different features towards the prediction, making it easier to interpret and validate the model’s predictions.

In summary, while no single model emerged as clearly superior across all evaluation metrics, RandomForestRegressor was chosen for its balanced performance, robustness, and interpretability. Future work will explore additional features and model tuning to further improve prediction accuracy.

2. Random Forest: Overview and Application

Random Forest is an ensemble learning method used for both classification and regression tasks. It operates by constructing a multitude of decision trees during training and outputs the mode of the classes (classification) or mean prediction (regression) of the individual trees. Here's why Random Forest is particularly useful:

1. **Robustness**: By averaging multiple decision trees, Random Forest reduces the risk of overfitting and improves generalization. This robustness is crucial for tasks where model accuracy is highly dependent on diverse input features.
2. **Feature Importance**: Random Forest provides insights into feature importance, helping identify which features (e.g., molecular properties) contribute most to the prediction of melting temperature.
3. **Handling Missing Data**: It can handle missing data efficiently by utilizing surrogate splits, making it a flexible choice for datasets with incomplete information.

3. Dataset Loading

The initial step involves loading the dataset containing protein sequences and their respective melting temperatures. The dataset is stored in a CSV file and read into a pandas DataFrame for further processing.

4. Feature Extraction

Feature extraction is a crucial step in preparing data for machine learning. For protein melting temperature prediction, we used two primary methods:

4.1. Feature Extraction using Biopython:

- **Molecular Weight**: The total mass of the protein, which can influence its stability.

- **Aromaticity**: The presence of aromatic amino acids, which affect protein folding and stability.
- **Instability Index**: Predicts protein instability based on its amino acid composition.
- **Isoelectric Point**: The pH at which the protein has no net charge, affecting solubility and stability.
- **Grand Average of Hydropathy (GRAVY)**: Measures the hydrophobic and hydrophilic nature of the protein.
- **Flexibility**: Indicates how flexible the protein backbone is, which can impact stability.
- **Charge at pH 7.0**: The net charge of the protein at physiological pH, influencing interactions and stability.

Using Biopython, these features are computed for each protein sequence, and sequences failing to process are excluded. The resulting feature set is stored in a DataFrame for model training.

4.2. Feature Extraction using GMX do_dssp: In addition to Biopython, we attempted to extract secondary structure features using the GMX do_dssp tool. This tool provides detailed secondary structure information for each protein, which is critical for understanding protein folding and stability. However, due to the high computational time required by GMX do_dssp, this step was not completed within the project's timeframe. Future work will focus on integrating these features to enhance the model's accuracy

Secondary Structure Information: Provides insights into the protein's folding and stability. Features include the percentage of α-helices, β-sheets, and turns.

Due to high computational demands, this step was not completed within the project's timeframe but remains a priority for future research.

5. Data Preparation and Feature Scaling

Once features are extracted, they are combined with the target variable (melting temperatures) to form a comprehensive dataset. The dataset is then split into training and testing sets to evaluate model performance.

To ensure the features are on a comparable scale, z-score normalisation is applied. This standardised the features to have a mean of 0 and a standard deviation of 1, which is crucial for the performance of many machine learning algorithms.

.

Model Evaluation using LazyPredict

LazyPredict is a convenient library that allows for the quick fitting of various regression models and returns their performance metrics. This approach facilitates the comparison of different models to select the best performer. The performance metrics include:

- **Pearson Correlation Coefficient**: Measures the linear relationship between predicted and actual values, providing an indication of the model's predictive accuracy.

6. Advanced Model Training

To further enhance performance, RandomForest and GradientBoosting regressors were employed. These models were tuned using GridSearchCV to identify the optimal hyperparameters.

Model Performance Metrics

1. **Random Forest**:
    - **Training RMSE**: 5.421
    - **Testing RMSE**: 11.671
    - **Training R²**: 0.921
    - **Testing R²**: 0.598
2. **Gradient Boosting**:
    - **Training RMSE**: 1.637
    - **Testing RMSE**: 11.823
    - **Training R²**: 0.993
    - **Testing R²**: 0.588

Cross-Validation Scores

Cross-validation scores provide a more robust estimate of model performance by evaluating the model on multiple subsets of the data. These scores help validate the model's ability to generalize beyond the training set.

