الجمهورية الجزائرية الديمقراطية الشعبية

Ministry of Higher Education and Scientific Research

Yahia Fares University of Medea

Faculty of Sciences
Department of Mathematics and Computer Science



Final Master Project

Submitted for Obtaining the Master's Degree in Computer Science

Speciality : Applied Artificial Intelligence

Entitled

Intelligent Analysis of Biological Sequences for Protein Function Classification

---

Presented by :

- ZIRARI Ahlem
- TOUATI Ines

Supervised by :

- Dr. ALI KHOUDJA Meriem

Academic Year : 2025–2026

Acknowledgements

First and foremost, I thank Allah Almighty for His success and guidance, and for granting me patience and determination to complete this work.

I would like to express my sincere gratitude to my supervisor, Dr. Ali Khoudja Mariem, for her valuable guidance, continuous follow-up, and constant support throughout this work, which had a great impact on completing this research.

I also wish to thank all the professors of the Department of Mathematics and Computer Science for the useful knowledge, assistance, and guidance they have provided throughout my academic journey.

I am deeply grateful to my dear parents, may Allah protect them, as well as to my brothers and sister, who played a major role in supporting and encouraging me throughout this academic path.

I would also like to thank my colleague Touati Ines, who was my research partner during this journey.

Finally, I would like to thank all my friends and everyone who, directly or indirectly, contributed to encouraging and supporting me throughout my academic journey.

ZIRARI Ahlem.

Acknowledgements

Praise be to Allah, by whose grace good deeds are completed, and blessings and peace be upon the Master of creation, Muhammad.

I extend my sincere thanks and appreciation to the supervising doctor, Meriem Ali Khoudja, for her continuous follow-up and advice, which contributed to the completion of this thesis.

I would also like to thank all the professors at Yahia Fares University, especially the professors of the Faculty of Sciences, for their academic training throughout our years of study.

I extend my heartfelt gratitude and appreciation to all my esteemed family members, my mother and father, and my brothers Anas, Abdalrahman, and Wassim, for their support throughout my academic journey and for their moral support and continuous encouragement, which had a significant impact on achieving this accomplishment and completing this work.

I would also like to extend my heartfelt thanks to my colleague, Ahlem Zirari, for her constant support and assistance in completing this work.

And finally, I thank myself for the effort and perseverance I have shown throughout the years of study and during my work on this thesis. Thank you.

TOUATI Ines.

Abstract

Protein function classification represents an important research area in bioinformatics. With the rapid growth of protein sequence data, there is an increasing need to develop efficient classification systems based on Artificial Intelligence techniques.
This work aims to classify protein sequences functionally using the Protein Data Bank database, employing Machine Learning and Deep Learning models, in addition to hybrid and ensemble models, based on different protein sequence representation methods. Furthermore, this study leverages protein language models such as ProtBERT and Evolutionary Scale Modeling to extract rich representations of protein sequences.
The models are evaluated using performance metrics such as Accuracy, Precision, Recall, and F1-score, as well as the assessment of model generalization ability. The results demonstrate the effectiveness of the proposed approaches in improving protein function classification performance, particularly the ESM model, which achieved the best performance with an accuracy of 94.95%.
This study contributes to the development of protein function classification tools by providing accurate and effective models based on advanced sequence representations, thereby improving classification accuracy and quality.

Keywords : Protein function classification, Bioinformatics, Artificial Intelligence, Machine Learning, Deep Learning, Protein Language Models.

Résumé

La classification fonctionnelle des protéines constitue un domaine de recherche important en bioinformatique. Avec l'augmentation rapide des données de séquences protéiques, il devient nécessaire de développer des systèmes de classification efficaces basés sur les techniques de l'intelligence artificielle.
Ce travail vise à classifier fonctionnellement les séquences protéiques en utilisant la base de données Protein Data Bank, à l'aide de modèles d'apprentissage automatique et d'apprentissage profond, ainsi que des modèles hybrides et des modèles d'ensemble, en s'appuyant sur différentes méthodes de représentation des séquences protéiques. Par ailleurs, cette étude exploite des modèles de langage protéique tels que ProtBERT et Evolutionary Scale Modeling afin d'extraire des représentations riches des séquences protéiques.
Les modèles sont évalués à l'aide de métriques de performance telles que l'Accuracy, la Précision, le Rappel et le score F1, ainsi que l'évaluation de la capacité de généralisation des modèles. Les résultats obtenus démontrent l'efficacité des approches proposées dans l'amélioration des performances de classification des fonctions des protéines, en particulier le modèle ESM, qui a obtenu la meilleure performance avec une accuracy de 94.95 %.
Cette étude contribue au développement d'outils de classification des fonctions protéiques en proposant des modèles précis et efficaces basés sur des représentations avancées des séquences, permettant ainsi d'améliorer la performance et la qualité de la classification.

Mots-clés : classification fonctionnelle des protéines, bioinformatique, intelligence artificielle, apprentissage automatique, apprentissage profond, modèles de langage protéique.

الملخص

يشكّل تصنيف وظائف البروتينات مجالًا بحثيًا مهمًا في المعلوماتية الحيوية، ومع التزايد الكبير في البيانات البروتينية، أصبح من الضروري تطوير أنظمة تصنيف فعالة تعتمد على تقنيات الذكاء الاصطناعي.

يهدف هذا العمل إلى تصنيف تسلسلات البروتين وظيفيًا اعتمادًا على قاعدة بنك البيانات البروتينية (PDB) باستخدام نماذج التعلم الآلي والتعلم العميق، إضافةً إلى النماذج الهجينة والنماذج التجميعية، وذلك بالاعتماد على طرق مختلفة لتمثيل السلاسل البروتينية.

كما تسعى هذه الدراسة إلى الاستفادة من نماذج اللغة البروتينية مثل ProtBERT وESM لاستخراج تمثيلات غنية للتسلسلات البروتينية.

يتم تقييم النماذج باستخدام مقاييس الأداء مثل الدقة، والدقة الإيجابية، والاسترجاع، ودرجة F1، بالإضافة إلى تقييم قدرة النماذج على التعميم. اظهرت النتائج فعالية المقاربات المقترحة في تحسين أداء تصنيف وظائف البروتينات، خاصة نموذج ESM الذي حقق افضل اداء بدقة بلغت 94.95%.

تساهم نتائج هذه الدراسة في تطوير برامج التصنيف الوظيفي للبروتينات من خلال توفير نماذج فعالة ودقيقة تعتمد على تمثيلات متقدمة للتسلسلات، مما يساعد على تحسين دقة التصنيف وجودته.

الكلمات المفتاحية: تصنيف وظائف البروتينات، المعلوماتية الحيوية، الذكاء الاصطناعي، التعلم الالي، التعلم العميق، نماذج اللغة البروتينية.

Contents

List of Tables **iv**

List of Figures **vi**

General Introduction **1**

Chapter 1: Fundamentals and State of the Art **4**
1.1 Introduction . . . 4
1.2 Background Knowledge . . . 4
1.2.1 Biological Foundations of Proteins . . . 4
1.2.1.1 Definition and Role of Proteins . . . 4
1.2.1.2 Amino Acids . . . 5
1.2.1.3 Protein Structure . . . 5
1.2.1.4 Protein Functions . . . 6
1.2.1.5 Relationship between Structure and Function . . . 6
1.2.2 Protein Functional Classification . . . 6
1.2.2.1 Importance of Protein Classification . . . 6
1.2.2.2 Enzyme Classification System . . . 7
1.2.2.3 Gene Ontology System . . . 7
1.2.2.4 Traditional Methods for Protein Classification . . . 7
1.2.2.5 Challenges in Functional Classification . . . 8
1.3 Computational Protein Function Classification . . . 8
1.3.1 Sequence-based Methods . . . 8
1.3.2 Structure-based Methods . . . 8
1.4 Feature Representation of Protein Sequences . . . 8
1.4.1 Importance of Feature Representation . . . 8
1.4.1.1 Feature Extraction Techniques . . . 9
1.4.1.2 Sequence Encoding Methods . . . 9
1.5 Fundamentals of Machine Learning . . . 9
1.5.1 Definition of Machine Learning . . . 10
1.5.2 Supervised Learning . . . 10
1.5.3 Classification Principles . . . 10
1.5.4 Common Algorithms . . . 10
1.6 Fundamentals of Deep Learning . . . 11
1.6.1 Definition of Deep Learning . . . 11
1.6.2 Neural Networks Basics . . . 12
1.6.2.1 Artificial Neural Networks . . . 12
1.6.2.2 Activation Functions . . . 12
1.6.2.3 Loss Functions . . . 13

1.6.2.4 Optimization . . . 13
1.6.2.5 Backpropagation . . . 13
1.6.3 Deep Learning Architectures . . . 13
1.6.3.1 Convolutional Neural Networks . . . 13
1.6.3.2 Recurrent Neural Networks . . . 15
1.6.3.3 Long Short-Term Memory . . . 15
1.6.3.4 Gated Recurrent Units . . . 16
1.6.3.5 Transformer . . . 17
1.6.3.6 Attention . . . 18
1.6.3.7 Bidirectional Encoder Representations from Transformers 18
1.6.3.8 Hybrid Models . . . 19
1.6.4 Representation Learning . . . 19
1.6.5 Transfer Learning . . . 19
1.6.6 Regularization and Prevention Methods . . . 20
1.7 Model Evaluation . . . 20
1.7.1 Accuracy : . . . 20
1.7.2 Precision : . . . 20
1.7.3 Recall : . . . 21
1.7.4 F1-score : . . . 21
1.7.5 Confusion Matrix : . . . 21
1.8 Literature Review . . . 21
1.9 Conclusion . . . 25

Chapter 2: Research Methodology **27**
2.1 Introduction . . . 27
2.2 Research Overview . . . 27
2.3 Processing Flow . . . 29
2.3.1 Data Acquisition and Preparation . . . 29
2.3.1.1 Dataset Description . . . 29
2.3.1.2 Data Preprocessing . . . 29
2.3.1.3 Class Distribution and Imbalance Analysis . . . 30
2.3.2 Feature Engineering and Sequence Representation . . . 31
2.3.2.1 Feature Extraction for Machine Learning . . . 31
2.3.2.2 Sequences Representation . . . 32
2.3.3 Pretrained Embeddings . . . 33
2.3.3.1 ProtBERT . . . 33
2.3.3.2 Evolutionary Scale Modeling . . . 33
2.4 Model Design and Development . . . 34
2.4.1 Machine Learning Models . . . 34
2.4.2 Deep Learning Models . . . 35
2.4.2.1 Deep Learning Architectures . . . 35
2.4.2.2 Hyperparameter Tuning and Configuration . . . 39
2.4.3 Hybrid Models . . . 39
2.4.3.1 Hybrid Models Based on Raw Sequences Representation . 39
2.4.3.2 Hybrid Models Based on k-mer Representation . . . 43
2.4.4 Hybrid Ensemble Learning . . . 48
2.4.5 Transfer Learning . . . 49
2.5 Training Strategy . . . 51

2.6 Evaluation Methodology . . . 51
2.6.0.1 Evaluation Metrics and Confusion Matrix . . . 51
2.6.0.2 Cross-Validation Strategy . . . 51
2.6.0.3 Generalization Evaluation . . . 52
2.7 Conclusion . . . 52

Chapter 3: Implementation Framework and Experimental Analysis 54
3.1 Introduction . . . 54
3.2 Experimental Design . . . 54
3.2.1 Development Environment . . . 54
3.2.1.1 Programming Language . . . 54
3.2.1.2 Libraries and Frameworks . . . 54
3.2.1.3 Tools . . . 55
3.2.2 Hardware Specifications . . . 55
3.3 Experimental Results . . . 55
3.3.1 Machine Learning Models Results . . . 55
3.3.2 Deep Learning Models Results . . . 56
3.3.2.1 Models Results Based on One-Hot Encoding Representations . . . 56
3.3.2.2 Models Results Based on K-mer Representations . . . 57
3.3.3 Hybrid Models Results . . . 58
3.3.3.1 Models Results Based on Raw Sequences . . . 58
3.3.3.2 Models Results Based on K-mer Representations . . . 59
3.3.4 Hybrid Ensemble Learning Models Results . . . 60
3.3.5 Transfer Learning Models Results . . . 61
3.3.5.1 Results Using ProtBERT Embeddings . . . 61
3.3.5.2 Results Using ESM Embeddings . . . 62
3.3.5.3 ESM Fine-Tuned Model Results . . . 63
3.4 Comparative Analysis and Discussion . . . 63
3.4.1 Intra-Study Model Comparison . . . 63
3.4.2 Interpretation of Results . . . 64
3.4.3 Impact of Feature Representation . . . 65
3.4.4 Overfitting Analysis and Generalization . . . 66
3.4.4.1 Overfitting Analysis . . . 66
3.4.4.2 Cross Validation Results . . . 67
3.4.4.3 External Dataset Evaluation . . . 68
3.4.4.4 Generalization Discussion . . . 68
3.4.5 Error Analysis . . . 68
3.4.6 Impact of Dataset Composition . . . 69
3.4.7 Model Complexity and Computational Cost . . . 70
3.4.8 Comparison with Related Work . . . 71
3.4.9 Final Summary . . . 72
3.5 Conclusion . . . 73

General Conclusion 75

Bibliography 76

List of Tables

1.1 Summary of Related Works in Protein Sequence Classification. 24
2.1 Hyperparameter Settings of Machine Learning Models. 35
2.2 Hyperparameter Settings of Deep Learning Models. 39
2.3 Hyperparameter Settings of Hybrid Models Based on Raw Sequences Representation. 43
2.4 Hyperparameter Settings of Hybrid Models Based on k-mer Representation. 47
2.5 Hyperparameter Settings of Hybrid Ensemble Learning Models. 49
2.6 Hyperparameter Settings of Machine Learning Models with ProtBERT and ESM Embeddings. 50
2.7 Hyperparameter Settings for the Fine-Tuned ESM Model 50
3.1 Performance Results of Machine Learning Models. 55
3.2 Performance Results of CNN Model Based on One-Hot Encoding. 56
3.3 Performance Results of Deep Learning Models Based on k-mer Representation. 57
3.4 Performance Results of Hybrid Learning Models Based on Raw Sequences Representation. 58
3.5 Performance Results of Hybrid Learning Models Based on k-mer Representation. 59
3.6 Performance Results of Hybrid Ensemble Learning Models. 60
3.7 Performance Results of Machine Learning Models Using ProtBERT Embeddings. 61
3.8 Performance Results of Machine Learning Models Using ESM Embeddings. 62
3.9 Performance Results of the Fine-Tuned ESM Model. 63
3.10 Cross-Validation Results of the Fine-Tuned ESM Model. 67
3.11 Evaluation Results of the Fine-Tuned ESM Model on External Test Data. 68
3.12 Comparison with Related Works. 72

List of Figures

1.1 Levels of protein structure 5
1.2 Artificial Neural Networks 12
1.3 Convolutional Neural Network (CNN) Architecture 14
1.4 One-Dimensional Convolutional Neural Network (1D-CNN) Architecture . 14
1.5 Recurrent Neural Network (RNN) Architecture 15
1.6 Long Short-Term Memory (LSTM) 16
1.7 Bidirectional Long Short-Term Memory (BiLSTM) Architecture 16
1.8 Gated Recurrent Units (GRUs) Architecture 17
1.9 Bidirectional Gated Recurrent Unit (BiGRU) Architecture 17
1.10 Transformer Architecture 18
1.11 Bidirectional Encoder Representations from Transformers (BERT) Architecture 19
1.12 Confusion Matrix 21
2.1 Protein Sequence Classification Pipeline. 28
2.2 Amino Acid Frequency Distribution. 30
2.3 Class Distribution in the Dataset 31
2.4 Feature Engineering and Sequence Representation Methods 32
2.5 Sequence Length Distribution in the Dataset. 33
2.6 Evolutionary Scale Modeling (ESM) Architecture 34
2.7 Proposed CNN Architecture. 36
2.8 Proposed LSTM Architecture. 37
2.9 Proposed BiLSTM Architecture. 38
2.10 Proposed Enhanced Transformer Encoder Architecture. 39
2.11 Proposed CNN-BiLSTM Architecture. 40
2.12 Proposed CNN-BiGRU Architecture. 41
2.13 Proposed CNN-BiLSTM-Attention Architecture. 42
2.14 Proposed CNN-BiGRU-Attention Architecture. 43
2.15 Proposed CNN-BiLSTM Architecture. 44
2.16 Proposed CNN-BiGRU Architecture. 45
2.17 Proposed CNN-Self-Attention Architecture. 46
2.18 Proposed CNN-BiLSTM-Attention Architecture. 46
2.19 Proposed CNN-BiGRU-Attention Architecture. 47
2.20 Hybrid Ensemble Learning Pipeline. 49
3.1 Results of Machine Learning Models. 56
3.2 Results of Deep Learning Models Based on k-mer Representation. 57
3.3 Results of Hybrid Models Based on Raw Sequences Representation. 58
3.4 Results of Hybrid Learning Models Based on K-mer Representation. 59
3.5 Results of Hybrid Ensemble Learning Models 60

3.6 Results of Machine Learning Models Performance based on ProtBERT Embeddings. 61
3.7 Results of Machine Learning Models Performance based on ESM Embeddings. 62
3.8 Comparison of the Best Models Across Different Proposed Methods. 64
3.9 Comparison of the Best Models Results Across Different Representation Methods. 65
3.10 Training and Validation Curves of the Fine-Tuned ESM Model Showing Accuracy and Loss Evolution. 67
3.11 Confusion Matrices of the Fine-Tuned ESM Model on PDB and External Datasets. 69
3.12 Confusion Matrices of the CNN-BiLSTM-Attention Model. 70
3.13 Final Summary. 73

List of Acronyms

AI Artificial Intelligence

ANNs Artificial Neural Networks

BERT Bidirectional Encoder Representations from Transformers

BiGRU Bidirectional Gated Recurrent Unit

BiLSTM Bidirectional Long Short-Term Memory

BLAST Basic Local Alignment Search Tool

CNN Convolutional Neural Network

DL Deep Learning

DT Decision Tree

EC Enzyme Classification

ETE Enhanced Transformer Encoder

GO Gene Ontology

GRU Gated Recurrent Unit

HMM Hidden Markov Model

KNN K-Nearest Neighbors

LR Logistic Regression

LSTM Long Short-Term Memory

ML Machine Learning

NB Naïve Bayes

NCBI National Center for Biotechnology Information

1D-CNN One-Dimensional Convolutional Neural Network

PDB Protein Data Bank

RF Random Forest

RCSB Research Collaboratory for Structural Bioinformatics

SVM Support Vector Machine

TF-IDF Term Frequency-Inverse Document Frequency

General Introduction

Proteins are among the most essential biomolecules in living organisms, playing critical roles in a wide range of fundamental biological processes, including enzymatic catalysis, cellular signaling, and immune defense. Protein research has therefore attracted significant attention, particularly in the field of protein function classification, which aims to associate proteins with their corresponding biological role.

Protein function classification plays a crucial role in understanding complex biological mechanisms, elucidating various biological phenomena, and facilitating the development of novel therapeutic treatments and pharmaceutical drugs.

The rapid advances in biotechnology have led to a tremendous increase in the number of protein sequences. Despite the continuous growth of biological data, many proteins still lack accurate functional annotation. Experimental methods used to determine protein functions are highly costly in terms of time and resources and require extensive laboratory efforts, making them unable to keep pace with the increasing volume of data. In addition, traditional computational approaches rely heavily on sequence similarity principles for classification, which limits their performance and reduces their effectiveness.

Artificial Intelligence has demonstrated its effectiveness in addressing complex problems across various fields, including bioinformatics. It relies on Machine Learning and Deep Learning techniques that are capable of automatically learning and extracting complex features and patterns from data, making it a promising approach for protein classification and an effective alternative to traditional experimental and computational methods. Furthermore, it has contributed to achieving accurate and promising results in protein function classification tasks.

