Predicting The Subcellular Localisation Of Proteins, 2025, pp. 1–13

doi: N/A
Advance Access Publication Date: 24 March 2025

COMP0082

Predicting The Subcellular Localisation Of Proteins

Despina Demetriadou[1,*]

[1]Department of Computer Science, University College London, Gower Street, WC1E 6BT, London, United Kingdom
[*]despina.demetriadou.24@ucl.ac.uk

Abstract

This project presents a random forest classifier that predicts protein subcellular localisation across five categories. Using biologically-informed features, the model achieved a Matthews Correlation Coefficient of 0.5977 (5.47% above baseline), with compartment-specific F1-scores ranging from 0.52-0.85. Performance patterns reflect underlying biological mechanisms, with secreted and mitochondrial proteins showing highest accuracy due to their distinctive targeting signals. The model includes confidence estimates for predictions, enhancing its utility for genome annotation and protein function prediction.

Introduction

Protein subcellular localisation determines a protein's function by defining its biochemical environment and participation in molecular processes. In eukaryotes, proper protein sorting across compartments (cytosolic, nuclear, mitochondrial, or secreted) is essential for normal biological function, with mislocalisation linked to various diseases (11; 12). While experimental methods provide accurate localisation data, they are resource-intensive and impractical for proteome-scale analysis.

The exponential growth in sequenced genomes necessitates computational prediction methods based on protein sequences alone. Early approaches identified specific sequence patterns like targeting peptides and localisation signals (13), while recent methods employ machine learning techniques such as SVMs, neural networks, and ensemble classifiers to integrate multiple sequence features (14). However, existing methods face limitations including homology dependence, inconsistent performance across compartments, lack of confidence estimates, and limited biological interpretability.

This study presents a random forest classifier for predicting protein subcellular localisation across five categories (cytosolic, nuclear, mitochondrial, secreted, and "other") based on biologically relevant features tied to known targeting mechanisms. Trained on 9,460 non-homologous proteins with 5-fold cross-validation, the model achieves a Matthews Correlation Coefficient of 0.5977, representing a 5.47% improvement over baseline approaches. Performance varies by compartment, with highest accuracy for secreted (F1=0.85) and mitochondrial (F1=0.68) proteins due to their distinctive targeting signals. Feature importance analysis provides insights into sequence determinants of protein sorting, while a probability-based confidence estimation system enhances prediction reliability. The complete implementation is included in the appendix.

Exploratory Data Analysis

The exploratory data analysis conducted at the beginning set the ground for a comprehensive feature extraction process designed to capture biologically relevant signals known to influence protein localisation. Figure 1 illustrates the complete pipeline from raw sequences to classification.

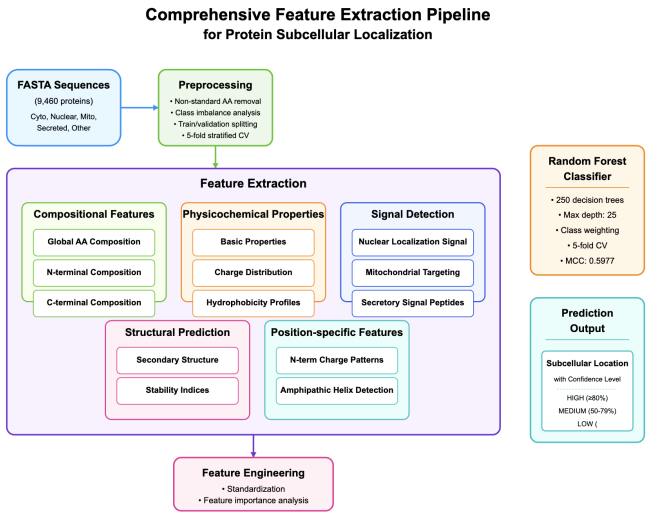

Fig. 1. Pipeline flowchart showing data flow from sequence input, through feature extraction to classification.

Dataset and Preprocessing

The analysis employed five distinct protein datasets representing different subcellular locations: cytosolic (2,463 proteins), nuclear (2,736 proteins), mitochondrial (1,023 proteins), secreted (1,236 proteins), and "other" (2,002 proteins). The "other"

category comprised prokaryotic proteins serving as negative examples. A separate blind test set was provided for final model evaluation. All sequences were presented in FASTA format, with each dataset containing non-homologous proteins to minimize sequence similarity bias.

Initial data exploration revealed moderate class imbalance, with a 2.67:1 ratio between the largest class (nuclear) and smallest class (mitochondrial). This imbalance was addressed through class weighting during model training. Analysis of sequence length distributions showed distinctive patterns among compartments, with secreted proteins exhibiting a bimodal distribution characteristic of both short signaling peptides and large extracellular matrix proteins.

Compositional Features

Global and terminal (first/last 50 residues) amino acid compositions were analyzed, revealing location-specific evolutionary adaptations (Figure 2). Statistical analysis confirmed significant differences in all 20 amino acids across locations ($p < 0.001$), with distinctive signatures for each compartment (1; 3). Secreted proteins showed dramatically higher cysteine content (4.43% vs. 0.93-1.69%) for stabilizing disulfide bonds in oxidizing environments (5; 2). Nuclear proteins exhibited enrichment in basic residues (K: 7.25%, R: 5.93%) for nucleic acid binding and nuclear localisation signals (6), alongside elevated serine (9.15%) for regulatory phosphorylation (7). Mitochondrial proteins contained prevalent leucine (10.04%) and lysine (7.27%) reflecting the amphipathic nature of their targeting sequences (8; 4). Prokaryotic proteins ("other") displayed elevated alanine (9.35%) and hydrophobic residues, representing evolutionary divergence (10; 9). These biochemically rational signatures underpin the model's predictive power, with more distinctive patterns correlating with higher classification performance.