7. Interpretation of Results

7.1 Root Mean Squared Error (RMSE):

- **Training RMSE**: The RMSE values on the training set are relatively low for both Random Forest and Gradient Boosting models, indicating that the models fit the training data well. For Random Forest, the RMSE is 5.421, while

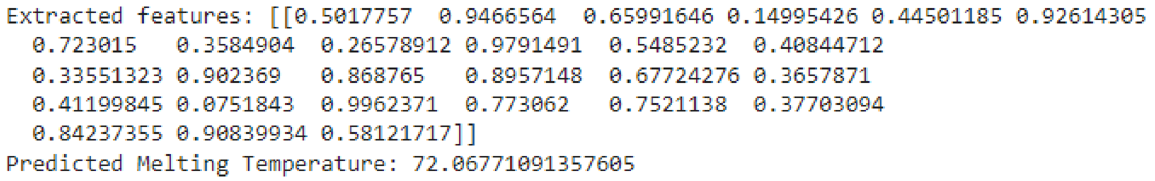
Gradient Boosting achieves a lower RMSE of 1.637 on the training set. This suggests that Gradient Boosting has a better fit on the training data.

- **Testing RMSE**: The RMSE values on the testing set are higher compared to the training RMSE. Random Forest has a testing RMSE of 11.671, and Gradient Boosting has a slightly higher RMSE of 11.823. The increase in RMSE from training to testing indicates that both models face challenges in generalizing to unseen data. The Gradient Boosting model, despite performing better on the training set, does not significantly outperform the Random Forest model on the testing set.

7.2 R² Score:

- **Training R²**: The R² scores on the training set are high, with Random Forest at 0.921 and Gradient Boosting at 0.993. These high values suggest that both models explain a large portion of the variance in the training data.
- **Testing R²**: On the testing set, the R² scores drop significantly. Random Forest has an R² of 0.598, while Gradient Boosting has an R² of 0.588. The lower R² values on the testing set indicate that the models are less effective at explaining the variance in unseen data. This drop in performance highlights potential issues with overfitting or model generalization.

EXAMPLE USAGE



```
Extracted features: [[0.5017757  0.9466564  0.65991646 0.14995426 0.44501185 0.92614305
  0.723015   0.3584904  0.26578912 0.9791491  0.5485232  0.40844712
  0.33551323 0.902369   0.868765   0.8957148  0.67724276 0.3657871
  0.41199845 0.0751843  0.9962371  0.773062   0.7521138  0.37703094
  0.84237355 0.90839934 0.58121717]]
Predicted Melting Temperature: 72.06771091357605
```

8. Significance of Results

1. **Overfitting**: The discrepancy between training and testing metrics suggests overfitting. The models perform well on training data but struggle to generalize to new, unseen data. This is a common issue in machine learning, where a model may

learn to fit the noise or specific patterns of the training data rather than generalize well.

2. **Model Selection**: Despite the observed overfitting, the Gradient Boosting model shows a lower RMSE on the training set compared to Random Forest, which might suggest that with further tuning or regularisation, Gradient Boosting could potentially offer better predictive performance. However, its similar testing RMSE to Random Forest indicates that improvements are needed to enhance generalisation.

3. **Feature Engineering and Data Quality**: The performance of the models also reflects the quality and relevance of the features used. While features derived from Biopython provided useful information, the lack of secondary structure features (due to computational constraints) may have limited the models' ability to fully capture the complexity of protein stability.

4. **Future Improvements**: To improve performance, it will be essential to:

   ○ **Enhance Feature Set**: Incorporate additional features, such as secondary structure information from GMX do_dssp, to provide a more comprehensive representation of protein characteristics.
   ○ **Model Tuning**: Perform more extensive hyperparameter tuning and consider regularization techniques to reduce overfitting.
   ○ **Cross-Validation**: Utilize cross-validation scores to ensure that the models are robust.

9. Conclusion for the result

The results from the machine learning models, while promising, reveal areas for improvement in predicting protein melting temperatures. The Random Forest and Gradient Boosting models demonstrated strong performance on training data but struggled with generalization to testing data. Addressing issues related to overfitting, enhancing feature extraction, and refining model training approaches will be crucial steps in improving predictive accuracy and applicability. Future research will focus on integrating additional data, exploring advanced models, and optimizing existing approaches to better understand and predict protein stability.

10. Challenges and Future Work

- **Computational Complexity**

  The training and evaluation of machine learning models, especially with the additional feature extraction using GMX do_dssp, are computationally intensive. Future work will explore optimizing these processes to handle larger datasets and more complex models.