This study aims, in general, to explore and evaluate the effectiveness of a wide range of Artificial Intelligence approaches for protein function classification. More specifically, it seeks to compare several Machine Learning and Deep Learning models using different protein sequence representation techniques. In addition, this work investigates the use of large protein language models to enhance classification performance.

The main contributions of this work include a unified comparative study of multiple learning paradigms (machine learning, deep learning, hybrid, ensemble, and transfer learning), along with an extensive analysis of protein sequence representations from classical methods to advanced pretrained embeddings (ProtBERT and ESM). The study also proposes and evaluates novel deep and hybrid architectures, as well as ensemble strategies combining deep models with classical classifiers to improve robustness. In addition, it validates transfer learning through fine-tuning, assesses generalization using cross-validation and external datasets, and provides error analysis alongside a systematic evaluation of model complexity, computational cost, and performance.

This study adopted a structured methodology comprising a series of integrated stages, starting with data collection and preprocessing, followed by model development and evaluation, and culminating in results analysis and the assessment of classification effectiveness.

This dissertation is organized into three main chapters :

Chapter 1: Fundamentals and State of the Art :

This chapter presents the theoretical background of the study. It covers the fundamental concepts of proteins and their functions, along with protein functional classification. It

also introduces the main principles of Machine Learning and Deep Learning. Finally, it provides a review and analysis of related previous studies in the field.

Chapter 2: Research Methodology :

This chapter describes the research methodology adopted for protein sequence classification, including the dataset, preprocessing stages, feature engineering, and pretrained embeddings. It then describes the developed machine learning, deep learning, hybrid, ensemble, and transfer learning models, and concludes with the training strategy and evaluation methodology.

Chapter 3: Implementation Framework and Experimental Analysis :

This chapter is dedicated to presenting the experimental framework and results of the study, including the setup, development environment, and hardware specifications. It compares the performance of the different proposed models using various feature representations and embeddings. It also provides a comparative analysis of generalization, overfitting, error analysis, and computational cost, before concluding with a summary of the findings.

Chapter 1
Fundamentals and State of the Art

Chapter 1

Fundamentals and State of the Art

1.1 Introduction

Proteins are among the most important biomolecules in living organisms because they play a fundamental role in various biological processes, such as cell and tissue formation, signal transmission, and immune regulation. With the continuous discovery of protein sequences and the increasing number and diversity of functions, their classification has become a major challenge in the field of bioinformatics.
The field of protein function classification has witnessed significant progress in recent years, thanks to advances in artificial intelligence techniques, especially machine learning and deep learning. These methods are capable of extracting complex patterns from protein sequences and classifying them with high accuracy compared with traditional methods.
This chapter reviews key biological concepts related to proteins, methods for representing protein sequences, as well as the basic concepts related to machine learning and deep learning used in this field. It also reviews previous studies that applied these techniques to protein sequence classification.

1.2 Background Knowledge

This section provides the fundamental knowledge required to understand protein-related concepts by presenting the biological foundations in terms of protein composition, structure, and function. It also covers the concept of protein classification and its importance, in addition to the systems and methods used in this field.

1.2.1 Biological Foundations of Proteins

1.2.1.1 Definition and Role of Proteins

The term ”protein” originates from the Greek word Proteios, which means ”essential” or ”of first importance”. It was introduced by the Dutch chemist Gerardus Johannes Mulder in 1838, and later popularized by the Swedish chemist Jöns Jacob Berzelius. This term reflects the great importance of proteins in all living organisms. Proteins are essential biological macromolecules composed of one or more chains of amino acids linked by peptide bonds. These amino acids differ from each other in their side chains (R), which give them specific chemical and functional properties [1].
Proteins play a vital role in various biological processes such as the stimulation of metabolic reactions, the synthesis, transformation, and breakdown of chemicals, transport of oxygen in the blood, regulation of hormones and enzymes, immune system response, blood clotting and maintenance of the structural integrity of cells and tissues [2].

1.2.1.2 Amino Acids

Amino Acids are the basic structural units of proteins, which consist of a central alpha carbon atom bonded to a hydrogen atom, an amino group, a carboxyl group and a side chain called the R group. In nature, there are 20 amino acids that differ only in the chemical structure of their side chain (R), and they are linked together by peptide bonds to form polypeptide chains. These amino acids are : Alanine (A), Arginine (R), Asparagine (N), Aspartic acid (D), Cysteine (C), Glutamine (Q), Glutamic acid (E), Glycine (G), Histidine (H), Isoleucine (I), Leucine (L), Lysine (K), Methionine (M), Phenylalanine (F), Proline (P), Serine (S), Threonine (T), Tryptophan (W), Tyrosine (Y) and Valine (V) [3].

1.2.1.3 Protein Structure

Protein Structure is the shape and organization adopted by a sequence of amino acids within a molecule after it folds in space. There are four levels of protein structure [5] :

- **Primary structure** is represented by the amino acid sequence.
- **Secondary structure :** which is the chain that makes consist of (α) helices with a helical structure and (β) sheets with a sheet-like structure.
- **Tertiary structure :** is the complete three-dimensional organization of a polypeptide chain.
- **Quaternary structure :** is the structure of a protein composed of more than one polypeptide chain.

The following Figure 1.1 represents the levels of protein structure.

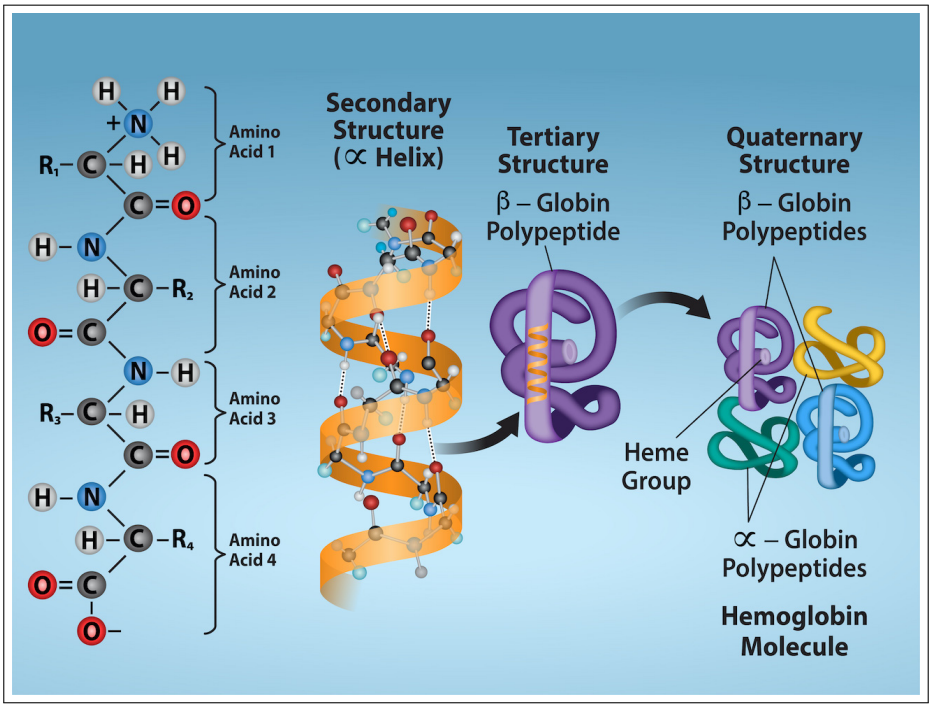


Figure 1.1 : Levels of protein structure [3].

1.2.1.4 Protein Functions

Protein function refers to the specific biological role performed by a protein within the cells of living organisms, describing its activity or role at the molecular or cellular level. Proteins are classified into different functional groups, where one protein can belong to one or more of these groups to determine its final function. These groups are [4] :

- **Enzymes :** Enzymes catalyze biochemical reactions within cells and facilitate chemical processes.
- **Structural proteins :** They form and maintain cellular structure, and also regulate the distribution of organelles within the cytoplasm.
- **Storage proteins :** Their role is to store nutrients, especially amino acids and metal ions, for use when the cell needs them.
- **Signalling proteins :** They are found in all parts of cells and are essential for their functions, as they enable communication between cells and response to different signals.
- **Transporter proteins :** They allow the movement of various molecules within cells across the plasma membrane, and also transport molecules between cells in multicellular organisms.
- **Regulatory proteins :** They help maintain internal balance by controlling the activity of proteins and vital processes, to ensure they work at the right time and place.
- **Defence proteins :** They are found in many living organisms and protect them from various external attacks.

1.2.1.5 Relationship between Structure and Function

The function of a protein is closely related to its structure, as the amino acid sequence determines how the protein folds and forms its three-dimensional structure, which in turn determines its function within the cell [6]. Therefore, any change in the amino acid sequence may lead to a change in the three-dimensional structure, and consequently to a loss or modification of function.

1.2.2 Protein Functional Classification

Protein functional classification is a method of organizing proteins according to their biological roles within the cells of living organisms.

1.2.2.1 Importance of Protein Classification

The importance of functional classification of proteins lies in several key aspects :

- **Understanding biological functions :** Protein classification helps researchers understand the functions of proteins within cells and more accurately determine their properties and role.

- **Predicting the functions of new proteins :** Classifying an unknown protein into a specific functional group helps predict its function and characteristics.
- **Medical and pharmaceutical applications :** It enables researchers to develop drugs and treatments by understanding the essential proteins associated with diseases.
- **Improving nutritional health :** It helps in assessing nutritional value by determining the physical and chemical properties of proteins used in the food industry, thus improving nutritional quality.
- **Facilitating and understanding scientific research :** It saves time and effort by collecting proteins with similar properties into groups, making them easier to study.

1.2.2.2 Enzyme Classification System

The Enzyme Classification (EC) is a standard system used to classify enzymes based on the reactions they catalyze. It uses a four-part numerical code for each enzyme, reflecting its functional properties. It is divided into six main classes including : Oxidoreductases, Transferases, Hydrolases, Lyases, Isomerases, and Ligases [7].

1.2.2.3 Gene Ontology System

The Gene Ontology (GO) system is a structured terminology system used to describe gene and protein functions in a standardized way. Its terms are interconnected through hierarchical relationships similar to a graph representation, allowing for a more precise description of biological information [8]. It is divided into three main categories [9] :

- **Molecular Function :** refers to the activity performed by a protein at the molecular level.
- **Cellular Component :** determines the location of protein action within cells.
- **Biological Process :** represents the biological activity in which the protein is involved to achieve a specific biological objective.

1.2.2.4 Traditional Methods for Protein Classification

A. Basic Local Alignment Search Tool : Basic Local Alignment Search Tool (BLAST) is one of the most widely used tools in the field of bioinformatics. It was developed by the National Center for Biotechnology Information (NCBI) and is still undergoing continuous improvements. It is used to analyze and compare biological sequences by identifying regions of similarity between them to help predict the functions of newly discovered or unknown genes and proteins [10].

B. Hidden Markov Model : The Hidden Markov Model (HMM) is a statistical model used to model and analyze sequential data, such as biological sequences. This model is based on the assumption of hidden states that control the generation of observable data, where each state depends only on the previous state [11].

1.2.2.5 Challenges in Functional Classification

There are several difficulties and obstacles that face the process of accurately determining protein function based on biological data. The main challenges in protein classification include[12] :

- The presence of a large number of proteins that are very similar in sequence but differ in function.
- Lack of features related to function.
- A single protein may perform multiple functions at the same time.
- The absence of three-dimensional structures for many proteins.
- The presence of small parts of the sequence responsible for function makes prediction difficult, especially when they are located at the end of the sequence.
- Lack of experimentally confirmed functional data.

1.3 Computational Protein Function Classification

It aims to identify and classify proteins using computational algorithms and methods that replace experimental laboratory approaches.

1.3.1 Sequence-based Methods

Sequence-based methods for protein function prediction rely on protein sequences, by extracting features and directly capturing biological patterns from the sequences [14].

1.3.2 Structure-based Methods

Structure-based methods focus on extracting features related to biological function by analyzing the spatial structure of proteins and converting them into contact maps, and protein sequences are often used alongside structural data as inputs for these methods [14].

1.4 Feature Representation of Protein Sequences

Protein sequence representation is the process of converting amino acid sequences into numerical formats that can be processed by computational models using various techniques.

1.4.1 Importance of Feature Representation

Feature representation is a fundamental step in sequence classification, as it directly influences the performance of the models used. Its importance includes :

- Enabling models to extract patterns and properties from sequences.
- Capturing relationships within sequences.
- Improving the quality and accuracy of learning and performance.
- Reducing dimensionality and removing noise.

1.4.1.1 Feature Extraction Techniques

Several techniques are used for feature extraction from protein sequences, including :

- **K-mer :** Represents biological sequences as numerical vectors by calculating the frequency of short segments within the sequence. The dimensionality of the representation varies depending on the value of k, which allows capturing local patterns within the sequences [13].
- **N-gram :** Represents sequences by dividing them into successive subsequences of length n, which are used to capture local patterns and contextual information within the sequence.
- **Term Frequency-Inverse Document Frequency (TF-IDF) :** Represents sequences by assigning weights to each part of the sequence based on N-gram or K-mer division, TF calculates the frequency of the part within the sequence, while IDF measures its rarity across all sequences, with the aim of highlighting the important and distinctive parts within each sequence. TF-IDF is mathematically defined by the following equation :

$$TF\text{-}IDF(t,d) = TF(t,d) \times \log\left(\frac{N}{DF(t)}\right) \tag{1.1}$$

  where t represents the extracted part from the protein sequence, and d represents a protein sequence within the dataset. N denotes the total number of sequences in the dataset, while DF(t) represents the number of sequences in which the element t appears.
- **CountVectorizer :** Divides sequences into subsequences (k-mers/n-grams) which are treated as words, where the frequency of each k-mer/n-gram is calculated across all sequences, resulting in a numerical vector for each protein that represents the distribution of patterns within a sequence.

1.4.1.2 Sequence Encoding Methods

There are several encoding techniques used for protein sequences, among them, we mention :

- **Integer Encoding :** A method for numerically representing amino acid sequences, where each amino acid is assigned a specific numerical value using a mapping table. The protein sequence is then converted from its letter form into a numerical sequence while preserving the order of the elements.
- **One-Hot Encoding :** Represents sequences numerically by converting each amino acid into a binary vector, where only one position contains the value 1 corresponding to that amino acid, while the remaining values are 0. Thus, the entire protein sequence is represented as a matrix.

1.5 Fundamentals of Machine Learning

Machine learning is considered one of the fundamental techniques used in protein classification, as it provides the ability to efficiently handle protein data.

1.5.1 Definition of Machine Learning

Machine Learning (ML) is a branch of Artificial Intelligence (AI) that aims to create models capable of learning from data and making decisions or predictions without direct human intervention. It relies on algorithms trained on specific datasets, which analyze data and discover patterns within it, allowing the system to generalize to new data and produce accurate results [15]. Machine learning is used in many applications, such as image and face recognition, as well as voice assistants. It is also considered a fundamental part of modern artificial intelligence and is applied in several fields, especially intelligent systems and modern language models.

1.5.2 Supervised Learning

Supervised Learning is a type of machine learning algorithm that relies on labeled data containing known inputs and outputs. The dataset is divided into training data and testing data, with the goal of building a model capable of making predictions and decisions. Supervised Learning is used in classification problems dealing with discrete values, and regression problems used to predict continuous values [16].

1.5.3 Classification Principles

Classification is one of the supervised learning tasks in machine learning, where the model assigns input data to predefined classes or categories based on the features and patterns it learns from the training data. Classification problems are divided into three types :

- **Binary classification :** which includes only two categories.
- **Multi-class classification :** which involves more than two categories.
- **Multi-label classification :** where a single instance can belong to more than one class at the same time.

The success of classification depends on the quality of the data, the choice of the appropriate algorithm, and the model parameters. However, two main issues may occur during training :

- **A/Overfitting :** occurs when the model learns the details of training data, including noise and irrelevant patterns, which leads to poor generalization on new unseen data.
- **B/Underfitting :** occurs when the model is too simple and unable to represent the data, resulting in poor performance on both training and test data.

1.5.4 Common Algorithms

The following section presents several widely used machine learning algorithms for classification tasks :

- **Support Vector Machine (SVM) :** is a machine learning model used for classification problems. Its main idea is to find the optimal separating hyperplane between classes, such that the margin between them is maximized, allowing better separation between the elements of each class while achieving a high generalization capability [59].

- **Naïve Bayes (NB) :** is a machine learning classification model that determines whether an instance belongs to a specific class or not, based on conditional probability, assuming feature independence. This allows building a probabilistic model based on Bayes theorem [60].

- **K-Nearest Neighbors (KNN) :** is an instance-based machine learning method, in which the algorithm predicts the class of new data based on its similarity to instances in the training data. This is done by measuring similarity between data instances and assigning a class based on the nearest neighbors [61].

- **Decision Tree (DT) :** It is a supervised learning model used for classification tasks. This model builds a predictive representation that relates observations to their corresponding target outcomes. A decision tree consists of internal nodes that represent data features and branches that represent decision rules, while the terminal nodes (leaves) represent the final classes or output values [62].

- **Logistic Regression (LR) :** Logistic Regression is a classification algorithm that models the relationship between one or more independent variables and a binary outcome by estimating the probability that an observation belongs to a particular class [63].

- **Random Forest (RF) :** is an ensemble learning method that combines multiple learning models based on the Bagging technique. A set of decision trees is trained on different randomly sampled subsets of the data. The outputs of the trees are combined through voting in the case of classification or by averaging in the case of regression, which helps improve the model's performance [64].

- **XGBoost :** It is an ensemble learning method based on the Tree Boosting technique. This approach combines a set of decision trees to build a strong model, where each model attempts to correct the errors of the previous one in order to improve the overall performance [65].

1.6 Fundamentals of Deep Learning

Deep Learning (DL) is one of the modern techniques that plays a fundamental role in protein functional classification. This section presents the fundamental concepts of deep learning, in addition to the most popular models and techniques used for processing protein sequences.

1.6.1 Definition of Deep Learning

Deep learning is a branch of machine learning and one of the most widely used AI techniques. It is based on neural networks composed of multiple layers, where each layer gradually interprets the input data. This gives it the ability to automatically learn features for different tasks and link them to the outputs without any manual intervention, which provides high performance and accuracy, especially with large-scale data. Deep learning is used in many fields, such as computer vision, image and audio processing, data analysis and natural language processing [17].

1.6.2 Neural Networks Basics

Neural networks are one of the most important deep learning techniques. They rely on layers of artificial neurons to extract features from data. They are widely used in classification tasks, including protein classification.

1.6.2.1 Artificial Neural Networks

Artificial Neural Networks (ANNs) are computational systems inspired from the structure and function of the human brain, used to process data and learn patterns in order to solve complex problems. They consist of interconnected neurons organized into layers, including ***input layers*** that receive external data such as images and text, ***hidden layers*** that process data and learn patterns and relationships using weights and activation functions, and ***output layers*** that produce the final output, such as classification or a predicted value [18]. Figure 1.2 below represents the general architecture of artificial neural networks.

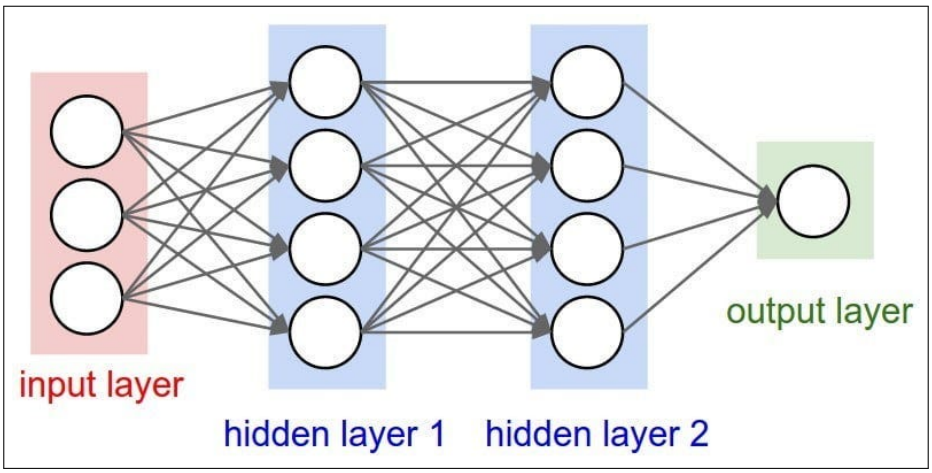


Figure 1.2 : Artificial Neural Networks [31].