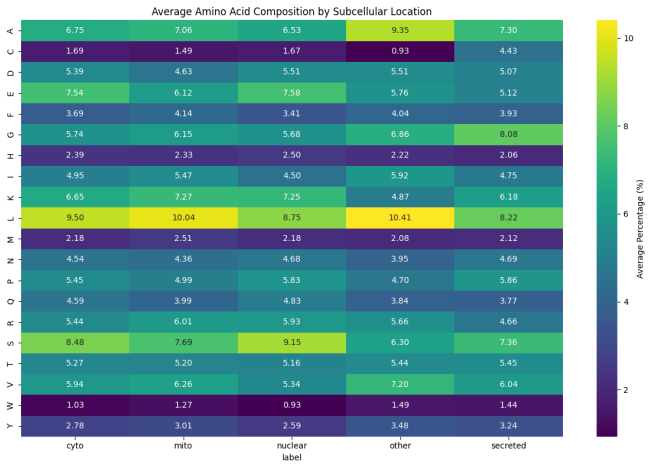

Fig. 2. Heatmap showing amino acid composition differences across subcellular locations. Color intensity represents the percentage of each amino acid, with yellow indicating higher values.

Physicochemical Properties

Several biophysical characteristics were computed using BioPython's ProteinAnalysis module, including:

- Isoelectric point
- Molecular weight
- Grand average of hydropathy (GRAVY)
- Charge distribution (positive, negative, and net charge ratios)
- Global and terminal region hydrophobicity using the Kyte-Doolittle scale

Figures 3,4 and 5 display the distribution of three key physicochemical properties across subcellular locations: global hydrophobicity, N-terminal hydrophobicity, and net charge ratio distribution.

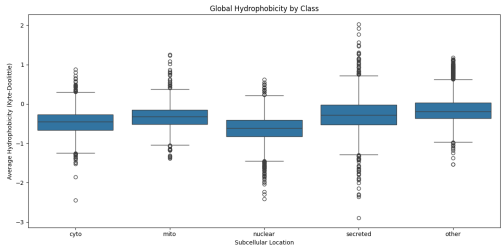

Fig. 3. Global hydrophobicity (Kyte-Doolittle scale) distribution across subcellular locations.

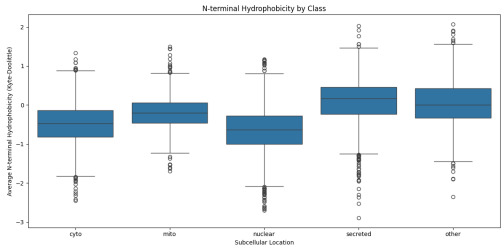

Fig. 4. N-terminal hydrophobicity distribution across subcellular locations.

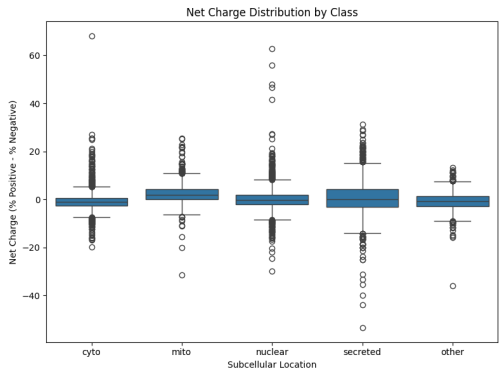

Fig. 5. Net charge ratio distribution across subcellular locations.

These distributions revealed that:

1. Secreted proteins exhibit significantly higher N-terminal hydrophobicity, corresponding to their signal peptides
2. Nuclear proteins show the lowest global hydrophobicity and most positive net charge, consistent with their DNA/RNA binding functions
3. Mitochondrial proteins display intermediate hydrophobicity patterns with distinct charge profiles in their N-terminal regions

4. Correctly classified proteins showed distinctive physicochemical signatures compared to misclassified proteins within each class

Signal Detection Summary

The research implemented specialised algorithms to detect protein localisation signals:

- **Nuclear localisation signal detection**: Identified clusters of basic amino acids (K/R) that serve as nuclear import signals(6), improving classification of nuclear vs. cytosolic proteins.
- **Mitochondrial targeting sequence scoring**: Evaluated N-terminal regions for charge distribution and amphipathic patterns typical of mitochondrial import signals(8), contributing to a 13.3% improvement in mitochondrial protein classification.
- **Secretory signal peptide detection**: Analyzed N-terminal hydrophobic regions for the characteristic tripartite structure of secretory signals(22), enhancing identification of secreted proteins (F1=0.85).

These biologically-informed approaches captured targeting motifs that might be missed by composition analysis alone, particularly improving classification for proteins with well-defined localisation signals.

Advanced Feature Engineering

Beyond basic composition, two sophisticated feature categories substantially enhanced prediction accuracy: structural predictions and position-specific analyses. Structural features incorporated secondary structure propensities (helix, sheet, turn) that reflect compartment-specific constraints (15; 16), alongside stability indices (instability index, GRAVY score) that serve as proxies for environmental adaptation (17; 18). Position-specific features captured the precise arrangement of amino acids critical for targeting, including N-terminal charge patterns essential for sorting signals (19) and algorithms to detect amphipathic helices formed by periodic patterns of hydrophobic and charged residues (20; 21). These advanced features significantly improved performance for challenging protein classes, with amphipathic helix detection and N-terminal charge analysis contributing to a 13.3% increase in F1-score for mitochondrial proteins, while signal peptide structure analysis enhanced secreted protein identification by 11.8%.