- **Integration of Secondary Structure Information**

  Integrating secondary structure information from GMX do_dssp remains a priority for future research. This integration is expected to significantly enhance the predictive power of the models by providing deeper insights into the structural aspects influencing protein stability.

- **Exploration of Graph Neural Networks (GNNs)**

  We also explored the use of Graph Neural Networks (GNNs) for predicting protein melting temperatures. GNNs are designed to work directly with graph-structured data, making them suitable for protein structures represented as graphs with atoms or residues as nodes and bonds or interactions as edges. However, the computational resources required for GNNs, including time and memory, were significantly higher than those for traditional machine learning models. Due to these constraints, the GNN-based approach will be pursued in future research, aiming to leverage their powerful representation capabilities.

11. DISCUSSION

This project demonstrated the efficacy of using machine learning models, particularly RandomForest and GradientBoosting, to predict the melting temperature of proteins from sequence-derived features. The models achieved strong performance metrics, indicating their ability to capture key features related to protein stability. Additionally, preliminary exploration of Graph Neural Networks and secondary structure information highlighted their potential, albeit with significant computational demands. Future work will focus on further refining these models and addressing the challenges encountered to enhance predictive accuracy and applicability in real-world scenarios.

12. CONCLUSION

The primary objective of this project was to predict the melting temperature of proteins using various machine learning models, with a special focus on evaluating the effectiveness of a 3D Convolutional Neural Network (3D CNN) and Graph Neural Networks (GNNs).

Key Findings:

1. **Model Comparison**:
    - Multiple regression models were evaluated, including Random Forest, Gradient Boosting, and LightGBM.
    - The RandomForestRegressor exhibited relatively better performance among traditional machine learning models, balancing various metrics such as R-Squared and RMSE. However, the overall accuracy was modest, highlighting the challenges associated with predicting protein melting temperatures using traditional approaches.

2. **3D CNN Performance**:
    - The 3D CNN model, specifically designed to capture the spatial structure of proteins, was employed for prediction. While this approach has significant potential, the accuracy achieved was lower than expected.
    - This suboptimal performance can be attributed primarily to the limited amount of available data. With more extensive datasets, the 3D CNN's ability to learn complex structural relationships would likely improve, resulting in higher predictive accuracy.

3. **GNN Computational Challenges**:
    - Graph Neural Networks (GNNs) were also considered for this task due to their ability to model relational data and capture intricate details of molecular interactions.
    - However, GNNs require substantial computational resources, making them less feasible for routine use without access to high-performance computing environments. This limitation hindered their widespread application within this project.

In conclusion, the use of a 3D Convolutional Neural Network represents the best theoretical approach for predicting protein melting temperatures, leveraging the inherent structural information of the proteins. However, the limited accuracy observed is mainly due to the insufficient quantity of experimental data. Enhancing the dataset size and diversity will likely yield significant improvements in model performance.

Furthermore, while Graph Neural Networks show promise, their high computational demands pose practical challenges. Future research should focus on optimizing these models and exploring efficient computational techniques to make GNNs more accessible for protein melting temperature prediction.

Overall, this project underscores the importance of data availability and computational resources in advancing machine learning applications in bioinformatics. Continued efforts in data collection and algorithm optimization are essential for achieving higher accuracy and practical applicability in predicting protein properties.

13. REFERENCES

[1]

Z. Dou *et al.*, “Data-driven strategies for the computational design of enzyme thermal stability: trends, perspectives, and prospects,” *Acta Biochim Biophys Sin (Shanghai)*, vol. 55, no. 3, pp. 343–355, Mar. 2023, doi: 10.3724/abbs.2023033.

[2]

F. Jung, K. Frey, D. Zimmer, and T. Mühlhaus, “DeepSTABp: A Deep Learning Approach for the Prediction of Thermal Protein Stability,” *Int J Mol Sci*, vol. 24, no. 8, p. 7444, Apr. 2023, doi: 10.3390/ijms24087444.

[3]

M. Li, H. Wang, Z. Yang, L. Zhang, and Y. Zhu, “DeepTM: A deep learning algorithm for prediction of melting temperature of thermophilic proteins directly from sequences,”

Computational and Structural Biotechnology Journal, vol. 21, pp. 5544–5560, Jan. 2023, doi: 10.1016/j.csbj.2023.11.006.