1.6.2.2 Activation Functions

Activation functions are one of the most important components of a neural network, as they enable neural networks to learn complex relationships in data by introducing non-linearity [19]. The most common activation functions are :

- **ReLU :** converts negative values to 0, while keeping positive values unchanged.

$$f(x) = \max(0, x) \tag{1.2}$$

- **Tanh :** transforms values into the range [-1, 1] and is centered at 0.

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{1.3}$$

- **Sigmoid :** converts values into the range [0, 1] and is often used in the output layer for binary classification or multi-label classification tasks.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{1.4}$$

- **Softmax :** converts values into a probability distribution in the range [0, 1] where the sum of all probabilities equals 1, It is used in the output layer for multi-class classification problems.

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}} \tag{1.5}$$

1.6.2.3 Loss Functions

Neural networks rely on loss functions that measure the difference between the true and predicted values. Depending on the type of problem, Cross Entropy is used in classification tasks and includes two types : Binary Cross Entropy, used for binary classification and multi-label classification problems, and Categorical Cross Entropy, used for multi-class classification problems.

1.6.2.4 Optimization

Optimization is the process of updating model parameters by finding their optimal values, and it aims to minimize the loss function to the lowest possible value. The most important optimization algorithms are [19] :

- **Gradient Descent (GD) :** Updates the weights in the opposite direction of the gradient, to minimize the error.
- **Stochastic Gradient Descent (SGD) :** An improved version of GD that uses small batches of samples to update the weights, accelerating the training process.
- **ADAM :** It computes adaptive learning rates for each parameter and updates them individually. It is efficient and speeds up training. This has made it the most widely used algorithm in deep learning tasks.

1.6.2.5 Backpropagation

The backpropagation algorithm is used in the training of neural network models to compute and distribute the error from the output layer back to the input layer. Its goal is to determine the contribution of each weight to the error and update the weights to reduce the overall error [19].

1.6.3 Deep Learning Architectures

Deep learning architectures rely on artificial neural networks and vary depending on the nature of the data and the target task. These architectures include several different models used to extract features and patterns from data, which allows improving performance in various learning tasks.

1.6.3.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are considered supervised deep learning algorithms that were developed by LeCun et al. in 1998 [20]. They are used to process images by extracting the most important features and characteristics and classifying them accurately, through the following stages :

Convolution layer : responsible for feature extraction, and applies filters (kernels) to detect patterns in input data.
Pooling layer : reduces the spatial dimensions and preserves the most important features, thus reducing computational complexity. The most commonly used techniques in this layer are max pooling and average pooling.
Flatten layer : this layer transforms the extracted feature maps into a one-dimensional vector.
Fully Connected layer : it combines the extracted features to make the final decision (classification) [21]. Figure 1.3 illustrates the architecture of Convolutional Neural Network.

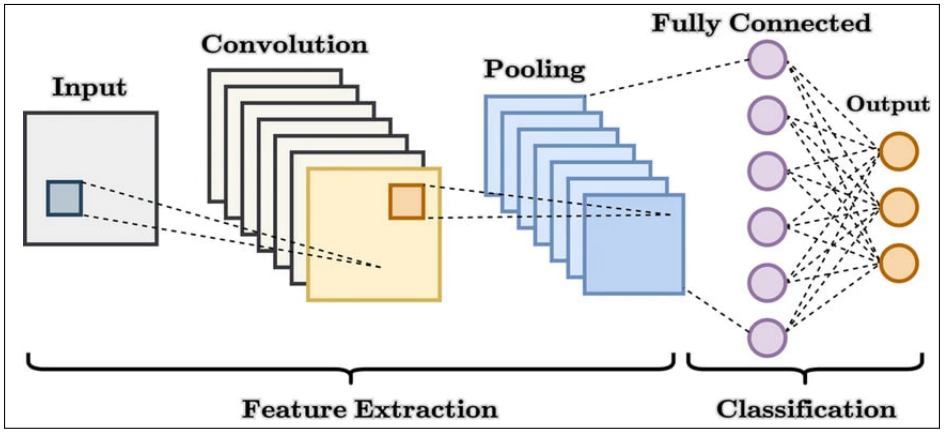


Figure 1.3 : Convolutional Neural Network (CNN) Architecture [32].

CNN is also used with text and time series data, known as **One-Dimensional Convolutional Neural Network (1D-CNN)**, where it can effectively extract patterns and detect relationships within sequences in a manner similar to standard CNN [22]. Figure 1.4 shows the architecture of One-Dimensional Convolutional Neural Network.

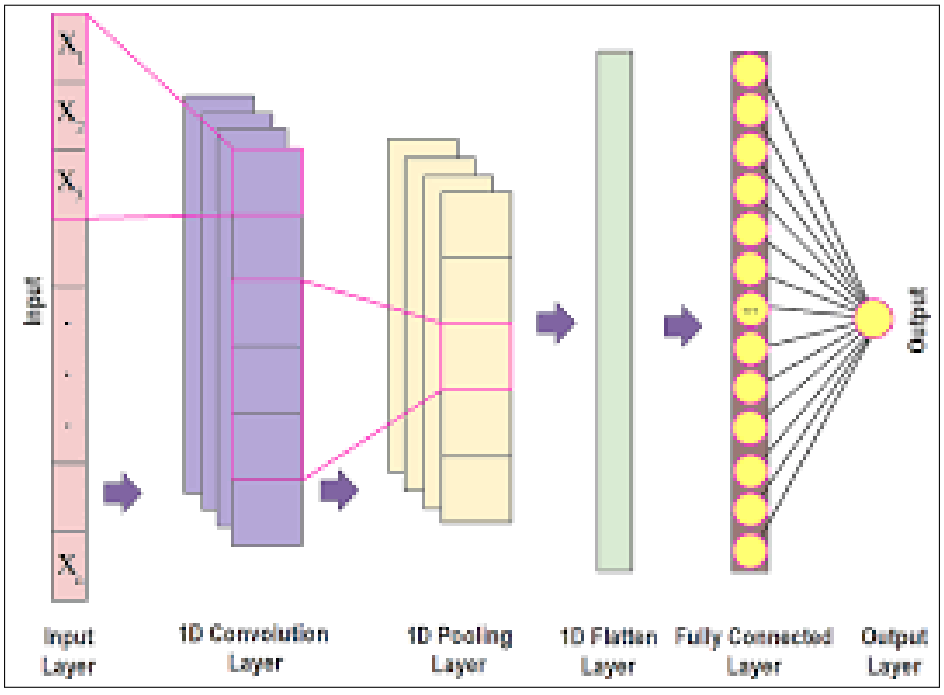


Figure 1.4 : One-Dimensional Convolutional Neural Network (1D-CNN) Architecture [33].

1.6.3.2 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a class of neural networks designed to process sequential data, such as text and time series. They are characterized by their ability to maintain temporal memory, where each step in processing is dependent on current and past information. This helps in understanding the relationship between sequential elements within the data. RNNs are used in many applications, such as natural language processing, text prediction, and time series analysis. However, it suffers from the vanishing gradient problem and the exploding gradient problem. Therefore, Long Short-Term Memory (LSTM) model was developed to address the vanishing gradient problem and improve information retention across long sequences [23]. The Figure 1.5 below represents the architecture of Recurrent Neural Networks.

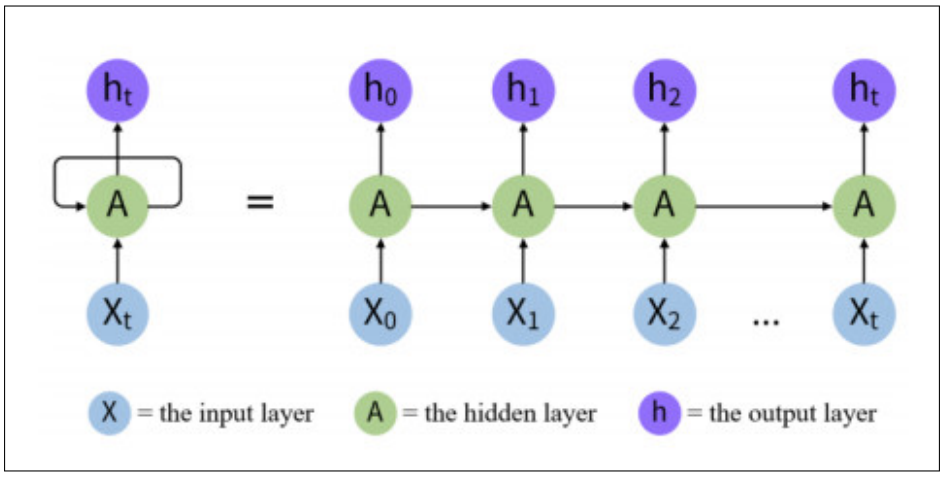


Figure 1.5 : Recurrent Neural Network (RNN) Architecture [34].

1.6.3.3 Long Short-Term Memory

Long Short-Term Memory (LSTM) is an improved type of recurrent neural network, introduced by Sepp Hochreiter and Jürgen Schmidhuber in 1997 [24]. It was designed to address the vanishing gradient problem in RNNs. It consists of three gates that control the flow of information : ***the input gate*** determines which new information should be added to memory ; ***the forget gate*** decides which information should be forgotten or retained ; and ***the output gate*** determines what information is passed from memory to the rest of the network [25]. Figure 1.6 illustrates the architecture of Long Short-Term Memory.

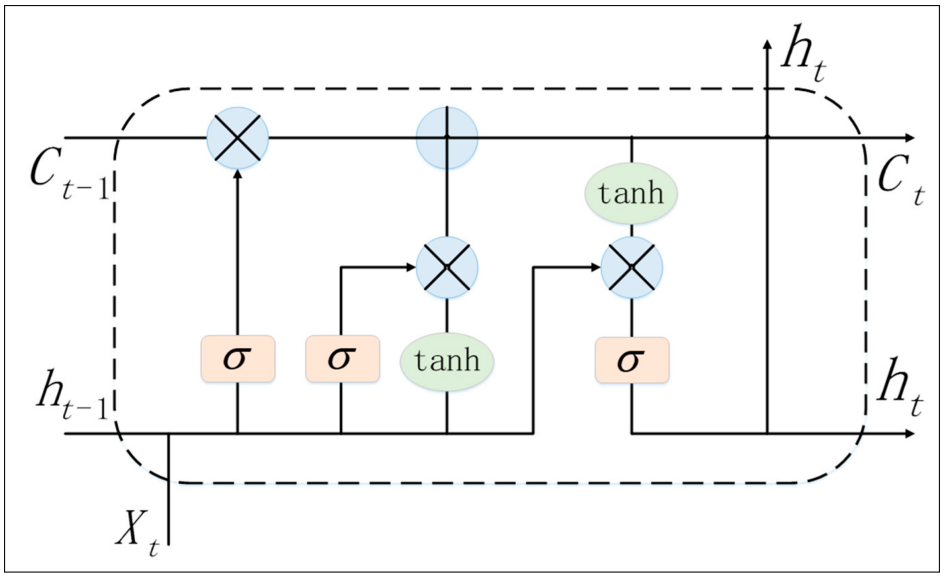


Figure 1.6 : Long Short-Term Memory (LSTM) Architecture [48].

Both RNNs and LSTMs process data in a single direction that depends only on past information. Therefore, the **Bidirectional Long Short-Term Memory (BiLSTM)** model was proposed. BiLSTM processes data in two directions to capture the full context of sequence elements (past and future). This is achieved by using two separate LSTM networks that share the same output layer. This model has shown better performance, especially in classification tasks [25], as shown in Figure 1.7.

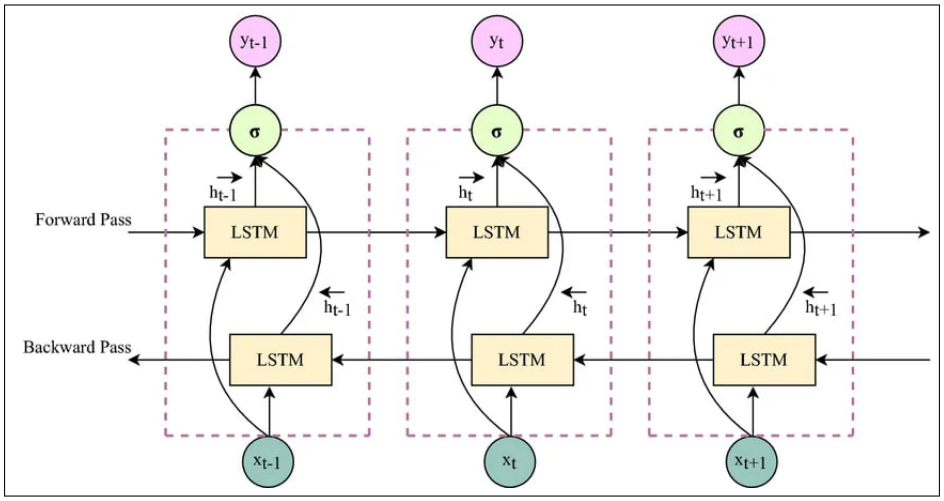


Figure 1.7 : Bidirectional Long Short-Term Memory (BiLSTM) Architecture [35].

1.6.3.4 Gated Recurrent Units

Gated Recurrent Units (GRUs) are a type of recurrent neural network, proposed by Cho et al [26]. in 2014, to address the vanishing gradient problem and improve memory retention in RNNs. It is considered a simplified version of LSTM because it does not include a separate memory cell. It relies on two basic gates : ***the update gate***, which controls how much of the previous information should be kept and combined with the current input ; and ***the reset gate***, which determines how much of the old information should be forgotten. Therefore, this model allows important information to be remembered for

a longer period while reducing computational complexity and accelerating training [27]. The Figure 1.8 shows the architecture of Gated Recurrent Units.

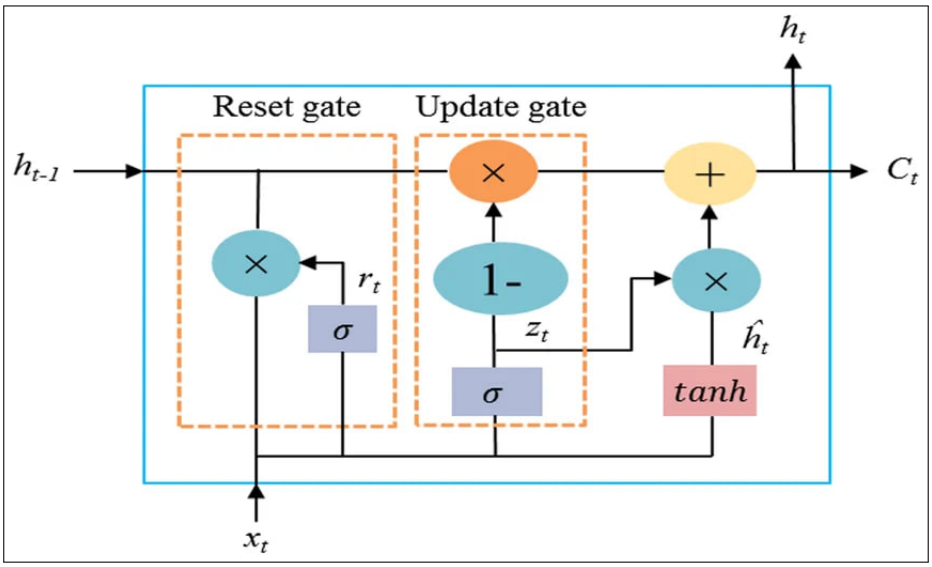


Figure 1.8 : Gated Recurrent Units (GRUs) Architecture [36].

The GRUs process data in only one direction, from past to future. Therefore, the **Bidirectional Gated Recurrent Unit (BiGRU)** model was introduced to process data in both forward and backward directions using two parallel layers of GRU. It processes from past to future and from future to past, and then the outputs from both directions are combined [27], as shown in Figure 1.9.

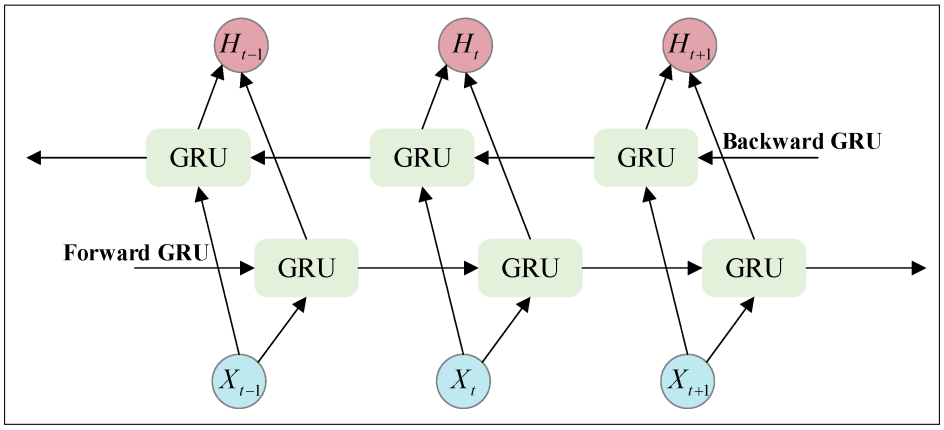


Figure 1.9 : Bidirectional Gated Recurrent Unit (BiGRU) Architecture [37].

1.6.3.5 Transformer

The Transformer architecture was introduced in 2017 by Vaswani et al. [55]. It is a fundamental deep learning architecture, especially used in natural language processing. It is based on the attention mechanism and Encoder-Decoder structure. The Encoder consists of several identical layers, where each layer consists of multi-head self-attention, a feed-forward neural network, as well as residual connections and normalization to improve stability. The decoder is similar to the encoder, except that it adds an attention layer over the encoder output and uses masking to ensure that each position depends only

on previous information. This makes the model capable of understanding relationships between elements efficiently [21]. The Figure 1.10 below represents the architecture of Transformer.

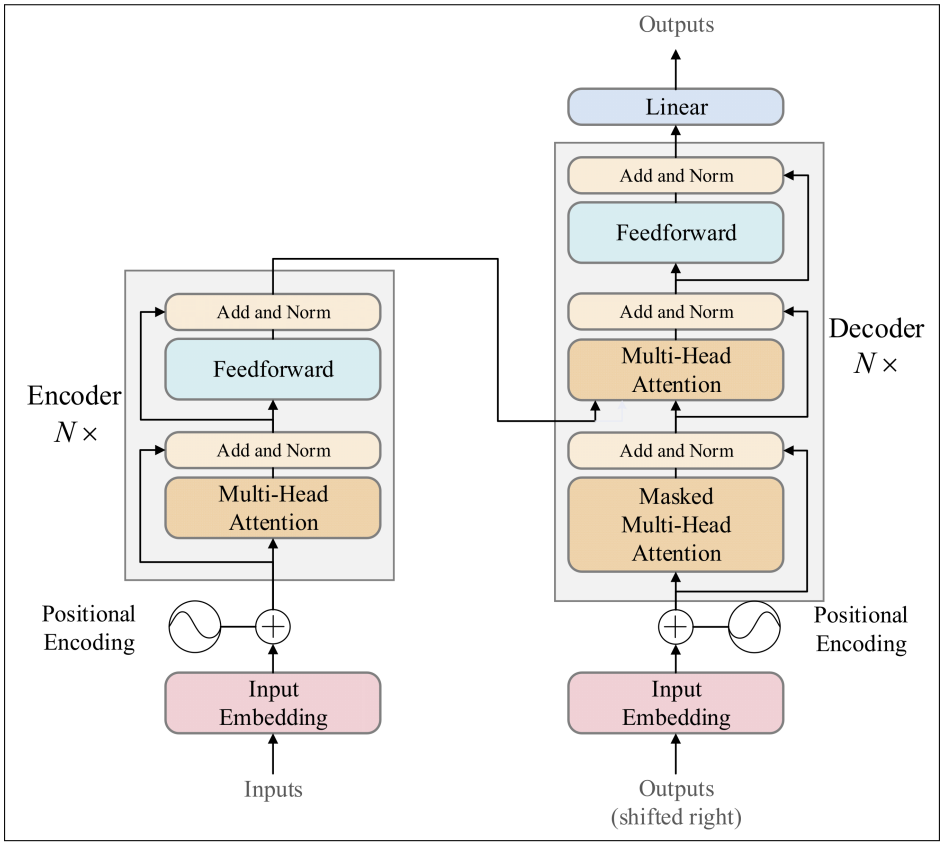


Figure 1.10 : Transformer Architecture [37].