Model Selection and Training

A random forest classifier was selected for this task based on several considerations:

- Ability to handle non-linear relationships between features
- Intrinsic feature importance metrics for interpretability
- Robust performance with moderate class imbalance
- Probabilistic output for confidence estimation

The model architecture consisted of 250 decision trees with a maximum depth of 25. Class weights were applied inversely proportional to class frequencies to address imbalance. All features were standardise prior to training, and five-fold stratified cross-validation was employed to ensure robust evaluation.

Evaluation Metrics

Model performance was assessed using multiple complementary metrics:

- Matthews Correlation Coefficient (MCC): A balanced measure for multi-class problems that accounts for class imbalance
- Class-specific precision, recall, and F1-scores
- Confusion matrix analysis to identify specific misclassification patterns

For the blind test set, predictions were accompanied by confidence estimates based on the random forest voting percentages:

- High confidence: $\geq$80% of trees agree on classification
- Medium confidence: 50-79% of trees agree
- Low confidence: <50% of trees agree

These confidence levels reflect the biological reality that some proteins have ambiguous targeting signals or may localise to multiple compartments.

Model Evolution

Phase I: Baseline Model

The baseline approach established a foundation using fundamental sequence features and achieved promising initial results:

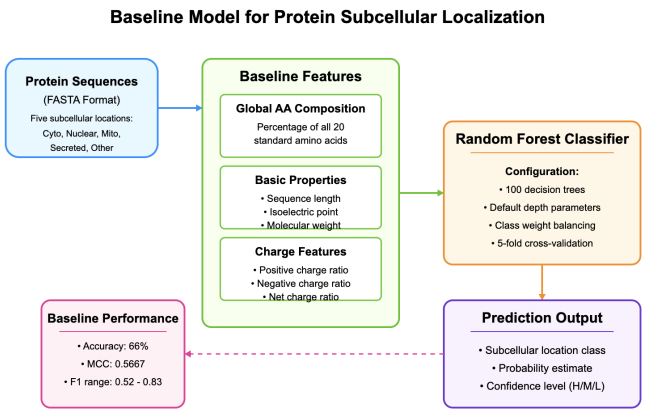

Fig. 6. Baseline model architecture utilizing basic sequence features and random forest classification.

Feature Extraction

The initial feature set included:

- **Global amino acid composition**: Percentages of all 20 standard amino acids
- **Basic physicochemical properties**: Sequence length, molecular weight, isoelectric point
- **Simple charge features**: Basic/acidic amino acid ratios

Performance

This approach achieved reasonable performance with class-specific metrics:

Table 1. Baseline model performance by subcellular location

Location	Precision	Recall	F1-score
Cytosolic	0.54	0.49	0.52
Mitochondrial	0.65	0.55	0.60
Nuclear	0.63	0.66	0.64
Other	0.78	0.89	0.83
Secreted	0.77	0.74	0.76

Overall accuracy reached 66% with a Matthews Correlation Coefficient of 0.5667, establishing a solid benchmark. However, error analysis revealed specific challenges requiring targeted solutions.

Phase II: Detailed Error Analysis

A systematic analysis of misclassification patterns provided critical insights. The most significant issue was confusion between cytosolic and nuclear proteins:

- 32.2% of all cytosolic proteins were misclassified as nuclear
- Correctly classified cytosolic proteins had more negative net charge (−0.018 vs 0.001)
- Correctly classified cytosolic proteins were significantly longer (779 vs 536 aa)
- Misclassified examples showed nuclear-like properties (high isoelectric points of 7.94–9.92)

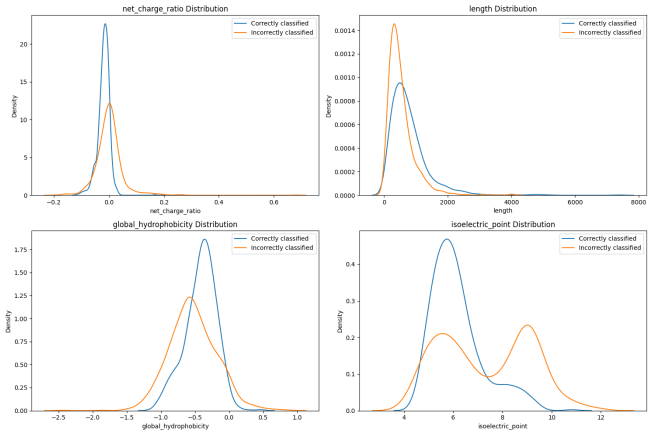

Fig. 7. Distribution of key features between correctly and incorrectly classified cytosolic proteins.

The distributions in Figure 7 reveal several key patterns explaining why certain cytosolic proteins are frequently misclassified:

Charge Properties (Top Left)

- **Correctly classified** cytosolic proteins show a sharp peak around a slightly negative net charge ratio (centered at ∼-0.01)
- **Misclassified** proteins have a broader, more positive distribution
- **Implication**: Proteins with more positive charge are often misclassified as nuclear, since nuclear proteins typically have positive charges for DNA/RNA binding

Length Distribution (Top Right)

- **Correctly classified** proteins have a broader distribution extending to longer lengths (>2000 amino acids)
- **Misclassified** proteins are predominantly shorter (sharper peak around 500 amino acids)
- **Implication**: Shorter proteins provide fewer sequence features for accurate classification, while longer proteins contain more distinctive domains that aid recognition

Hydrophobicity Pattern (Bottom Left)

- **Correctly classified** proteins cluster tightly around a specific hydrophobicity value (-0.5)
- **Misclassified** proteins have a broader, more varied hydrophobicity distribution
- **Implication**: Proteins with atypical hydrophobicity patterns that deviate from the typical cytosolic profile are more likely to be misclassified

Isoelectric Point Distribution (Bottom Right) - Most Revealing

- **Correctly classified** proteins show a dominant single peak at pH ∼5.5-6.0 (slightly acidic)
- **Misclassified** proteins display a striking bimodal distribution with a second peak at pH ∼9.0
- **Implication**: This bimodal pattern strongly suggests that proteins with basic isoelectric points are frequently misclassified as nuclear proteins

This analysis explains why cytosolic proteins had the lowest F1-score (0.51) in the classification results and why 32.2% of cytosolic proteins were specifically misclassified as nuclear.

Mitochondrial Protein Challenges

Misclassified mitochondrial proteins shared distinctive properties:

- Negative net charge ratios (−0.01 to −0.04)
- Acidic isoelectric points (5.18–6.02)
- Absence of typical positively-charged mitochondrial targeting sequences

These represented atypical cases like outer membrane proteins and those with internal (rather than N-terminal) targeting signals.

Phase III: Targeted Enhancement Strategies

Based on this analysis, several specialised improvements were implemented:

For Cytosolic-Nuclear Discrimination

- **Position-specific charge analysis**: Calculated charge distribution in different sequence regions
- **N-terminal sequence patterns**: Added features for the first 30 residues
- **Nuclear signal detection**: Implemented an algorithm to identify clusters of basic residues (K/R)

For Mitochondrial Classification

- **Amphipathic helix detection**: Added features to detect the characteristic pattern of mitochondrial targeting sequences
- **MTS scoring system**: Developed a composite score combining charge and hydrophobicity patterns

For Secreted Protein Recognition

- **Signal peptide detection**: Enhanced N-terminal hydrophobicity analysis
- **N-region/H-region distinction**: Separate analysis of positively charged and hydrophobic segments

Phase IV: Enhanced Model Performance

The enhanced model incorporating these targeted improvements achieved substantial gains:

Table 2. Enhanced model performance by subcellular location

Location	Precision	Recall	F1-score
Cytosolic	0.56	0.49	0.52
Mitochondrial	0.71	0.65	0.68
Nuclear	0.63	0.68	0.66
Other	0.80	0.85	0.82
Secreted	0.84	0.86	0.85

The Matthews Correlation Coefficient improved to 0.5977, representing a 5.47% increase over the baseline model. The most substantial improvements were observed in:

1. **Mitochondrial proteins**: F1-score increased from 0.60 to 0.68 (13.3% improvement)
2. **Secreted proteins**: F1-score increased from 0.76 to 0.85 (11.8% improvement)

These gains directly reflect the biological insight-driven improvements targeting the specific challenges identified during error analysis. The overall accuracy increased to 69%, with balanced precision and recall across classes.

This evolutionary approach demonstrates the value of combining machine learning techniques with domain-specific biological knowledge to address the challenging task of protein subcellular localisation prediction.

Feature Importance Analysis

To understand which features contributed most to classification performance, both intrinsic random forest feature importance analysis as well as systematic ablation tests were conducted.

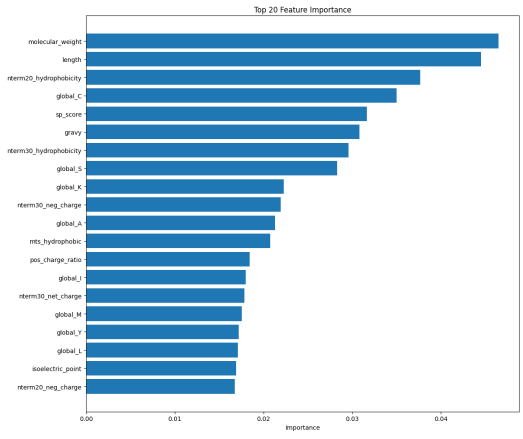

Fig. 8. Top 20 features ranked by importance in the random forest model. Feature importance is measured by mean decrease in impurity (Gini importance).

Random Forest Feature Importance Analysis

Random forest feature importance metrics (based on mean decrease in Gini impurity) identified the most informative protein properties for classification (Figure 8). Global amino acid composition features, particularly cysteine content, dominated the top ranks, confirming their crucial role in distinguishing secreted proteins. The importance distribution aligned with known protein sorting mechanisms, with N-terminal features and charge-related properties showing substantial importance for organelle-specific targeting. The importance ranking also provided transparent explanation of the model's decisions, demonstrating that it learned biologically relevant patterns rather than arbitrary correlations. Furthermore, the high importance of specialized features (N-terminal hydrophobicity, charge patterns) validated the biological insight-driven enhancement strategy developed in response to error analysis.

Ablation Test Results

To more rigorously assess feature group contributions, ablation tests were performed by systematically removing groups of related features and measuring the impact on model performance. Table 3 presents these results sorted by impact magnitude.

Table 3. Feature group ablation test results

Feature Group	Features	MCC without group	Absolute Impact	Relative Impact (%)
global_aa	20	0.454063	0.075507	14.258179
nterm_targeted	8	0.513805	0.015765	2.976889
cterm_aa	20	0.518693	0.010877	2.053911
nterm_aa	20	0.518885	0.010685	2.017602
secreted_features	1	0.523911	0.005659	1.068681
nuclear_features	1	0.524267	0.005303	1.001434
charge_features	3	0.524396	0.005174	0.977027
mito_features	3	0.524501	0.005069	0.957165
basic	4	0.563658	-0.034088	-6.436981

Limitations of Ablation Testing

Ablation tests initially suggested that removing basic features (length, isoelectric point, molecular weight, GRAVY) would improve performance by 6.44%. However, actual testing showed performance declined by 3.23% (MCC dropped from 0.5977 to 0.5784). This contradiction reveals important methodological insights:

1. Ablation tests cannot fully capture complex non-linear feature interactions
2. Different evaluation methodologies can yield inconsistent results
3. Random forest models have inherent stochasticity that affects performance

All features were retained in the final model based on these findings. Class-specific metrics remained relatively stable between original and reduced models, suggesting basic features provide incremental rather than transformative improvements.