1.6.3.6 Attention

A mechanism in neural networks used in natural language processing, which allows models to focus on important parts of data during processing. Its main idea is to make the model not process all elements in the sequence equally, but instead assign different weights to each element in the sequence depending on its importance [28]. This mechanism is widely used in natural language processing, especially in text analysis and conversational models, and is also one of the most important components that led to the development of modern architectures such as the Transformer architecture.

1.6.3.7 Bidirectional Encoder Representations from Transformers

Bidirectional Encoder Representations from Transformers (BERT) is a natural language processing model based on the Transformer architecture, introduced in 2019 by Devlin et al [49]. The model relies on bidirectional contextual learning by analyzing words using both the preceding and following contexts simultaneously, allowing for a deeper understanding of relationships and meaning within texts. BERT is used in many natural language processing tasks, such as text classification, question answering, and others. Figure 1.11 shows the architecture of Bidirectional Encoder Representations from Transformers (BERT).

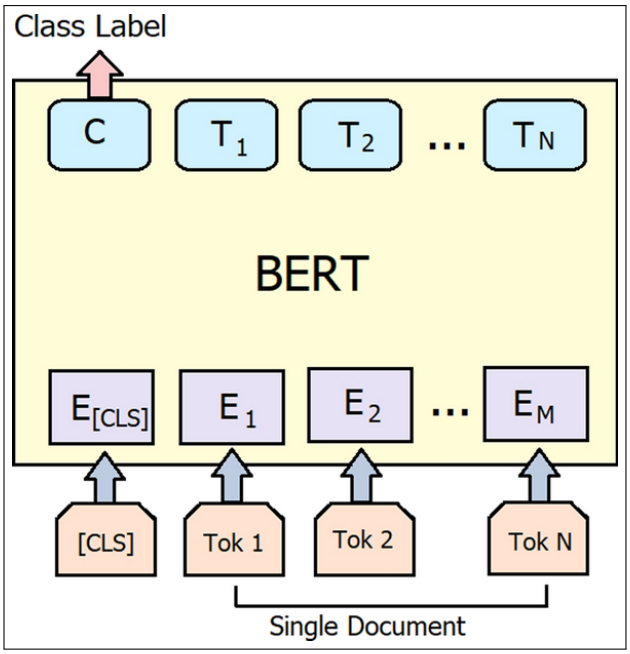


Figure 1.11 : Bidirectional Encoder Representations from Transformers (BERT) Architecture [50].

1.6.3.8 Hybrid Models

Hybrid models in deep learning integrate more than one type of neural network into a single model in order to leverage the strengths of each network in extracting and capturing features, thereby improving the model learning ability and increasing performance. Common examples of hybrid models include CNN-LSTM or CNN-GRU, which help capture both spatial and temporal relationships. Also, Transformer combined with other neural networks is widely used to enhance feature representation learning and context understanding. Hybrid models are used in many applications such as text and sequence processing, as well as images and videos analysis, where data is complex and requires multiple types of processing.

1.6.4 Representation Learning

In the field of deep learning, representation learning is a fundamental approach that aims to enable neural networks to automatically learn useful representations and features from raw data, instead of relying on manual representations. It makes models able to understand relationships and patterns within the data, and improve their performance in prediction or classification tasks.

1.6.5 Transfer Learning

Transfer Learning is a technique that uses the knowledge learned from a model trained on a specific domain or task to improve performance on another related task or domain, with the goal of addressing a data shortage problem. Model-based transfer learning is one of the most widely used approaches. A pre-trained model is used with the parameters (weights) it has learned, through freezing part of its layers (freezing) or fully retraining it on new data (fine-tuning) to adapt to the new task [27].

1.6.6 Regularization and Prevention Methods

As in machine learning algorithms, neural networks can also suffer from the problem of overfitting and poor generalization to new data. To avoid this problem, techniques called regularization methods are used. They reduce model complexity and prevent overfitting. The main methods are [19] :

- **Dropout :** It randomly deactivates some neurons during the training phase, which reduces model complexity.
- **Batch Normalization :** Contributes to the stability of the training process and reduces internal covariate shift, by normalizing the values within each batch, and reducing the problem of internal distribution shift.
- **Early Stopping :** It stops training when the model performance on validation data stops improving and starts to degrade, or when the validation loss starts increasing. It helps to select the best model in terms of performance and generalization ability.

1.7 Model Evaluation

It is the stage in which the quality of a classification model is evaluated. Numerical metrics are used to assess the model's performance by comparing predicted results with actual results, based on four main cases[30] :

- **True Positive (TP) :** When the model predicts the positive class correctly.
- **True Negative (TN) :** When the model predicts the negative class correctly.
- **False Positive (FP) :** When the model predicts the positive class, but it is actually negative.
- **False Negative (FN) :** When the model predicts a negative class but it is actually positive.

The most commonly used metrics for evaluation classification are[30] :

1.7.1 Accuracy :

Represents the proportion of correct predictions out of all predictions.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

1.7.2 Precision :

Measures the proportion of correctly predicted positive cases.

$$\text{Precision} = \frac{TP}{TP + FP}$$

1.7.3 Recall :

Measures the proportion of actual positive samples correctly identified.

$$\text{Recall} = \frac{TP}{TP + FN}$$

1.7.4 F1-score :

Represents the balance between Precision and Recall.

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

1.7.5 Confusion Matrix :

A table that represents TP, TN, FP, and FN values. Figure 1.12 illustrates the confusion matrix used to evaluate the performance of the models.

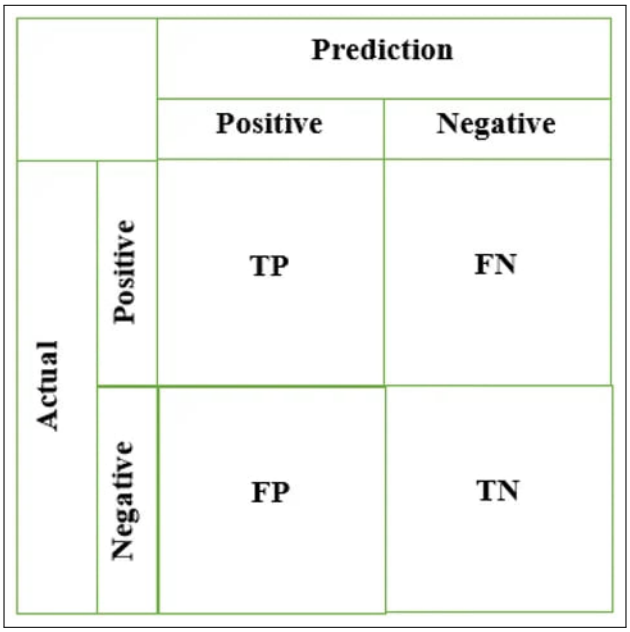


Figure 1.12 : Confusion Matrix [38].

1.8 Literature Review

Functional classification of protein sequences has received significant attention in scientific research over several years, as many studies have addressed the use of machine learning and deep learning techniques for protein sequence classification.

Dongardive and Abraham (2015) [39] proposed a protein classification approach using the KNN algorithm with a numerical representation of protein sequences based on a 3-gram technique. The study was applied to a dataset containing 717 protein sequences comprising 7 functional classes, extracted from the Swiss-Prot [56] database. The results of this study achieved an accuracy of 84% when using K=15.

In 2018, Vazhayil et al [40]. used the Swiss-Prot database, which contains 40,433 protein sequences belonging to 30 protein families, to perform protein classification. In this study, the sequences were represented using the 3-grams technique to extract patterns within the sequences, while setting the maximum sequence length to 1000 amino acids. The performance of deep neural network models was also compared with shallow neural networks and traditional machine learning models, where the LSTM model achieved the highest accuracy of 78.40%.

In the same year, Naveenkumar et al [41]. studied the role of protein sequence representation using the 3-grams technique, with a maximum sequence length of 1000 amino acids, in addition to the Keras Embedding technique on the Swiss-Prot database. The models were evaluated using deep learning approaches, and after applying cross-validation, the LSTM model achieved the highest accuracy of 91.24% when using the Keras Embedding technique.

Siddha (2020) [42] developed and compared machine learning and deep learning models for protein classification using the Protein Data Bank (PDB) database, based on sequences belonging to 20 functional classes. TF-IDF and N-gram were used for feature extraction in machine learning models, while Word Embedding using Keras was applied with deep learning models. Chi-square test was also used for feature selection, and Random Undersampling was applied to address the problem of class imbalance. The results showed that the Decision Tree model achieved the highest accuracy of 78.71%.

Similarly, in 2021, Sekhar et al [43]. applied Naïve Bayes, Random Forest, and LSTM algorithms on different subsets of protein sequences from the PDB database, which included 10 functional classes. The sequences were represented using Count Vectorizer technique. The highest prediction accuracy of 96% was achieved by the LSTM model using 10,000 sequences.

In 2022, Wang [44] studied protein classification using numerical experimental data extracted from the PDB database, where the data were normalized. The Extra Trees Classifier was used for feature importance analysis in machine learning models. The KNN model achieved the highest performance with an accuracy of 40%. In addition, amino acid sequences from the same database, which included 36,411 protein sequences distributed across 10 functional classes, were represented using k-mer and Word2Vec techniques. Among, the evaluated deep learning models, the CNN model achieved the best performance with an accuracy of 83%.

Tasnim et al.(2024) [45] used several Natural Language Processing techniques to efficiently classify protein sequences, where the sequences were represented using TF-IDF and Count Vectorizer with machine learning models. They also experimented with several other techniques (Embedding, Integer Encoding, One-Hot with Embedding, One-Hot with Integer Encoding, BLOSUM, and NLF) with deep learning models. This study was conducted using protein sequences from the PDB database, comprising 226,693 sequences distributed across 20 functional classes. The results showed that Count Vectorization achieved the highest accuracy with the SVM model, reaching 92%, while Integer Encoding with the CNN model achieved an accuracy of 90%.

In 2025, Perveen and Weeds [46] studied protein sequence classification using two different data splitting strategies : random splitting and ECOD family-based splitting, using the PDB database. Protein sequences and labels were represented using N-gram with CountVectorizer and LabelEncoder with machine learning models, while tokenization and integer encoding techniques were used with deep learning models. Several models, including transformer-based models using transfer learning, were evaluated with different sequence lengths. The Voting Classifier achieved an accuracy of 74% under random splitting, while the ProtBERT model, after fine-tuning, achieved the best results with an accuracy of 77%. However, performance decreased under ECOD-based splitting.

Finally, Chakraborty and Bhargava (2025) [47] used the PDB database by selecting protein sequences related to 10 functional classes. The sequences were represented using k-mer ($k = 5$) and integer encoding, with a maximum sequence length of 300 amino acids. Four different deep learning architectures were applied (CNN, BiLSTM, CNN-BiLSTM, CNN-Attention). The results showed that the CNN model achieved the highest accuracy of 91.8%.

The following Table 1.1 presents a summary of the previous studies conducted in the field of protein classification. Most studies relied on Accuracy as the evaluation metric for assessing model performance. The abbreviation ND in the table indicates that the corresponding information was not available in the study.

Table 1.1 : Summary of Related Works in Protein Sequence Classification.

Authors	**Dataset**	**Data Size/Nbr classes**	**Models**	**Feature Representation**	**Best-Performing Model**	**Accuracy (%)**
Dongardive and Abraham (2015) [39]	Swiss-Prot	717/ 7	KNN	3-gram	KNN with K=15	84.51
Vazhayil et al. (2018) [40]	Swiss-Prot	40,433/ 30	RNN, LSTM, GRU, DNN, ML Models	3-gram	LSTM	78.40
Naveenkumar et al. (2018) [41]	Swiss-Prot	40,433/ ND	DNN, RNN, LSTM, CNN	3-gram /Keras Embedding	LSTM	91.24
Siddha (2020) [42]	PDB	ND/ 20	DT, RF, CNN, LSTM	ML : TF-IDF +N-gram DL : Word Embedding	Decision Tree	78.71
Sekhar et al. (2021) [43]	PDB	10,000/ 10	NB, RF, LSTM	Count Vectorizer	LSTM	96.00
Wang (2022) [44]	PDB	36,411/ 10	Multinomial Regression, KNN, BiLSTM, CNN	DL : K-mer + Word2Vec	CNN	83.00
Tasnim et al. (2024) [45]	PDB	226,693/ 20	NB, LR, SVM, NN, CNN, Prot	ML : TF-IDF + Count Vectorizer DL : Embedding, Integer Encoding, One-Hot with Embedding, One-Hot with Integer Encoding, BLOSUM, NLF	SVM	92.00
Perveen and Weeds (2025) [46]	PDB	ND/ ND	ML, DL and Transformer Models	ML : N-gram + Count Vectorizer DL : Tokenization + Integer Encoding Pre-trained Transformer Rebresentation	ProtBert (Fine-Tuning)	77.00
Chakraborty and Bhargava (2025) [47]	PDB	ND/ 10	CNN, BiLSTM, CNN-BiLSTM, CNN-Attention	5-mer + Integer Encoding	CNN	91.80

Overall, studies have shown that deep learning models achieve relatively better performance than machine learning models in protein classification. In addition, hybrid models have demonstrated competitive performance compared to individual models. However, performance strongly depends on the type and distribution of data, as well as on protein sequence representation methods, which mainly focus on extracting statistical features that do not suffciently capture the complex biological relationships within the sequence. In contrast, transformer models have shown significant progress in sequence representation due to their ability to capture deep biological context. However, their performance remains unstable across different studies.

Existing studies suffer from several limitations. First, most studies rely on a single representation strategy and do not investigate the impact of multiple sequence encoding techniques. Second, many works evaluate either CNN or recurrent networks independently, without combining local motif extraction and contextual sequence modeling. Third, attention-based architectures remain insufficiently explored despite their potential biological relevance. Fourth, very few studies perform a comprehensive comparison across

machine learning, deep learning, hybrid learning, ensemble learning, and transfer learning paradigms within a unified experimental framework. Finally, existing studies rarely investigate how different architectures interact with different biological sequence representations.

1.9 Conclusion

This chapter provided a comprehensive overview of proteins and their representation methods, which is considered a key step in building effective models, in addition to presenting the main concepts related to machine learning and deep learning, including the techniques and algorithms applied to protein classification. It also reviewed and analyzed previous studies in this field, paving the way for the next chapter, which will present the methodology, adopted approaches and proposed models aimed at improving the performance of protein sequence classification.

Chapter 2
Research Methodology

Chapter 2

Research Methodology

2.1 Introduction

This chapter presents the methodology adopted in this study for the functional classification of proteins based on amino acid sequences, starting from data collection and preprocessing, through sequence representation and feature extraction, and finally model development and performance evaluation. The proposed methodological framework is based on the study and integration of several approaches, including machine learning and deep learning models, hybrid and ensemble learning methods, as well as protein language models, along with the use of several methods for sequence representation. Furthermore, this study addresses training strategies designed to improve model performance, as well as the presentation of the evaluation methods used to analyze model performance.

2.2 Research Overview

Protein sequence classification is considered one of the major challenges in the field of bioinformatics due to the complexity of biological data and the difficulty of analyzing it. This study addressed this problem and aims to improve protein classification performance by applying a wide range of modeling approaches, including machine learning models, deep learning models, hybrid models, and hybrid ensemble models, in addition to transformer-based transfer learning models. This study also provides various protein sequence representation techniques, particularly those based on protein language models, to enhance representation quality and improve classification accuracy.

The main contributions of this work can be summarized as follows :

- A comprehensive comparative analysis of multiple learning methods, including Machine Learning, Deep Learning, Hybrid Learning, Ensemble Learning, and Transfer Learning, within a unified experimental framework.
- An extensive investigation of protein sequence representation techniques, ranging from classical approaches (TF-IDF, N-grams, One-Hot Encoding, K-mer Encoding, Raw Sequences) to advanced pretrained protein language model embeddings (ProtBERT and ESM).
- The design and exploration of novel deep learning architectures based on CNN, LSTM, BiLSTM, and Transformer models, aimed at effectively capturing local patterns, sequential dependencies, and global contextual information in protein sequences.
- The design and evaluation of novel hybrid deep learning architectures combining CNN, BiLSTM, BiGRU, and Attention mechanisms to capture local patterns, long-range dependencies, and functionally relevant sequence regions.
- The exploration of ensemble learning strategies by combining deep feature extractors with classical machine learning classifiers such as Random Forest and XGBoost, improving robustness and generalization.
- The validation of transfer learning effectiveness through fine-tuning of pretrained models.

- The examination of the generalization ability of the proposed models using cross-validation and an independent ex ternal dataset.
- A comprehensive error analysis along with an investigation of the impact of dataset composition on classification performance.
- A complete and systematic experimental pipeline analyzing the relationship between model complexity, computational cost, and classification performance.

Figure 2.1 illustrates the general workflow of the proposed methodology for protein sequence classification, beginning with data collection and ending with the prediction of protein functional classes. The pipeline is organized into five major stages : data acquisition, preprocessing, feature extraction and sequence representation, modeling approaches, and classification.

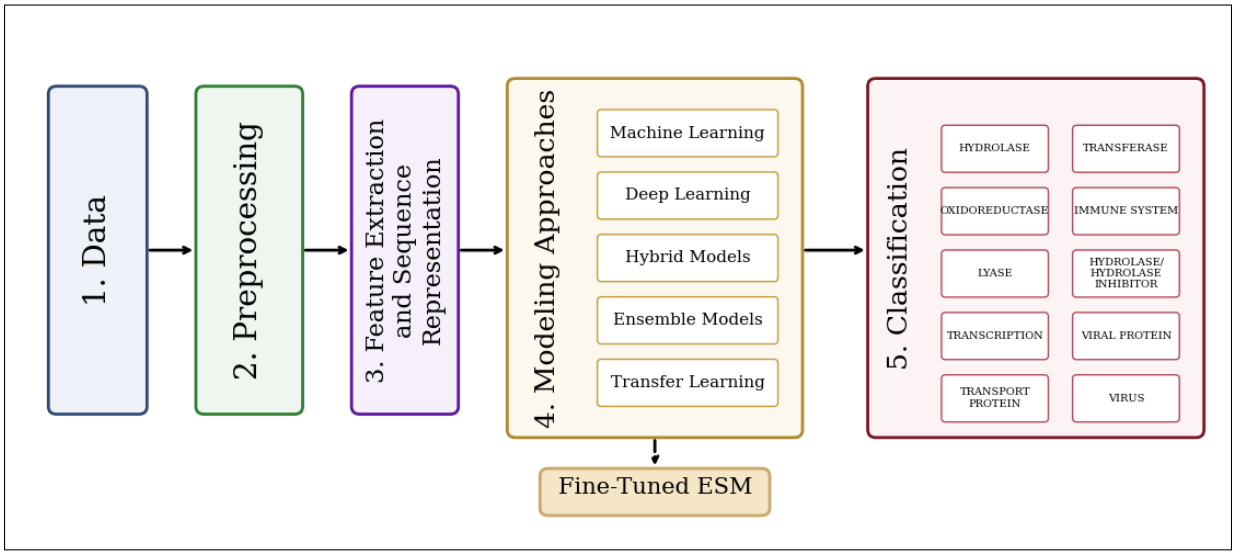


Figure 2.1 : Protein Sequence Classification Pipeline.