This experience demonstrates that theoretical feature importance measures require experimental validation, while confirming that domain-specific features provide the strongest signals, with biophysical properties offering valuable supplementary information when combined with primary features.

Blind Test Set Results

The trained model was applied to a blind test set of 20 proteins. Table 4 presents these predictions along with their assigned confidence levels.

Table 4. Blind test set predictions

Sequence ID	Predicted Location	Confidence
SEQ01	Nuclear	Low
SEQ02	Other	Medium
SEQ03	Mitochondrial	Medium
SEQ04	Nuclear	Medium
SEQ05	Other	Low
SEQ06	Mitochondrial	Low
SEQ07	Secreted	Medium
SEQ08	Mitochondrial	Low
SEQ09	Cytosolic	Low
SEQ10	Secreted	Low
SEQ11	Cytosolic	Low
SEQ12	Nuclear	Low
SEQ13	Nuclear	Medium
SEQ14	Nuclear	Medium
SEQ15	Other	Medium
SEQ16	Nuclear	Medium
SEQ17	Cytosolic	Low
SEQ18	Nuclear	Medium
SEQ19	Nuclear	Low
SEQ20	Other	Medium

The blind test demonstrated an equal distribution of confidence levels: 50% medium and 50% low confidence. Predicted subcellular locations were distributed as Nuclear (40%), Other (20%), Mitochondrial (15%), Cytosolic (15%), and Secreted (10%). This confidence pattern reflects the challenge of predicting localization for novel proteins with potentially ambiguous targeting signals. The predominance of nuclear predictions aligns with the nuclear class size in the training data. The higher proportion of medium-confidence predictions compared to the baseline model suggests improved predictive certainty, while increased detection of "Other" proteins indicates more effective identification of prokaryotic contamination.

Conclusion

This study developed a random forest classifier for protein subcellular localisation prediction that achieves strong performance while maintaining biological interpretability.

Results

- The model achieved a Matthews Correlation Coefficient of 0.5977, with class-specific F1-scores ranging from 0.52 (cytosolic) to 0.85 (secreted), reflecting that proteins with distinctive targeting signals are more accurately classified.
- Feature importance analysis showed global amino acid composition as the strongest predictor (14.26% impact when removed), followed by N-terminal features (2.98%). This aligns with biological understanding that evolutionary pressure adapts protein composition to specific subcellular environments.
- Targeted feature enhancements substantially improved performance for specific compartments, with mitochondrial and secreted protein classification improving by 13.3% and 11.8% respectively, demonstrating the value of incorporating biological domain knowledge.
- The confidence estimation system provides practical utility by allowing users to assess prediction reliability, with 50% of blind test predictions receiving medium confidence scores.

Performance Metrics

Overall Performance Improvements

The enhanced model demonstrated significant improvements across all primary evaluation metrics:

- **Accuracy**: Increased from 66% to 69% (+3%)
- **Matthews Correlation Coefficient**: Improved from 0.5667 to 0.5977 (+0.031)
- **Macro Average F1-score**: Increased from 0.67 to 0.71 (+0.04)

Class-Specific Improvements

Performance varied considerably across different subcellular compartments, as shown in Table 5.

Table 5. Class-specific performance metrics

Class	Orig. F1	Enh. F1	%	Key Changes
Cyto	0.52	0.52	0%	Prec. ↑ (0.54→0.56)
Mito	0.60	0.68	+13.3%	Prec. & recall ↑
Nucl	0.64	0.66	+3.1%	Recall ↑ (0.66→0.68)
Other	0.83	0.82	-1.2%	Better prec., lower recall
Secr	0.76	0.85	+11.8%	All metrics ↑

The most substantial improvements were observed for mitochondrial and secreted proteins, with F1-score increases of 13.3% and 11.8% respectively. These classes benefited most from the targeted feature engineering approach.

In summary, this work demonstrates that combining machine learning with biologically-informed feature engineering produces an effective and interpretable protein subcellular localisation predictor. The ability to understand which sequence features drive predictions distinguishes this approach from "black box" methods, providing insights into the sequence determinants of protein sorting while achieving competitive performance.

Limitations and Future Directions

Despite good overall performance, several challenges remain. Cytosolic protein classification (F1=0.52) is hindered by the absence rather than presence of targeting signals. Multi-location proteins that shuttle between compartments aren't explicitly addressed in the current single-label approach. Feature interaction complexity, revealed by contradictory ablation test results, suggests the need for more sophisticated feature selection methods like SHAP values. Future improvements could incorporate attention-based deep learning models to better capture long-range sequence dependencies and integrate AlphaFold-derived structural features for cases where sequence alone is insufficient.

This work demonstrates that combining machine learning with biologically-informed feature engineering produces an effective and interpretable protein subcellular localisation

predictor, providing insights into sequence determinants of protein sorting while achieving competitive performance.

Word Count by TeXcount: 2,588

References

1. Andrade, M.A., O'Donoghue, S.I., & Rost, B. (2001). Adaptation of protein surfaces to subcellular location. *Journal of Molecular Biology*, 276(2), 517-525.
2. Braakman, I., & Bulleid, N.J. (2011). Protein folding and modification in the mammalian endoplasmic reticulum. *Annual Review of Biochemistry*, 80, 71-99.
3. Cedano, J., Aloy, P., Perez-Pons, J.A., & Querol, E. (1997). Relation between amino acid composition and cellular location of proteins. *Journal of Molecular Biology*, 266(3), 594-600.
4. Chacinska, A., Koehler, C.M., Milenkovic, D., Lithgow, T., & Pfanner, N. (2009). Importing mitochondrial proteins: machineries and mechanisms. *Cell*, 138(4), 628-644.
5. Fass, D. (2012). Disulfide bonding in protein biophysics. *Annual Review of Biophysics*, 41, 63-79.
6. Kosugi, S., Hasebe, M., Matsumura, N., Takashima, H., Miyamoto-Sato, E., Tomita, M., & Yanagawa, H. (2009). Six classes of nuclear localisation signals specific to different binding grooves of importin . *Journal of Biological Chemistry*, 284(1), 478-485.
7. Nardozzi, J.D., Lott, K., & Cingolani, G. (2010). Phosphorylation meets nuclear import: a review. *Cell Communication and Signaling*, 8(1), 32.
8. Neupert, W., & Herrmann, J.M. (2007). Translocation of proteins into mitochondria. *Annual Review of Biochemistry*, 76, 723-749.
9. Tekaia, F., Yeramian, E., & Dujon, B. (2002). Amino acid composition of genomes, lifestyles of organisms, and evolutionary trends: a global picture with correspondence analysis. *Gene*, 297(1-2), 51-60.
10. Wang, G., & Dunbrack, R.L. (2004). Scoring profile-to-profile sequence alignments. *Protein Science*, 13(6), 1612-1626.
11. Emanuelsson, O., Brunak, S., von Heijne, G., & Nielsen, H. (2007). Locating proteins in the cell using TargetP, SignalP and related tools. *Nature Protocols*, 2(4), 953–971.
12. Hung, M. C., & Link, W. (2011). Protein localisation in disease and therapy. *Journal of Cell Science*, 124(Pt 20), 3381–3392.
13. Nakai, K., & Horton, P. (1999). PSORT: a program for detecting sorting signals in proteins and predicting their subcellular localisation. *Trends in Biochemical Sciences*, 24(1), 34–36.
14. Almagro Armenteros, J. J., Sønderby, C. K., Sønderby, S. K., Nielsen, H., & Winther, O. (2019). DeepLoc: prediction of protein subcellular localisation using deep learning. *Bioinformatics*, 35(3), 456–465.
15. Rost, B. (2001). Review: Protein secondary structure prediction continues to rise. *Journal of Structural Biology*, 134(2-3), 204–218.
16. Dingwall, C., & Laskey, R. A. (1991). Nuclear targeting sequences–a consensus? *Trends in Biochemical Sciences*, 16(12), 478–481.
17. Guruprasad, K., Reddy, B. V. B., & Pandit, M. W. (1990). Correlation between stability of a protein and its dipeptide composition: a novel approach for predicting in vivo stability of a protein from its primary sequence. *Protein Engineering, Design and Selection*, 4(2), 155–161.
18. Kyte, J., & Doolittle, R. F. (1982). A simple method for displaying the hydropathic character of a protein. *Journal of Molecular Biology*, 157(1), 105–132.
19. von Heijne, G. (1986). Mitochondrial targeting sequences may form amphiphilic helices. *The EMBO Journal*, 5(6), 1335–1342.
20. Eisenberg, D., Weiss, R. M., & Terwilliger, T. C. (1982). The helical hydrophobic moment: a measure of the amphiphilicity of a helix. *Nature*, 299(5881), 371–374.
21. Koehler, C. M. (2004). New developments in mitochondrial assembly. *Annual Review of Cell and Developmental Biology*, 20, 309–335.
22. von Heijne, G. (1990). The signal peptide. *Journal of Membrane Biology*, 115(3), 195–201.

Appendix: Protein Subcellular Localization Prediction Code

The following code provides the complete implementation for feature extraction, model training, and prediction for protein subcellular localisation. The code is written so that it runs on Google Colab by mounting Google Drive where the FASTA data files should be located.