1. The first stage, Data, involves collecting protein sequences from publicly available databases. The quality and diversity of the data collected are crucial because they directly affect the performance and generalization ability of the classification models.
2. In the second stage, Preprocessing, the raw protein sequences are cleaned and prepared for analysis. Proper preprocessing ensures that the learning algorithms are trained on reliable and representative data.
3. The third stage, Feature Extraction and Sequence Representation, transforms protein sequences into numerical representations that machine learning algorithms can process.
4. The fourth stage, Modeling Approaches, presents various computational strategies used for classification. Machine Learning algorithms rely on extracted features to perform classification. Deep Learning models automatically learn discriminative features directly from sequence representations. In addition, Hybrid Models combine multiple techniques to leverage their complementary strengths, while Ensemble Models aggregate predictions from multiple classifiers to improve robustness and accuracy. Transfer Learning has also emerged as an effective approach by exploiting knowledge learned from large-scale protein datasets and adapting it to specific classification tasks.
5. Finally, the Classification stage produces the predicted protein categories. As illustrated in the figure, proteins can be assigned to different functional or biological classes, such as hydrolases, transferases, oxidoreductases, lyases, or others.

Overall, this pipeline provides a systematic framework for protein sequence classification. The following sections provide a comprehensive overview of the adopted methodology for functional protein classification, covering all stages from data collection and preprocessing, through feature extraction and sequence representation, to model development and performance evaluation.

2.3 Processing Flow

This section presents the stages adopted for collecting and processing the data used in this study.

2.3.1 Data Acquisition and Preparation

In this study, we use a dataset of protein sequences derived from the Protein Data Bank (PDB)[57]. PDB is a publicly accessible database that stores structural data of proteins. It was established in 1971 by Dayhoff et al. Initially, it contained only protein metadata. Later, it evolved to include more complex three-dimensional (3D) structural data. In 1998, the PDB was managed by the Research Collaboratory for Structural Bioinformatics (RCSB), which developed it into a fundamental scientific resource in the study of the structures and biological functions of macromolecules[51].

2.3.1.1 Dataset Description

The dataset used in this study is available on the Kaggle platform [52], and contains more than 400,000 entries, including DNA, RNA, and protein molecules. The data is organized into two main files : the first file contains molecular metadata, including biological function information, while the second file contains molecular sequence data. The two datasets are merged into a single collection using the common column StructureID.

2.3.1.2 Data Preprocessing

Following dataset integration, only protein-related sequences are retained using the macromoleculeType attribute, which represents the molecule type. The data are then cleaned to ensure quality by removing missing values and invalid entries. We keep only the essential columns, including : **structure ID**, representing the structure identifier ; **chain ID**, representing the sequence identifier ; **sequence**, representing the amino acid sequence composed of consecutive characters, where each character corresponds to a specific amino acid ; and **classification**, representing the functional class.
The sequences belonging to the top 10 functional classes are then selected. In addition, sequences containing ambiguous amino acid symbols such as "X" are retained due to their low frequency and negligible impact, as illustrated in Figure 2.2. As a result, a dataset consisting of **188,353** protein sequences is obtained and used in the subsequent stages of this study, including feature extraction, model training, and performance evaluation.

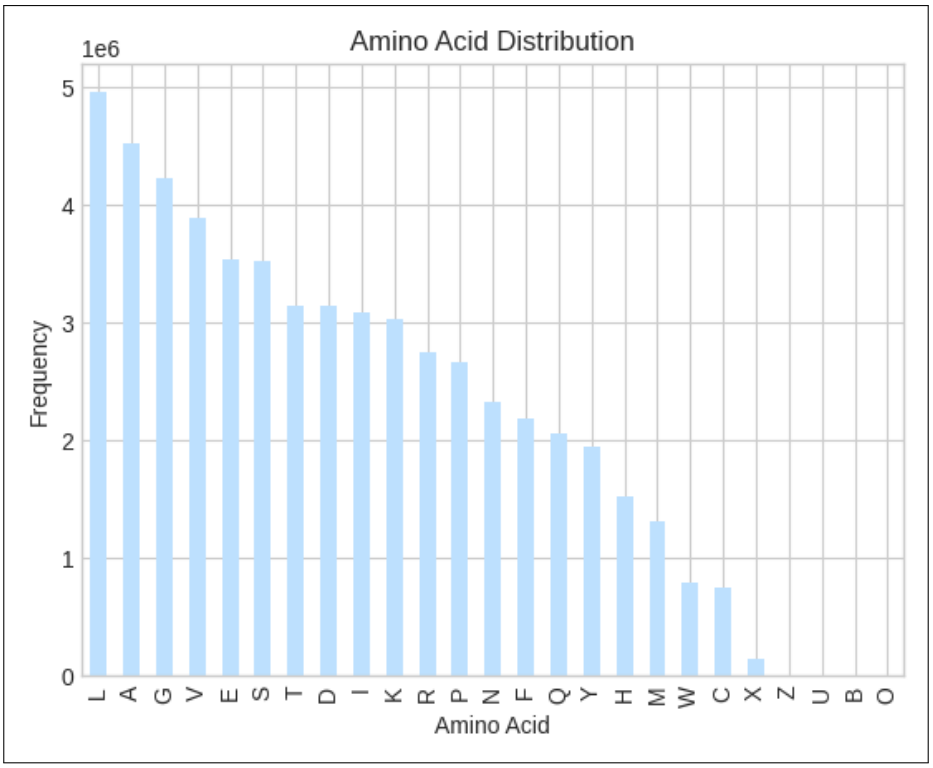


Figure 2.2 : Amino Acid Frequency Distribution.

2.3.1.3 Class Distribution and Imbalance Analysis

Understanding the distribution of classes within biological datasets is an essential step in classification tasks. As shown in Figure 2.3, the dataset suffers from class imbalance, where some classes are represented by a significantly larger number of samples compared to others. This variation may affect the learning process of models, as they may become biased toward the majority classes at the expense of the minority ones. This ultimately could reduce the ability of the models to effectively recognize all classes and achieve balanced performance.

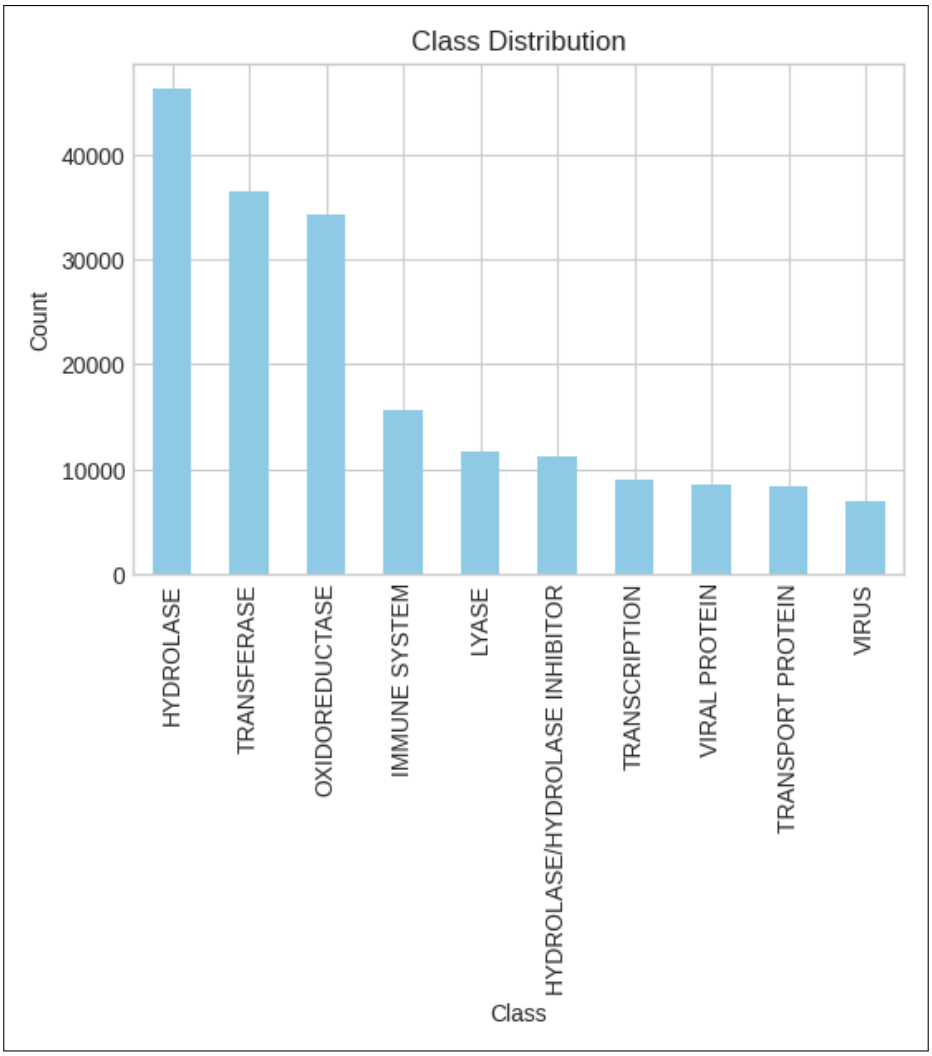


Figure 2.3 : Class Distribution in the Dataset

.

2.3.2 Feature Engineering and Sequence Representation

Feature engineering is considered one of the fundamental stages in building effective models for protein sequence classification, as it involves converting the characters forming the sequences into numerical representations that can be processed by the models. Several techniques are adopted for feature extraction and sequence representation according to the nature of each model. Figure 2.4 summarizes the representation methods used in this study along with the corresponding models for each method.

2.3.2.1 Feature Extraction for Machine Learning

N-gram and TF-IDF techniques are used to convert protein sequences into numerical representations suitable for machine learning models. The N-gram technique divides sequences into overlapping subsequences in order to extract local relationships between amino acids. In addition, the TF-IDF technique is adopted to highlight the most influential patterns in amino acid sequences by assigning higher weights to the most important features (n-grams). The feature dimensionality depends on the used model.

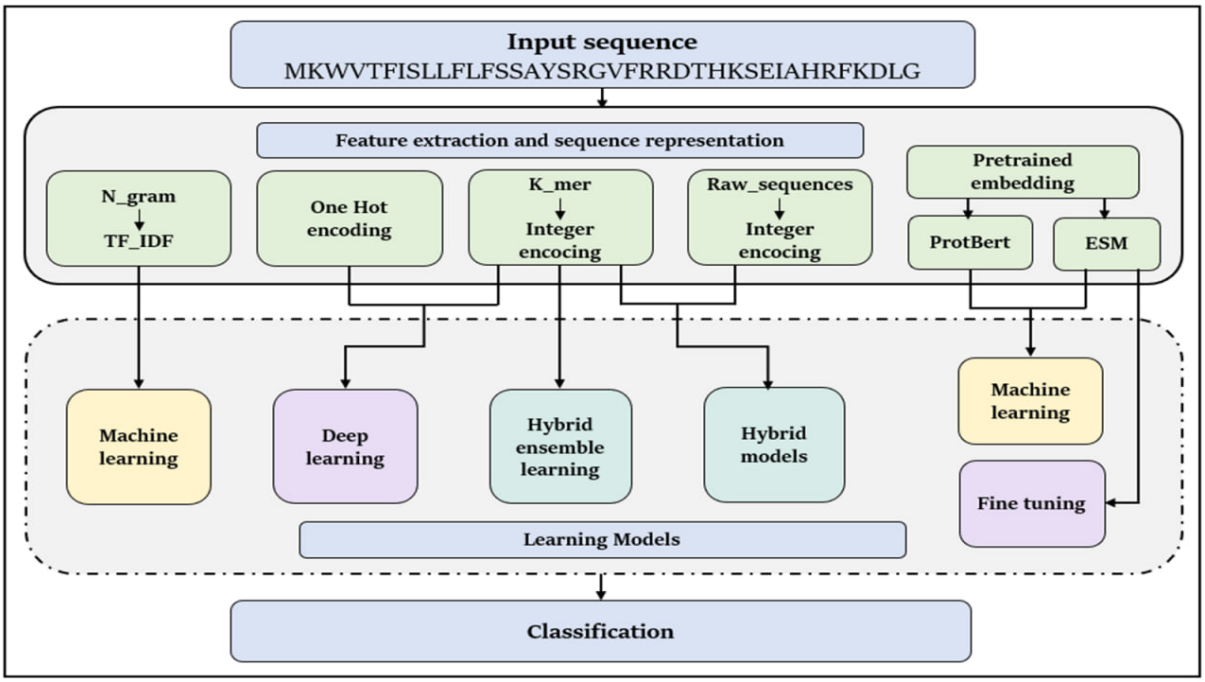


Figure 2.4 : Feature Engineering and Sequence Representation Methods

2.3.2.2 Sequences Representation

Several methods are used for sequence representation, including :

One-Hot Encoding : One-Hot Encoding is used to numerically represent protein sequences in deep learning models (CNN) by converting each amino acid into a binary vector with a length equal to the vocabulary size. A vocab_size of 21 is adopted, representing the 20 standard amino acids in addition to an extra token for non-standard amino acids. Each amino acid is represented by a value of 1 in its corresponding position, while all other positions are filled with 0.

Raw-sequences encoding : Raw protein sequences are used with hybrid models to enable them to automatically learn features and patterns directly from the original data. These sequences are numerically represented using Integer Encoding at the amino acid level using a vocabulary of 21 tokens, including the standard amino acids in addition to a special symbol representing unknown amino acids.

k-mer-based encoding : K-mer combined with Integer Encoding is applied to obtain more expressive representations that capture complex patterns within sequences. These representations are processed using deep learning models, hybrid models, and ensemble learning models.
Protein sequences are first segmented into overlapping subsequences of length 5 (K=5), where the value of K was selected based on empirical experiments that provided the best balance between representation quality and computational cost.
Integer Encoding converts the extracted K-mers into unique numerical values, based on a vocabulary constructed from all K-mers extracted from the training data, resulting in a vocabulary size of 1,904,347.

Sequence Length and Padding Strategy : Protein sequences in the dataset show significant variation in sequence lengths, as illustrated in Figure 2.5, which necessitates standardizing these lengths before feeding them into the models. After statistical analysis of the dataset, it is found that the median sequence length is 260 amino acids, and that 75% of the sequences are shorter than 368 amino acids. Based on this, a fixed maximum length of 300 is selected, as it provides a balance between covering most sequences, preserving biological information, and reducing computational complexity.
A Padding procedure is applied by adding zero values to the end of sequences shorter than the defined length, while longer sequences are truncated. This ensures the standardization of input dimensions across all samples.

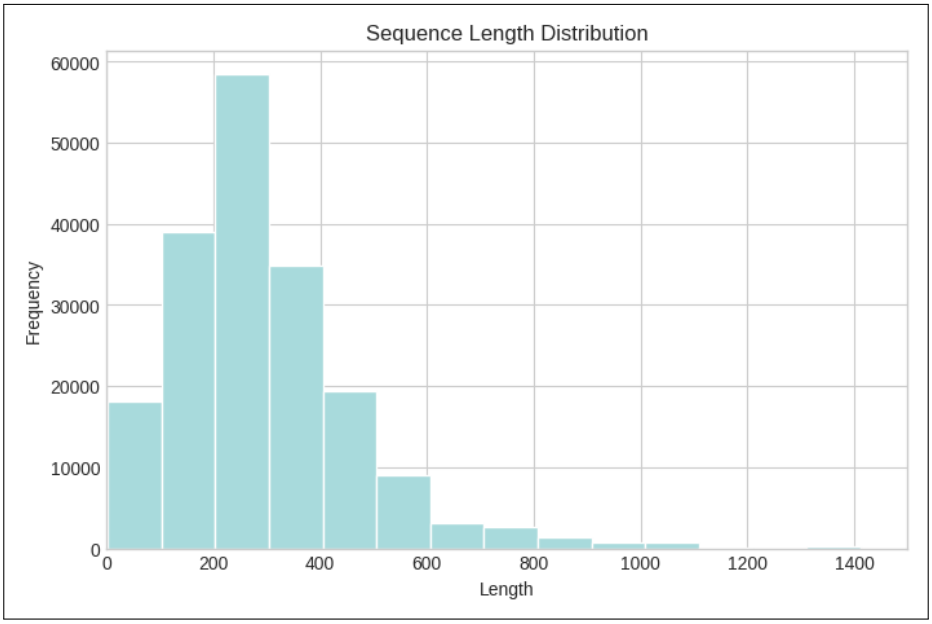


Figure 2.5 : Sequence Length Distribution in the Dataset.

Label Encoding : To prepare the dataset for classification tasks, the functional classes are encoded into a numerical format using Label Encoding, where a unique numerical value is assigned to each protein class, in order to enable the models to properly process the classification outputs.

2.3.3 Pretrained Embeddings

2.3.3.1 ProtBERT

ProtBERT is a protein language model and an improved version of (BERT) based on the Transformer architecture. It was developed in 2021 by Elnaggar et al [58]. through adapting the BERT architecture and retraining it on more than 200 million protein sequences, in order to enhance its ability to represent biological properties related to protein structure and function.[53].

2.3.3.2 Evolutionary Scale Modeling

Evolutionary Scale Modeling (ESM) is a series of protein language models trained on hundreds of millions of proteins using large language model techniques and based on

the Transformer architecture. These models are characterized by their ability to capture patterns and features, in addition to generate contextual representations that reflect biological properties and their functional importance within protein sequences[54]. The corresponding Figure 2.6 represents the architecture of the ESM model.

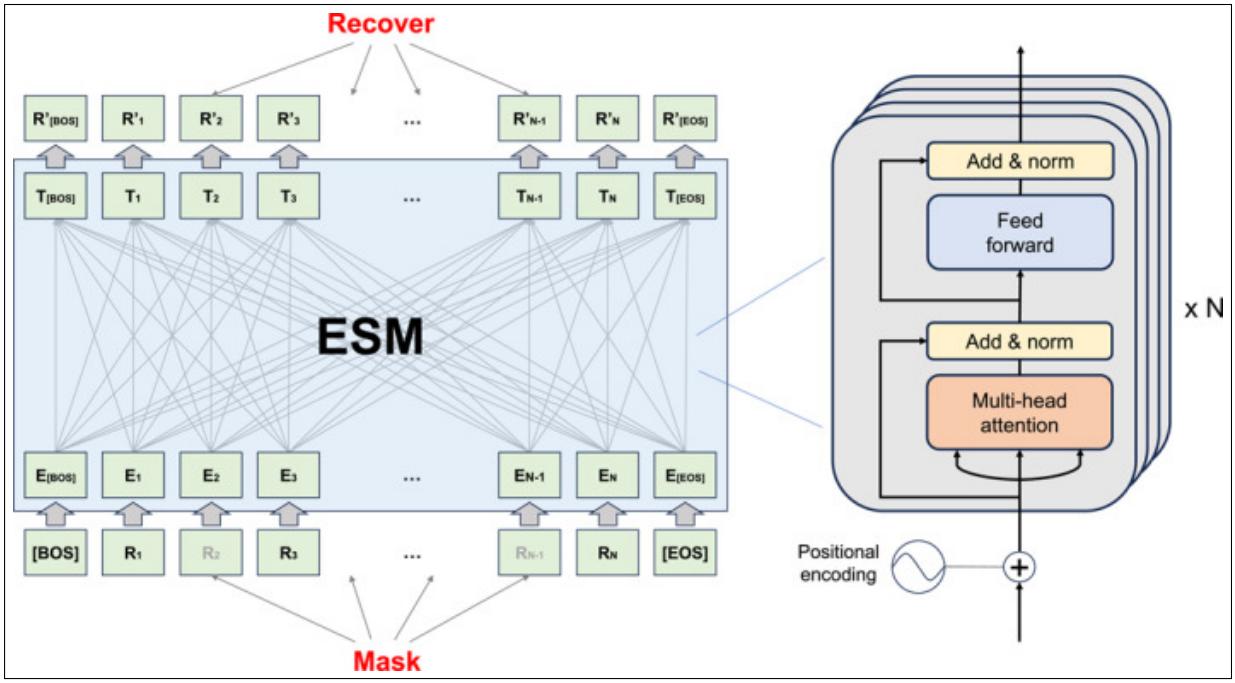


Figure 2.6 : Evolutionary Scale Modeling (ESM) Architecture[54].

Pretrained embeddings are adopted, where protein sequences are represented in a rich and compact manner by passing them through pretrained language models, including ProtBERT and ESM. These models segment sequences into tokens and convert each token into a numerical representation based on its sequential context. Finally, these representations are aggregated to produce a single vector representing the entire sequence, enabling the capture of long-range and complex relationships between amino acids within the protein sequence.

2.4 Model Design and Development

This study employs a set of approaches, including machine learning and deep learning models, hybrid and ensemble learning models, as well as protein language models.

2.4.1 Machine Learning Models

Several machine learning models are employed in this study to evaluate their performance in protein sequence classification tasks. These models include **Support Vector Machine (SVM)**, **Logistic Regression**, **Naive Bayes**, **K-Nearest Neighbors (KNN)**, **Decision Tree**, **Random Forest**, and **XGBoost**. The selection of these models aims at covering a diverse set of learning paradigms, including linear models, probabilistic approaches, instance-based learning, tree-based methods, and ensemble learning techniques, in order to ensure a comprehensive evaluation of classification performance. The following

Table 2.1 presents the final values of the N-gram range, maximum features, and the hyperparameters adopted for each machine learning model after the hyperparameter tuning process.

Table 2.1 : Hyperparameter Settings of Machine Learning Models.

Model	**N-gram**	**max-features**	**Hyperparameters**
SVM	n=4	30000	kernel = linear, C=10, max_iter=10000
Logistic Regression	n= rang (2–5)	50000	C=10, max_iter=5000, n_jobs=-1
Naive Bayes	n=5	30000	alpha=0.1
KNN	n=3	10000	k=5, metric=cosine, n_jobs=-1
Decision Tree	n=2	30000	max_depth=30, min_samples_split=5
Random Forest	n=2	30000	n_estimators=200, max_depth=30, min_samples_split=5, criterion=gini, n_jobs=-1
XGBoost	n=3	20000	n_estimators=200, max_depth=6, learning_rate=0.1, objective=multi :softmax, num_class=len(set(y_train)), eval_metric=mlogloss

2.4.2 Deep Learning Models

A set of deep learning models is adopted due to their ability to learn and capture complex patterns within protein sequences, in addition to automatically extract features from protein data. Protein sequences are numerically represented and encoded using the One-Hot Encoding and K-mer with Integer Encoding techniques to provide suitable inputs for the models. The architectures are selected based on their ability to learn representations and improve performance in protein classification tasks.

2.4.2.1 Deep Learning Architectures

Several deep learning architectures are adopted, including :

CNN : The proposed CNN architecture consists of several sequential layers designed to extract discriminative features from protein sequence representations.
The input layer receives the numerical representations of the sequences, which are first processed by an Embedding layer that converts them into dense trainable vectors. After that, two 1D-CNN layers are used to extract important local patterns. Batch Normalization is applied after each convolutional layer to improve training stability. A Max Pooling layer follows the first convolutional layer to reduce dimensionality, while a Global Max Pooling layer is applied after the second convolutional layer to aggregate the most significant extracted features. Finally, the data passes through a Dense layer followed by a Dropout layer to reduce overfitting, and a Softmax layer is used for final classification. The proposed CNN architecture is illustrated in Figure 2.7.

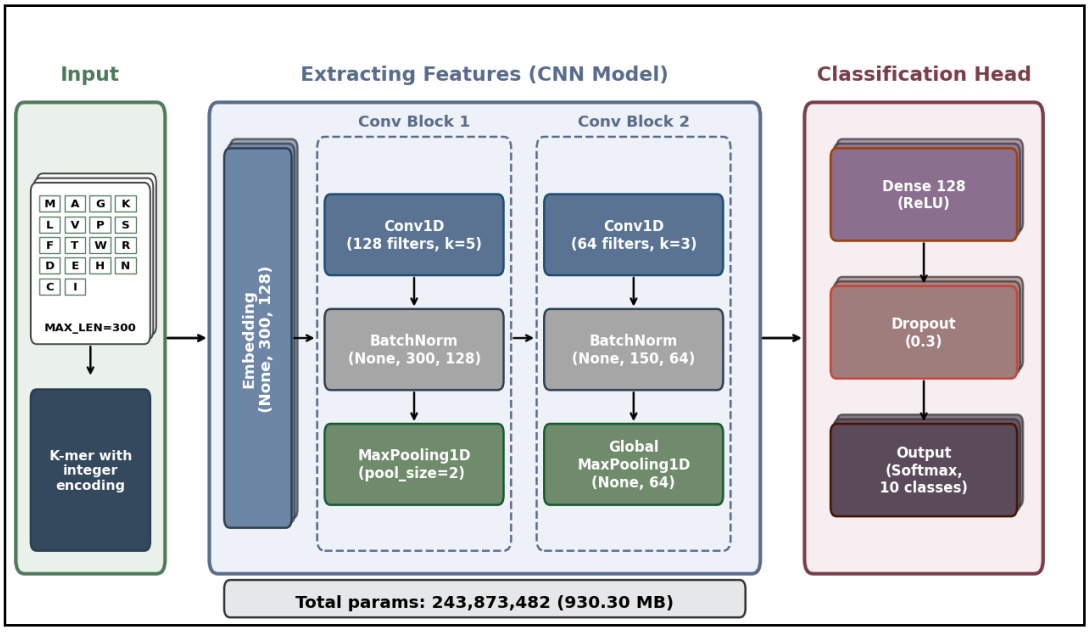


Figure 2.7 : Proposed CNN Architecture.

LSTM : The proposed LSTM architecture aims to learn sequential dependencies within protein sequences and extract temporally relevant features.
The input layer receives the numerical sequence representations, which are then transformed into dense vectors using an Embedding layer. Two LSTM layers are subsequently applied. The first uses return_sequences=True to extract full sequential representations, while the second summarizes the information at the sequence level. Batch Normalization is used to improve training stability, along with Dropout to reduce overfitting. Finally, the data is passed through a Dense layer and a Softmax output layer for classification. The proposed LSTM architecture is illustrated in Figure 2.8.

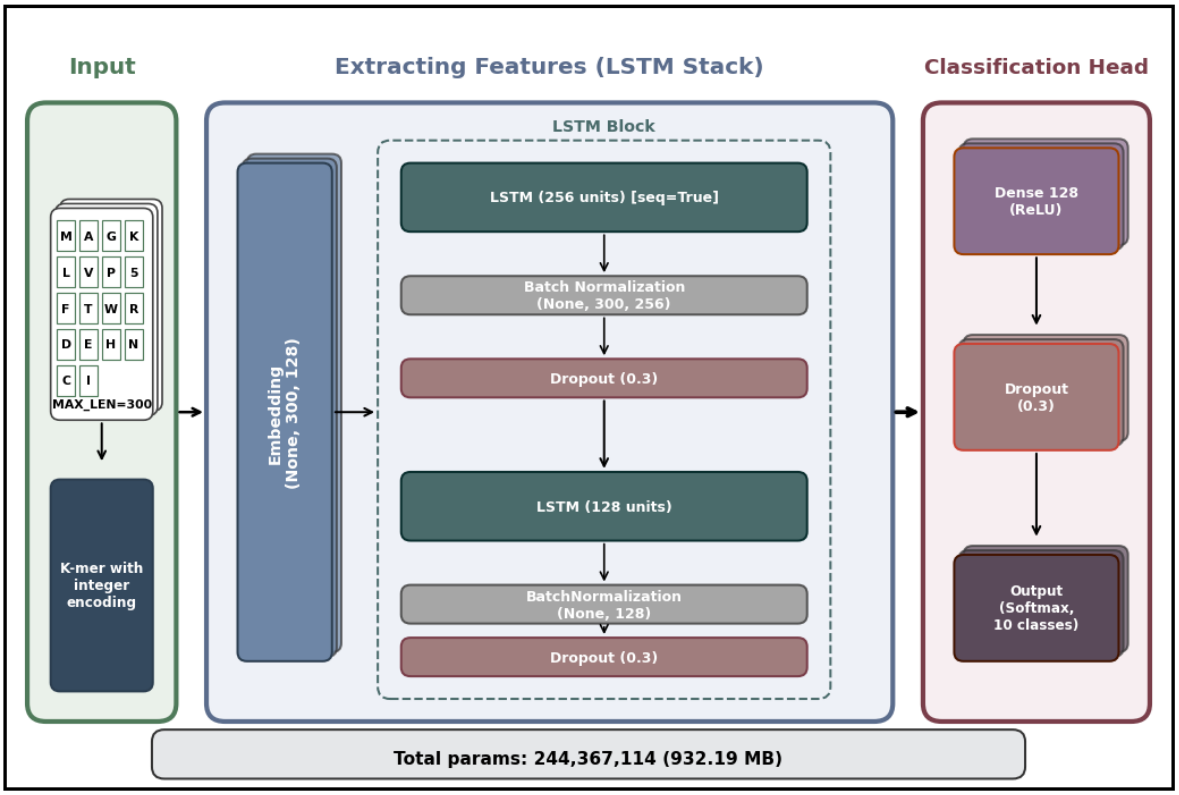


Figure 2.8 : Proposed LSTM Architecture.

BiLSTM : The BiLSTM architecture learns sequential dependencies within protein sequences in both directions, which enhances the contextual representation of amino acids. The input layer receives the numerical sequence representations, which are then transformed into dense trainable vectors using an Embedding layer, with Spatial Dropout applied to reduce feature dependence. Two BiLSTM layers are then used : the first extracts full bidirectional sequential representations using return_sequences=True, while the second summarizes the information at the sequence level. Batch Normalization is applied after the first BiLSTM layer to improve training stability, along with Dropout to reduce overfitting. Finally, the data is passed through a Dense layer followed by a Softmax output layer for classification. Figure 2.9 illustrates the proposed architecture of the BiLSTM model.

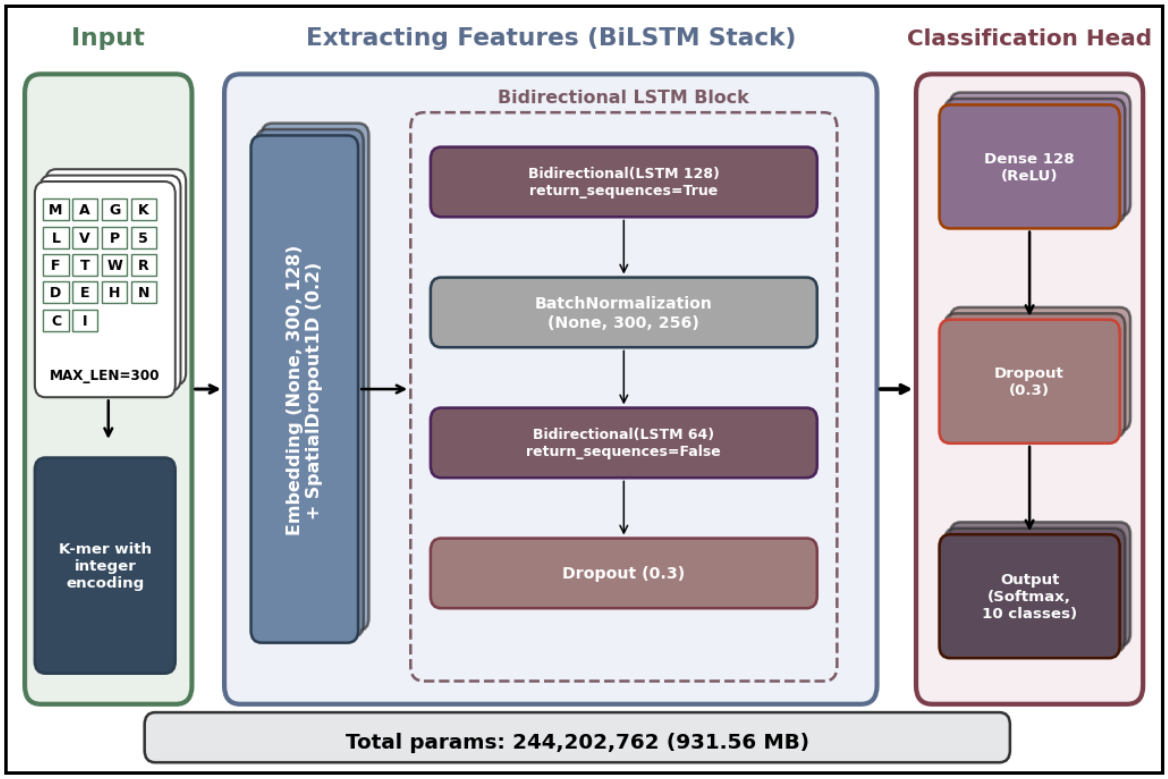


Figure 2.9 : Proposed BiLSTM Architecture.

Enhanced Transformer Encoder : The Enhanced Transformer Encoder (ETE) architecture is adopted to learn complex and long-range relationships between amino acids. The input layer receives numerical sequence representations, followed by an Embedding layer with L2 regularization to improve generalization and reduce overfitting. Then, Layer Normalization and Dropout are applied to enhance representation stability, along with learnable positional encoding to enable the model to capture the order of elements within the sequence.

The representations are then passed through four stacked Transformer Encoder layers, where each layer includes Layer Normalization and Multi-Head Self-Attention to capture dependencies between all amino acids, along with Dropout. Residual connections are used to link each layer's input with its intermediate output. This is followed by another Layer Normalization and a Feed Forward Network consisting of two Dense layers with Dropout, along with a residual connection to improve information flow and training stability. Finally, Global Average Pooling is applied to obtain a global sequence representation, followed by Layer Normalization. The representation is then passed through Dense layers with Dropout, and the final classification is performed using a Softmax layer. The proposed ETE architecture is illustrated in Figure 2.10.

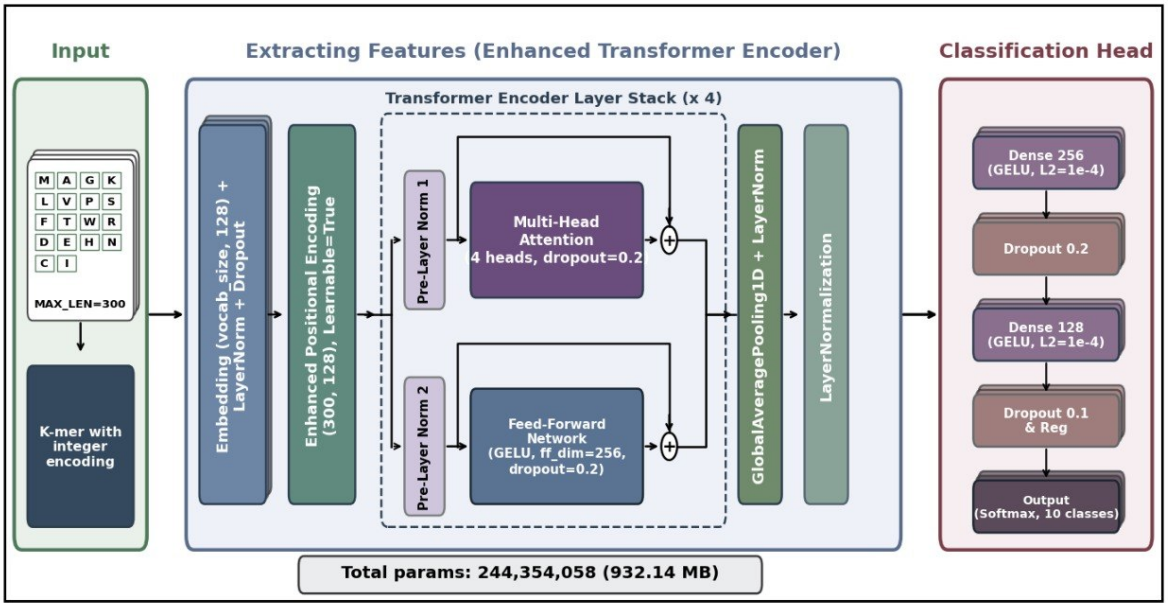


Figure 2.10 : Proposed Enhanced Transformer Encoder Architecture.

2.4.2.2 Hyperparameter Tuning and Configuration

The models are trained using the same processing pipeline, while different hyperparameter values are tested and illustrated in Table 2.2. The final values are selected based on the performance achieved on the validation set.

Tab. 2.2 : Hyperparameter Settings of Deep Learning Models.

Model	Learning Rate	Batch Size	Number of Epochs	Optimizer	Dropout	Spatial Dropout1D
CNN	0.001	64	30	Adam	0.3	/
LSTM	0.0001	64	50	Adam	0.3	/
BiLSTM	0.0005	64	30	Adam	0.3	0.2
ETE	0.001	32	30	AdamW	0.2/0.1	/

2.4.3 Hybrid Models

Hybrid models are adopted to improve protein classification performance by combining multiple deep learning architectures in order to benefit from the strengths of each model in processing and analyzing protein sequences. The sequences are represented using the Integer Encoding, in addition to K-mer based Integer Encoding.

2.4.3.1 Hybrid Models Based on Raw Sequences Representation

Several hybrid architectures based on the Integer Encoding are employed, with the hyperparameters of each model optimized.

Hybrid Models Architectures

CNN-BiLSTM : The proposed CNN-BiLSTM model benefits from the ability of both architectures to extract local patterns and learn sequential dependencies within protein sequences.
The model starts with an input layer, followed by an Embedding layer that transforms the representations into dense vectors, with Spatial Dropout applied to reduce overfitting. After that, two 1D-CNN layers are used to extract important local patterns, with Batch Normalization applied to improve training stability and Max Pooling to reduce dimensionality. The extracted features are passed through two BiLSTM layers to learn sequential dependencies, where Batch Normalization and Dropout are used to improve generalization. Finally, the data is passed through Dense layers with Dropout,where the dropout parameter is set to 0.5, since empirical studies have demonstrated that this value is near-optimal for many different neural network architectures [66]. The final classification is performed using a Softmax layer. The adopted CNN–BiLSTM architecture is illustrated in Figure 2.11 .