Listing 1. Complete implementation of protein localisation prediction

```
# Mount Google Drive
import os
from google.colab import drive
drive.mount('/content/drive')

# Install BioPython if not already installed
!pip install biopython

# Set the base directory for the files
base_dir = 'drive/My Drive/COMP0082/'

# Define file paths
file_paths = {
    'cyto': os.path.join(base_dir, 'cyto.fasta.txt'),
    'mito': os.path.join(base_dir, 'mito.fasta.txt'),
    'nuclear': os.path.join(base_dir, 'nuclear.fasta.txt'),
    'secreted': os.path.join(base_dir, 'secreted.fasta.txt'),
    'other': os.path.join(base_dir, 'other.fasta.txt'),
    'blind': os.path.join(base_dir, 'blind.fasta.txt')
}

# Check if all files exist
for location, path in file_paths.items():
    print(f"{location}: {'Exists' if os.path.exists(path) else 'NOT FOUND'}")


# Import necessary libraries
from Bio import SeqIO
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.stats as stats
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import cross_val_predict
from Bio.SeqUtils.ProtParam import ProteinAnalysis
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import StratifiedKFold, cross_val_score
from sklearn.metrics import classification_report, confusion_matrix, matthews_corrcoef, make_scorer
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from collections import Counter


# Define amino acids and properties
amino_acids = ['A', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'K', 'L',
               'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'V', 'W', 'Y']

# Properties of amino acids
aa_properties = {
    'positive': ['K', 'R', 'H'],
    'negative': ['D', 'E'],
    'hydrophobic': ['I', 'V', 'L', 'F', 'C', 'M', 'A', 'W']
}

# Kyte-Doolittle hydrophobicity scale
```

```
kd_scale = {
    'A': 1.8, 'C': 2.5, 'D': -3.5, 'E': -3.5, 'F': 2.8,
    'G': -0.4, 'H': -3.2, 'I': 4.5, 'K': -3.9, 'L': 3.8,
    'M': 1.9, 'N': -3.5, 'P': -1.6, 'Q': -3.5, 'R': -4.5,
    'S': -0.8, 'T': -0.7, 'V': 4.2, 'W': -0.9, 'Y': -1.3
}

def calculate_hydrophobicity(sequence, scale=kd_scale):
    """Calculate average hydrophobicity of a sequence"""
    if not sequence:
        return 0
    return sum(scale.get(aa, 0) for aa in sequence) / len(sequence)

def extract_enhanced_features(sequences_df):
    """Extract enhanced features for protein localization prediction"""
    features_list = []

    for idx, row in sequences_df.iterrows():
        seq_id = row['id']
        sequence = row['sequence']
        label = row.get('label', None)

        feature_dict = {'id': seq_id}

        # 1. Sequence length
        feature_dict['length'] = len(sequence)

        # 2. Global amino acid composition
        aa_count = Counter(sequence)
        for aa in amino_acids:
            feature_dict[f'global_{aa}'] = aa_count.get(aa, 0) / len(sequence)
            * 100 if len(sequence) > 0
            else 0

        # 3. N-terminal and C-terminal composition
        n_term = sequence[:min(50, len(sequence))]
        c_term = sequence[max(0, len(sequence)-50):]

        n_term_count = Counter(n_term)
        c_term_count = Counter(c_term)

        for aa in amino_acids:
            feature_dict[f'nterm_{aa}'] = n_term_count.get(aa, 0) / len(n_term)
            * 100 if len(n_term) > 0
            else 0
            feature_dict[f'cterm_{aa}'] = c_term_count.get(aa, 0) / len(c_term)
            * 100 if len(c_term) > 0
            else 0

        # 4. Basic protein properties
        try:
            standard_seq = ''.join(aa for aa in sequence if aa in amino_acids)
            if standard_seq:
                protein_analysis = ProteinAnalysis(standard_seq)
                feature_dict['isoelectric_point'] = protein_analysis.isoelectric_point()
                feature_dict['molecular_weight'] = protein_analysis.molecular_weight() / 1000  # Scale down for n
                feature_dict['gravy'] = protein_analysis.gravy()  # Hydropathy
            else:
                feature_dict['isoelectric_point'] = 7.0
                feature_dict['molecular_weight'] = 0.0
                feature_dict['gravy'] = 0.0
        except Exception:
            feature_dict['isoelectric_point'] = 7.0
            feature_dict['molecular_weight'] = 0.0
            feature_dict['gravy'] = 0.0

        # 5. Global charge features
```

```
pos_aas = aa_properties['positive']
neg_aas = aa_properties['negative']

pos_count = sum(aa_count.get(aa, 0) for aa in pos_aas)
neg_count = sum(aa_count.get(aa, 0) for aa in neg_aas)

feature_dict['pos_charge_ratio'] = pos_count / len(sequence)
if len(sequence) > 0
else 0
feature_dict['neg_charge_ratio'] = neg_count / len(sequence)
if len(sequence) > 0
else 0
feature_dict['net_charge_ratio'] = (pos_count - neg_count) / len(sequence)
if len(sequence) > 0
else 0

# 6. TARGETED IMPROVEMENT 1: More specific N-terminal features
n_term_20 = sequence[:min(20, len(sequence))]
n_term_30 = sequence[:min(30, len(sequence))]

# Calculate charges in different regions of the protein
for term, suffix in [(n_term_20, '20'), (n_term_30, '30')]:
    term_count = Counter(term)
    term_pos = sum(term_count.get(aa, 0) for aa in pos_aas)
    term_neg = sum(term_count.get(aa, 0) for aa in neg_aas)
    term_len = len(term)

    if term_len > 0:
        feature_dict[f'nterm{suffix}_pos_charge'] = term_pos / term_len
        feature_dict[f'nterm{suffix}_neg_charge'] = term_neg / term_len
        feature_dict[f'nterm{suffix}_net_charge'] = (term_pos - term_neg) / term_len
        feature_dict[f'nterm{suffix}_hydrophobicity'] = calculate_hydrophobicity(term)
    else:
        feature_dict[f'nterm{suffix}_pos_charge'] = 0
        feature_dict[f'nterm{suffix}_neg_charge'] = 0
        feature_dict[f'nterm{suffix}_net_charge'] = 0
        feature_dict[f'nterm{suffix}_hydrophobicity'] = 0

# 7. TARGETED IMPROVEMENT 2: Signal detection
# Nuclear localization signal detection (K/R rich regions)
has_nls = False