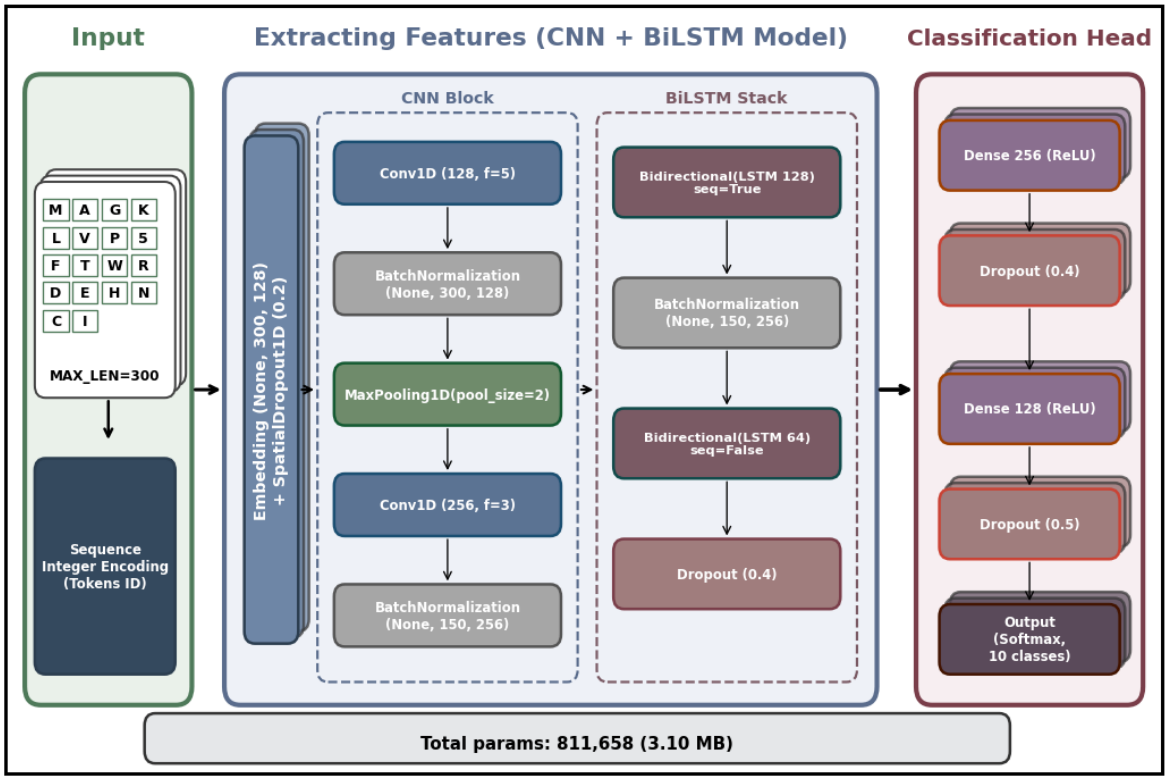


Figure 2.11 : Proposed CNN-BiLSTM Architecture.

CNN-BiGRU : The CNN-BiGRU model also benefits from the ability of both architectures to extract local patterns and learn sequential dependencies within protein sequences.
The input sequences are first embedded into dense vector representation using an Embedding layer, followed by Spatial Dropout to improve generalization. Two 1D-CNN layers are applied to extract meaningful local motifs, with Batch Normalization applied to improve training stability and Max Pooling to reduce feature dimensionality. The resulting features are processed by two BiGRU layers to learn sequential dependencies, with Batch

Normalization and Dropout used to reduce overfitting.
Finally, the data passes through Dense layers with Dropout, and the final classification is performed using a Softmax layer.The adopted CNN–BiGRU architecture is illustrated in Figure 2.12.

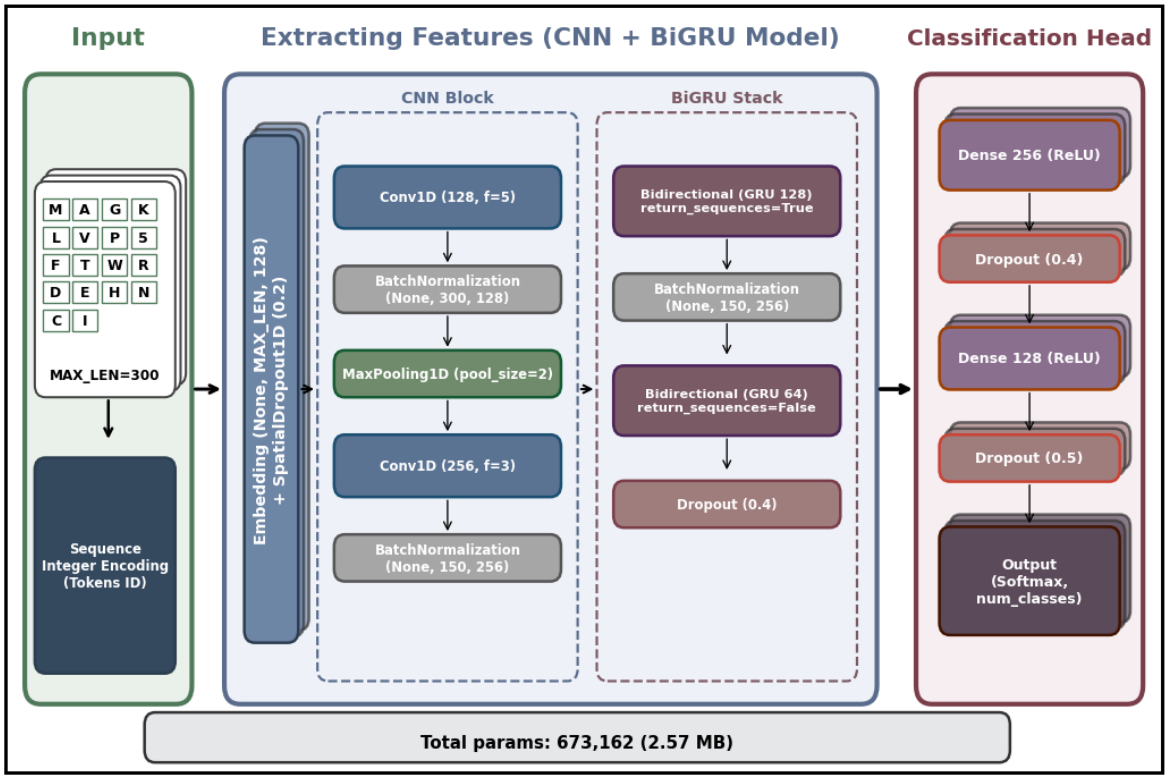


Figure 2.12 : Proposed CNN-BiGRU Architecture.

CNN-BiLSTM-Attention : The CNN-BiLSTM-Attention model is adopted using the same architecture as the CNN-BiLSTM model, with the addition of an Attention layer that assigns different weights to the features extracted from the BiLSTM layer in order to focus on the most important elements within the sequence.This is illustrated in Figure 2.13. Some modifications to the hyperparameters are also introduced according to the model requirements.

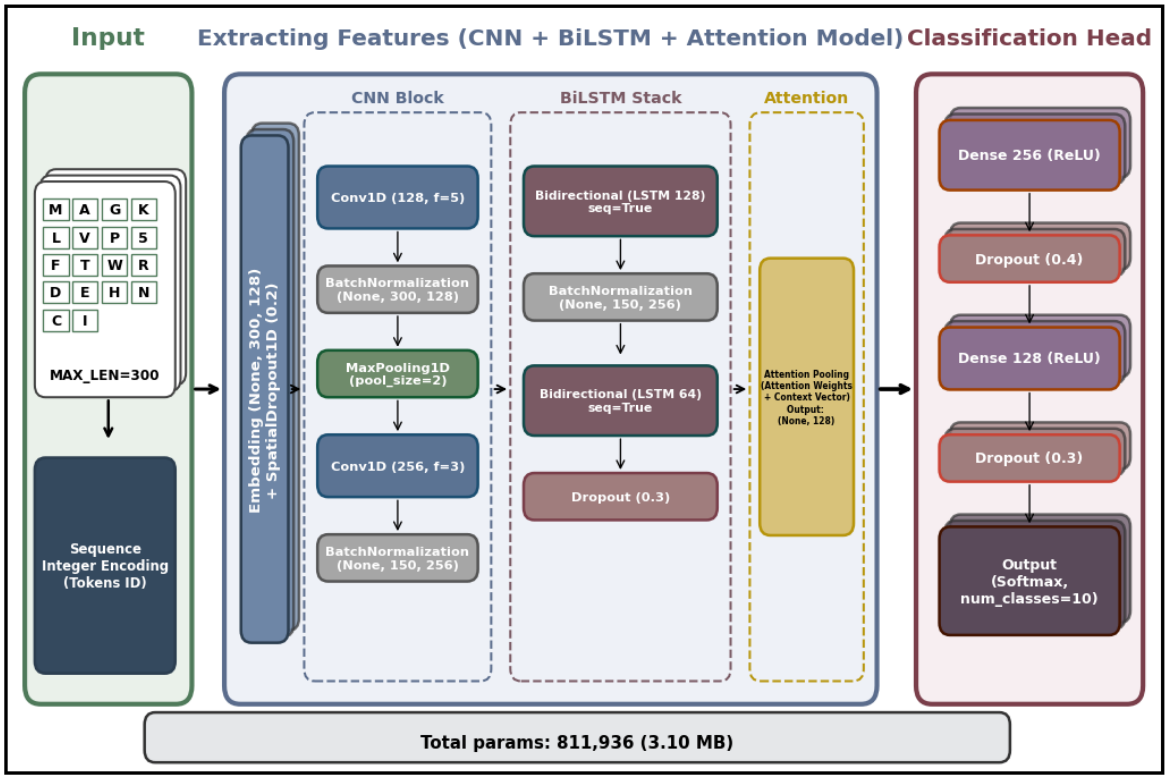


Figure 2.13 : Proposed CNN-BiLSTM-Attention Architecture.

CNN-BiGRU-Attention : The CNN-BiGRU-Attention model follows the same overall architecture as the CNN-BiGRU model, with the integration of an Attention layer that learns to assign different importance weights to the features produced by the BiGRU layer in order to focus on the most important elements within the sequence. This architecture is illustrated in Figure 2.14. Some modifications to the hyperparameters are also introduced according to the model requirements.

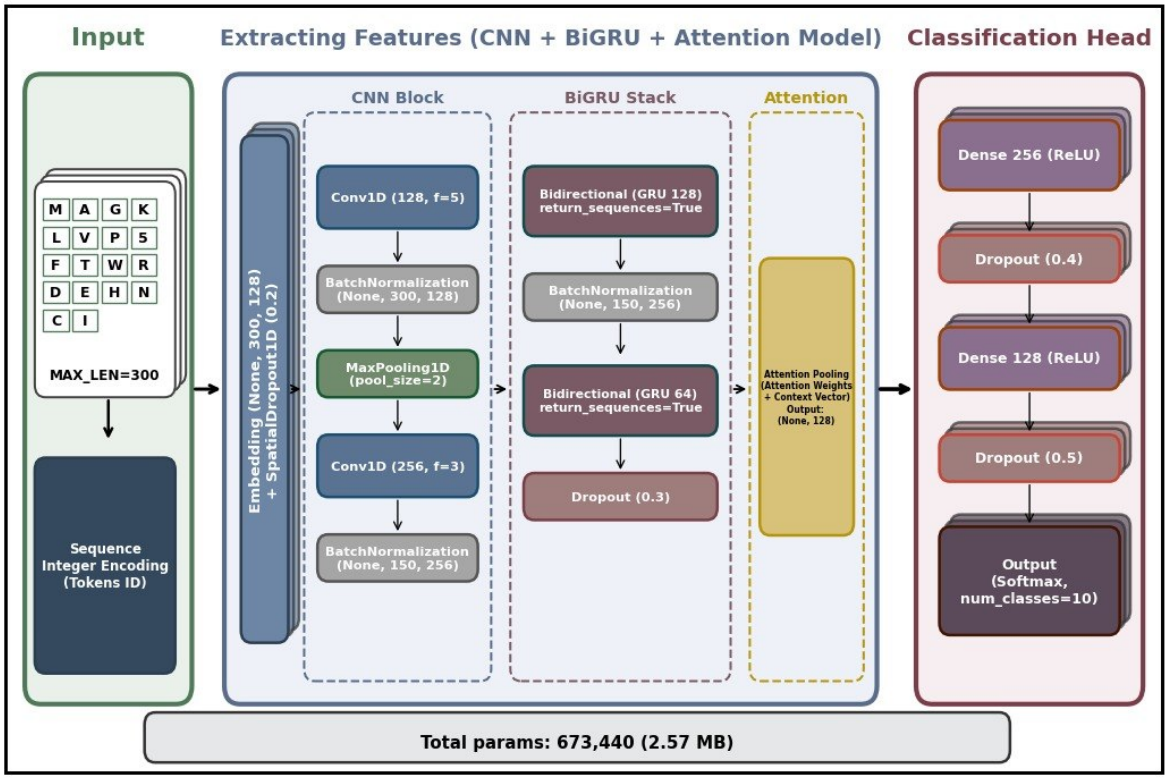


Figure 2.14 : Proposed CNN-BiGRU-Attention Architecture.

Hyperparameter Tuning and Configuration : The proposed models are trained using the same pipeline, with different hyperparameter values being tested. The final values are selected based on performance on the validation set. Table below 2.3 presents the hyperparameters used for the hybrid models.

Table 2.3 : Hyperparameter Settings of Hybrid Models Based on Raw Sequences Representation.

Model	Learning Rate	Batch Size	Number of Epochs	Optimizer	Dropout	Spatial Dropout1D
CNN–BiLSTM	0.0001	128	30	Adam	0.4/0.5	0.2
CNN–BiGRU	0.0003	128	30	Adam	0.4/0.5	0.2
CNN–BiLSTM–Attention	0.0001	128	50	Adam	0.3/0.4	0.2
CNN–BiGRU–Attention	0.0003	128	50	Adam	0.3/0.4/0.5	0.2

2.4.3.2 Hybrid Models Based on k-mer Representation

Several hybrid architectures based on the K-mer representation technique are employed, with the hyperparameters of each model optimized.

Hybrid Models Architectures

CNN-BiLSTM : The CNN-BiLSTM is used to combine the strengths of both architectures in extracting and learning patterns within protein sequences.
The input layer receives the numerical representations of the sequences, which are then transformed into trainable vectors using an Embedding layer. Next, a CNN1D layer is applied to extract important features, with Batch Normalization is used to improve training stability and Max Pooling to reduce dimensionality. The extracted representations are then passed through two BiLSTM layers to learn bidirectional sequential relationships and enhance contextual representation. Finally, the data passes through Dense layers with Dropout to reduce overfitting, and the final classification is performed using a Softmax layer. This is illustrated in Figure 2.15.

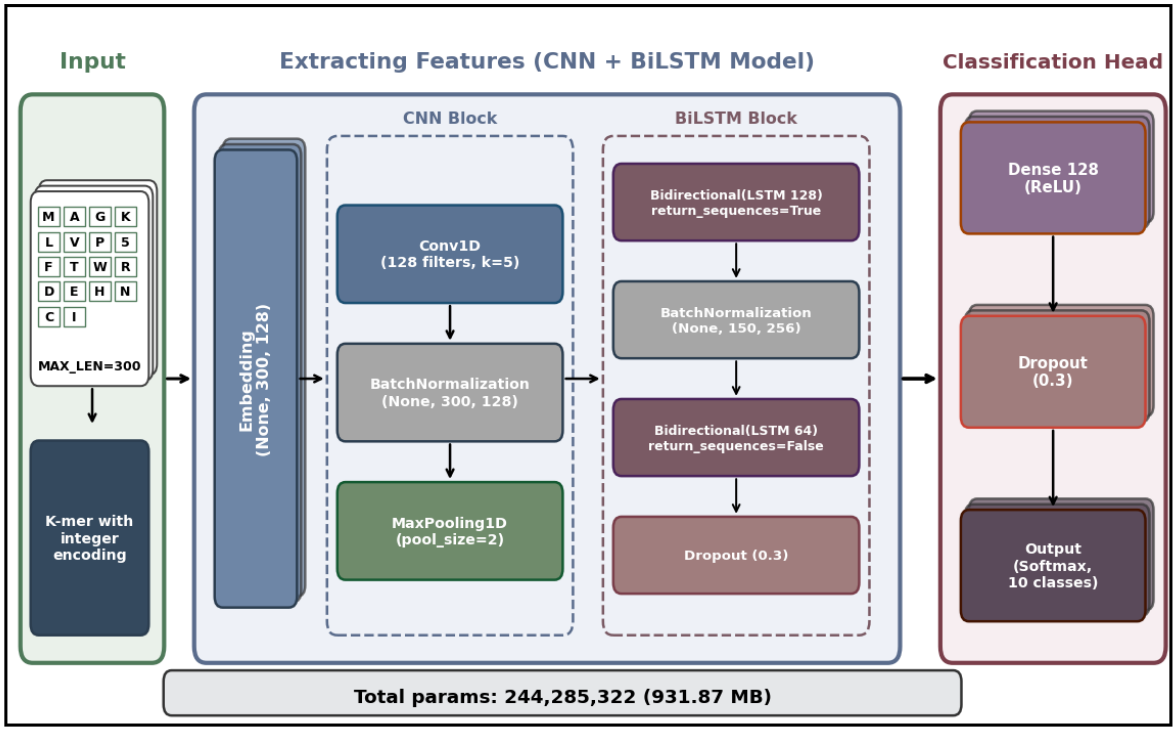


Figure 2.15 : Proposed CNN-BiLSTM Architecture.

CNN-BiGRU : The CNN-BiGRU hybrid model extracts and learns complex patterns within protein sequences. The numerical representations of the sequences are converted into vectors through an Embedding layer. Two CNN layers extract important local features, with Batch Normalization applied to improve training stability, Max Pooling to reduce dimensionality, and Dropout to reduce overfitting. The extracted representations are passed through a single BiGRU layer to capture sequential relationships in both directions, with Global Max Pooling used to extract the most important features. Finally, the data is passed through Dense layers with Dropout, and the final classification is performed using a Softmax layer. The architecture of the CNN-BiGRU model is presented in Figure 2.16.

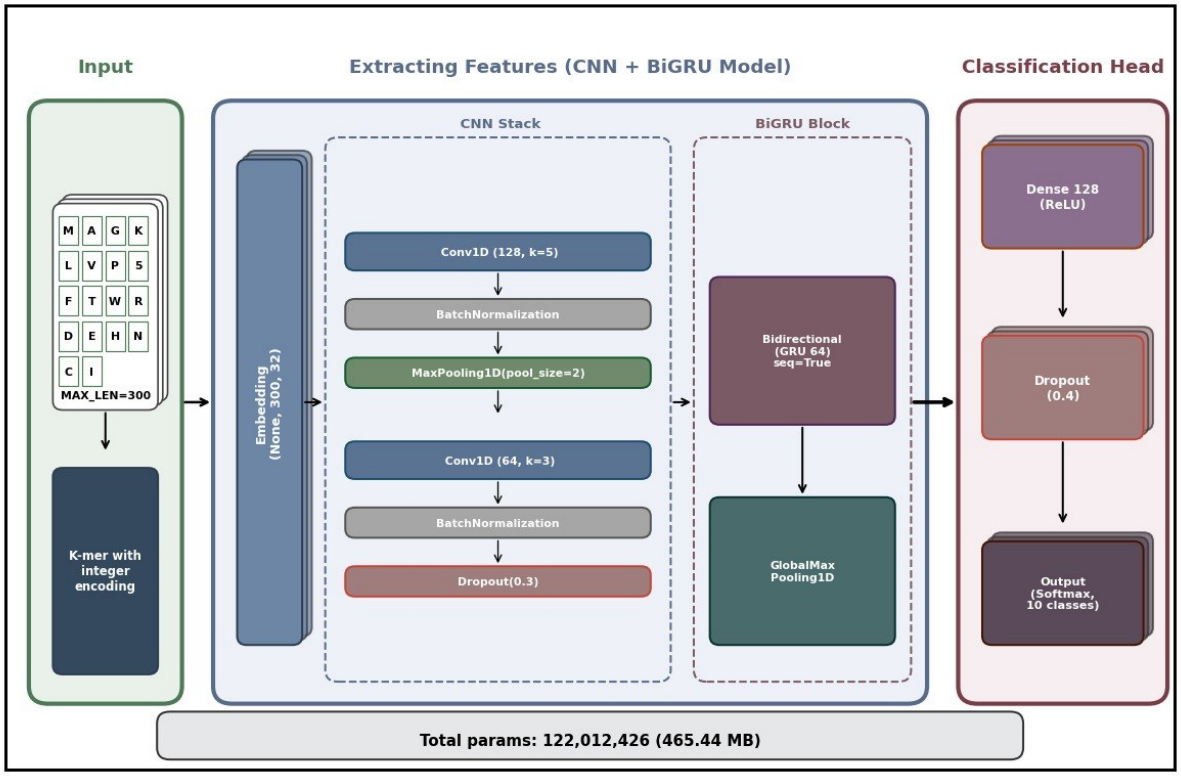


Figure 2.16 : Proposed CNN-BiGRU Architecture.