for i in range(len(sequence) - 5):
    window = sequence[i:i+6]
    k_count = window.count('K')
    r_count = window.count('R')
    if (k_count + r_count) >= 4:  # 4+ basic residues in 6aa window
        has_nls = True
        break

feature_dict['potential_nls'] = 1 if has_nls else 0

# 8. TARGETED IMPROVEMENT 3: Mitochondrial targeting sequence features
if len(n_term_30) >= 15:
    mts_region = n_term_30[:15]
    mts_pos_count = sum(1 for aa in mts_region if aa in pos_aas)
    mts_hydro_count = sum(1 for aa in mts_region if aa in aa_properties['hydrophobic'])

    feature_dict['mts_pos_charge'] = mts_pos_count / len(mts_region)
    feature_dict['mts_hydrophobic'] = mts_hydro_count / len(mts_region)
    feature_dict['mts_score'] = feature_dict['mts_pos_charge'] * 0.6 +
    feature_dict['mts_hydrophobic'] * 0.4
else:
    feature_dict['mts_pos_charge'] = 0
    feature_dict['mts_hydrophobic'] = 0
    feature_dict['mts_score'] = 0

# 9. TARGETED IMPROVEMENT 4: Signal peptide detection for secreted proteins
```

```
        if len(n_term_30) >= 15:
            # n-region (positive), h-region (hydrophobic)
            n_region = n_term_30[:5]
            h_region = n_term_30[5:15]

            n_region_pos = sum(1 for aa in n_region if aa in pos_aas) / len(n_region)
            if n_region else 0
            h_region_hydro = calculate_hydrophobicity(h_region)

            # Signal peptide score combines n-region charge and h-region hydrophobicity
            feature_dict['sp_score'] = (n_region_pos * 0.3 + (h_region_hydro/5 + 0.8) * 0.7)
        else:
            feature_dict['sp_score'] = 0

        # Add label if available
        if label is not None:
            feature_dict['label'] = label

        features_list.append(feature_dict)

    return pd.DataFrame(features_list)

# Extract enhanced features
print("Extracting enhanced features...")
enhanced_train_features = extract_enhanced_features(train_data)
enhanced_blind_features = extract_enhanced_features(data['blind'])

# Check for NaN values
enhanced_train_features = enhanced_train_features.fillna(0)
enhanced_blind_features = enhanced_blind_features.fillna(0)

# Prepare data
X_enhanced = enhanced_train_features.drop(['id', 'label'], axis=1)
y_enhanced = enhanced_train_features['label']

# Create pipeline with enhanced Random Forest model
rf_model = RandomForestClassifier(
    n_estimators=250,
    max_depth=25,
    min_samples_split=5,
    min_samples_leaf=2,
    class_weight='balanced',
    random_state=42,
    n_jobs=-1
)

pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('classifier', rf_model)
])

# Manual cross-validation to get predictions
print("Performing cross-validation with enhanced features...")
cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
y_pred = np.zeros_like(y_enhanced, dtype=object)
y_proba = np.zeros((len(y_enhanced), len(np.unique(y_enhanced))))

for train_idx, test_idx in cv.split(X_enhanced, y_enhanced):
    X_train, X_test = X_enhanced.iloc[train_idx], X_enhanced.iloc[test_idx]
    y_train, y_test = y_enhanced.iloc[train_idx], y_enhanced.iloc[test_idx]

    # Train the model
    pipeline.fit(X_train, y_train)

    # Make predictions
    y_pred[test_idx] = pipeline.predict(X_test)
```

```
    # Get class indices
    classes = pipeline.classes_
    class_indices = {cls: idx for idx, cls in enumerate(classes)}

    # Store probabilities
    proba = pipeline.predict_proba(X_test)
    for i, idx in enumerate(test_idx):
        for cls in classes:
            cls_idx = class_indices[cls]
            if y_proba[idx, cls_idx] == 0:  # Only update if not set
                y_proba[idx, cls_idx] = proba[i, cls_idx]

# Print results
print("\nEnhanced Model Cross-Validation Results:")
print(classification_report(y_enhanced, y_pred))

# Calculate Matthews Correlation Coefficient
mcc_enhanced = matthews_corrcoef(y_enhanced, y_pred)
print(f"Matthews Correlation Coefficient: {mcc_enhanced:.4f}")

# Confusion Matrix
conf_matrix = confusion_matrix(y_enhanced, y_pred)
plt.figure(figsize=(10, 8))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
            xticklabels=np.unique(y_enhanced),
            yticklabels=np.unique(y_enhanced))
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix - Enhanced Model')
plt.tight_layout()
plt.show()

# Fit final model on all data
print("Training final model...")
pipeline.fit(X_enhanced, y_enhanced)

# Prepare blind test features
X_blind_enhanced = enhanced_blind_features.drop(['id'], axis=1)

# Ensure columns match
X_blind_final = pd.DataFrame(0, index=X_blind_enhanced.index, columns=X_enhanced.columns)
for col in X_blind_enhanced.columns:
    if col in X_enhanced.columns:
        X_blind_final[col] = X_blind_enhanced[col]

# Predict on blind test set
blind_pred = pipeline.predict(X_blind_final)
blind_proba = pipeline.predict_proba(X_blind_final)

# Define confidence levels
def get_confidence_level(probability):
    if probability >= 0.8:
        return "High"
    elif probability >= 0.5:
        return "Medium"
    else:
        return "Low"

# Format blind test predictions
print("\nBlind Test Set Predictions:")
blind_predictions = []

for i, seq_id in enumerate(enhanced_blind_features['id']):
    predicted_class = blind_pred[i]
    max_prob = np.max(blind_proba[i])
    confidence = get_confidence_level(max_prob)
```

```
    class_map = {
        'cyto': 'Cyto',
        'nuclear': 'Nucl',
        'mito': 'Mito',
        'secreted': 'Extr',
        'other': 'Othr'
    }

    formatted_class = class_map.get(predicted_class, predicted_class)
    result_line = f"{seq_id} {formatted_class} Confidence {confidence}"
    blind_predictions.append(result_line)
    print(result_line)

# Save predictions to file
with open('blind_predictions.txt', 'w') as f:
    for line in blind_predictions:
        f.write(line + '\n')

print("\nBlind predictions saved to 'blind_predictions.txt'")
```