CNN-Self-Attention : The CNN-Self-Attention is used to benefit from the strengths of both architectures in extracting patterns and identifying the most important regions within protein sequences.
The input layer receives the numerical sequence representations, which are then transformed into trainable vectors using an Embedding layer. Next, two 1D-CNN layers are applied to extract important local patterns, with Batch Normalization used to improve training stability and Max Pooling employed to reduce dimensionality.
The extracted representations are then passed through a Self-Attention mechanism, which enables the model to focus on the most important elements within the sequence. Global Max Pooling is subsequently used to extract the most relevant features. Finally, the data passes through Dense layers with Dropout to reduce overfitting, and the final classification is performed using a Softmax layer. This is illustrated in Figure 2.17.

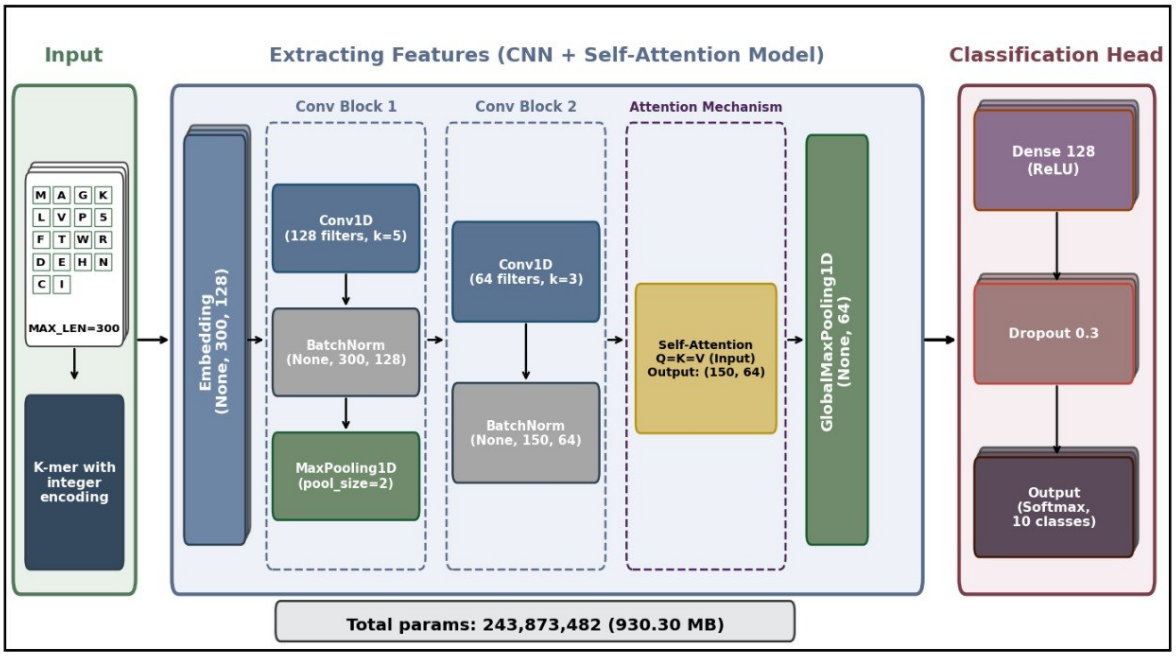


Figure 2.17 : Proposed CNN-Self-Attention Architecture.

CNN-BiLSTM-Attention : The hybrid CNN-BiLSTM-Attention model is adopted to benefit from the advantages provided by the three architectures. An Attention layer is added to the proposed CNN-BiLSTM model to assign different weights to the features extracted from the BiLSTM layers, allowing the model to focus on the most important elements within the sequence.This is illustrated in Figure 2.18. The architectural settings and hyperparameter values are maintained as in the proposed CNN-BiLSTM model.

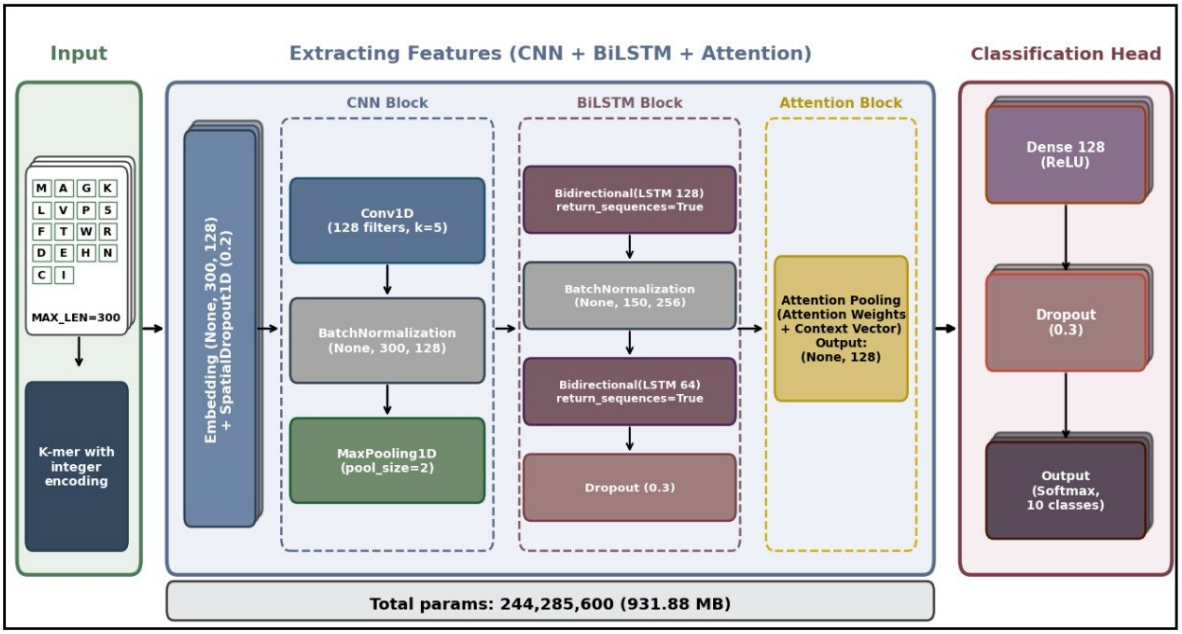


Figure 2.18 : Proposed CNN-BiLSTM-Attention Architecture.

CNN-BiGRU-Attention : The CNN-BiGRU-Attention model is used to combine local pattern extraction and bidirectional sequential dependency learning, while enhancing the focus on important regions within protein sequences. A structure similar to the CNN-BiGRU model is adopted, with the removal of GlobalMaxPooling1D and the addition of an Attention layer that assigns different weights to the features extracted from the BiGRU layer. This allows the model to enhance features that are more relevant to the classification task.This architecture is illustrated in Figure 2.19. Some modifications to the hyperparameters are also introduced according to the model requirements.

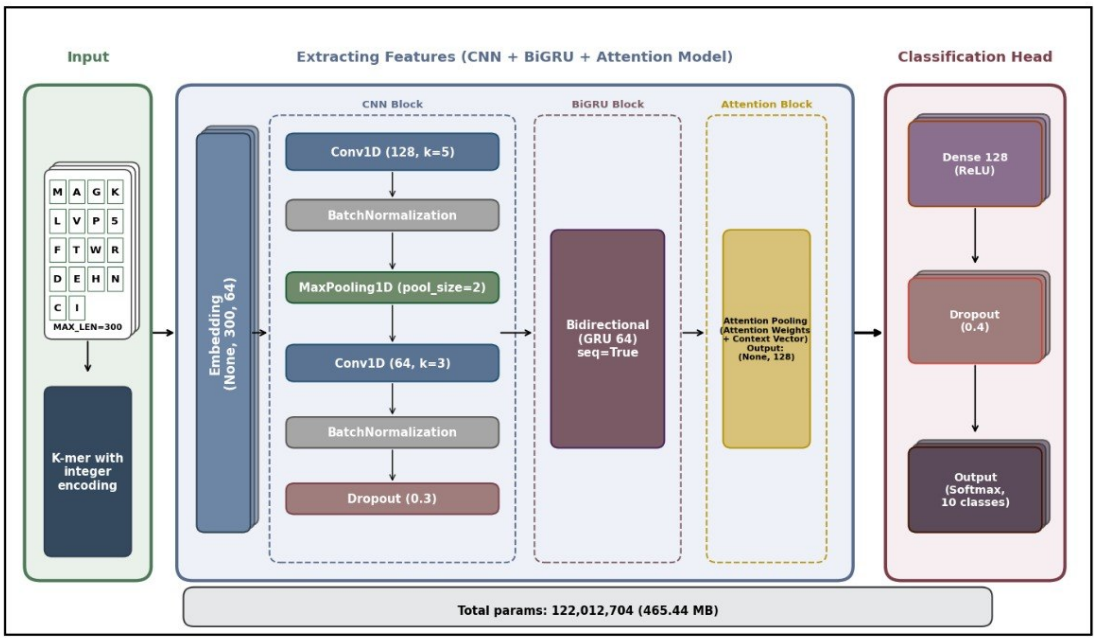


Figure 2.19 : Proposed CNN-BiGRU-Attention Architecture.

Hyperparameter Tuning and Configuration : The proposed hybrid models are trained using the same tuning pipeline, with different hyperparameter values being tested. The final values are selected based on performance on the validation set.The following Table 2.4 presents the hyperparameters used in the hybrid models.

Table 2.4 : Hyperparameter Settings of Hybrid Models Based on k-mer Representation.

Model	**Learning Rate**	**Batch Size**	**Number of Epochs**	**Optimizer**	**Dropout**	**Spatial Dropout1D**
CNN–BiLSTM	0.0003	64	30	Adam	0.3	0.2
CNN–BiGRU	0.0001	64	100	Adam	0.3/0.4	/
CNN–Self-Attention	0.0003	64	30	Adam	0.3	/
CNN–BiLSTM–Attention	0.0003	64	30	Adam	0.3	0.2
CNN–BiGRU–Attention	0.001	64	50	Adam	0.3/0.4	/

2.4.4 Hybrid Ensemble Learning

The Hybrid Ensemble Learning approach is adopted to improve classification performance by combining hybrid deep learning models with machine learning models. The ensemble models are built based on the CNN-BiLSTM-Attention and CNN-BiGRU-Attention architectures, as presented in the Hybrid Models with k-mer representation section, while maintaining the same architecture and hyperparameters and integrating them with Random Forest and XGBoost models.

Random Forest is selected due to its ability to handle high-dimensional data and reduce overfitting, whereas XGBoost is adopted because of its ability to enhance performance using Gradient Boosting, which reduces errors and improves classification accuracy, with hyperparameters tuned according to each model.

This approach relies on training the hybrid model to extract features from protein sequences, where the extracted features are used as inputs to the machine learning model. Both the hybrid model and the machine learning model independently generate prediction probabilities, after which a Weighted Soft Voting strategy is applied to combine the outputs. The optimal weights are determined using the validation set by testing different weight values and selecting those that achieve the best classification accuracy. Finally, the prediction probabilities are combined using these optimized weights to obtain the final classification result, as illustrated in the Figure 2.20.

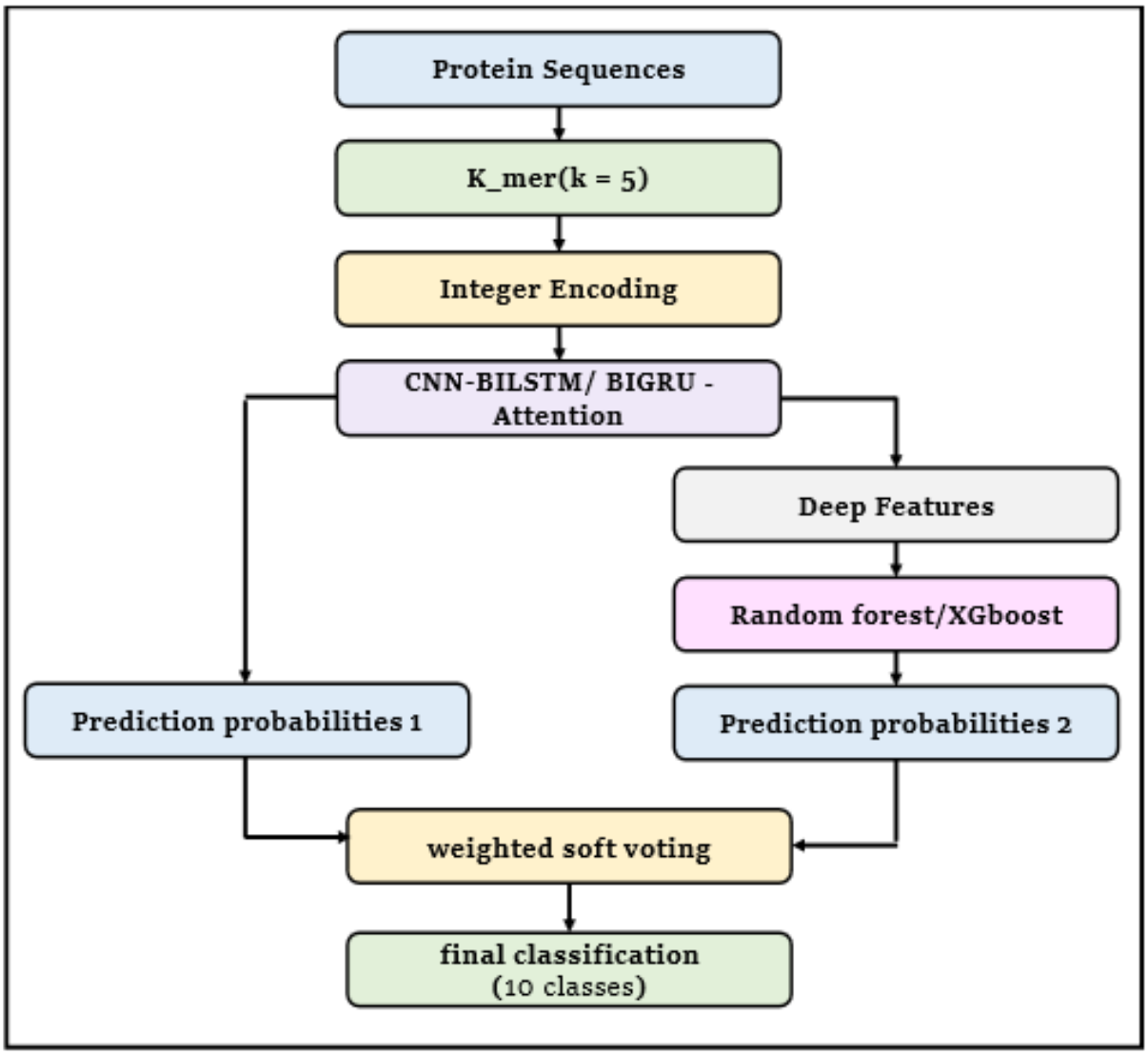


Figure 2.20 : Hybrid Ensemble Learning Pipeline.

The accompanying Table 2.5 presents the hyperparameters of both the Random Forest model and the XGBoost model used within the Hybrid Ensemble Learning approach after the tuning process.

Table 2.5 : Hyperparameter Settings of Hybrid Ensemble Learning Models.

Model	**Hyperparameters**
RandomForest	n_estimators=300, max_depth=20, n_jobs=-1
XGBoost	n _estimators=300, max _depth=6, learning _rate=0.05, subsample=0.8, colsample _by-tree=0.8, objective='multi :softprob', n _jobs=-1, num _class=num_classes

2.4.5 Transfer Learning

To improve classification performance on the target dataset, transfer learning is employed to leverage the knowledge acquired from large pre-trained protein language models for analyzing protein sequences and extracting biological relationships between amino acids within the sequences.
Two pre-trained models are used in this study, namely ProtBERT and ESM.

ProtBERT is used as a feature extraction model by converting each protein sequence into dense numerical representations, with a maximum sequence length of 512 amino acids. The data is also split into batches (batch size = 8) to reduce memory consumption and accelerate processing. A Mean Pooling strategy with an attention mask is applied to obtain a final representation for each sequence. Finally, these representations are used as input features for machine learning models to perform the classification task.
The **ESM** model is used in two different ways :

1. It is employed as a feature extractor similarly to ProtBERT. Protein sequences are converted into numerical representations by splitting the data into batches (batch size = 8) with a maximum sequence length of 1024 to match the model capability of processing long sequences. A Mean Pooling strategy is applied along with an Attention Mask to ensure an accurate final representation of the sequences. Finally, the resulting embeddings are saved and used as input features for machine learning models.

2. The ESM model is also used in a Fine-Tuning setting to adapt it to the target data and the protein sequence classification task. The model is fully fine-tuned by unfreezing all its layers and adding a task-specific classification head on top of the representation layers, allowing contextual embeddings to be transformed into predictions for the target classes.

Protein sequences are processed by converting them into numerical tokens using the models tokenizer, with a maximum sequence length set to (512) and data split into batches to ensure efficient processing. The sequences are then passed through the model to obtain deep contextual representations for each amino acid, which are then used by the final classification layer to generate predictions. Table 2.6 presents the hyperparameters of the machine learning models used after feature extraction using ProtBERT and ESM.

Table 2.6 : Hyperparameter Settings of Machine Learning Models with ProtBERT and ESM Embeddings.

Model	**Hyperparameters (Protbert)**	**Hyperparameters (ESM)**
SVM	kernel = linear, C=1.0, max_iter=5000	kernel linear, max_iter=5000
Logistic Regression	max_iter=50000	max_iter=3000, solver='lbfgs', multi_class='auto'
Naive Bayes	priors=None, var_smoothing=1e-9	priors=None, var_smoothing=1e-9
KNN	n_neighbors=5, weights='distance' metric='minkowski', p=2	n_neighbors=5
Decision Tree	max_depth=20	max_depth=20, min_samples_split=5
Random Forest	n_estimators=200, max_depth=None, n_jobs=-1	n_estimators=100, max_depth=20, n_jobs=-1
XGBoost	objective='multi :softprob', num_class=num_classes, n_estimators=200, max_depth=6, learning_rate=0.1, subsample=0.8, colsample_bytree=0.8, eval_metric='mlogloss'	n_estimators=200, learning_rate=0.1, max_depth=6, subsample=0.8, colsample_bytree=0.8, eval_metric='mlogloss'

Table 2.7 shows the hyperparameters of the ESM model with fine-tuning.

Table 2.7 : Hyperparameter Settings for the Fine-Tuned ESM Model

Model	**Learning Rate**	**Batch Size**	**Number of Epochs**	**Optimizer**
ESM	0.00002	8	20	Adam

2.5 Training Strategy

A unified training strategy is adopted to evaluate the proposed models in this study. The dataset is split into 80% for training and 20% for testing for machine learning models and feature extraction models. For deep learning, hybrid, hybrid ensemble learning models, and the ESM fine-tuning model, a more detailed split is used, consisting of 70% training, 15% validation, and 15% testing.
The validation set is used during training to monitor model performance and tune hyper-parameters, while the test set is reserved for final evaluation.
Class Weights are used to address class imbalance in the dataset, allowing the models to give more importance to underrepresented classes and improve classification performance.
Several techniques are applied to improve training stability and avoid overfitting, including Early Stopping, which stops training when validation performance stops improving, and ReduceLROnPlateau, which gradually reduces the learning rate when the loss stagnates.
Finally, Checkpointing is used with large protein language models to allow training to resume from the last saved state and to store the best-performing model during training.

2.6 Evaluation Methodology

This section presents the methodology adopted to measure and analyze the performance of the proposed models, as well as to evaluate their generalization ability, in order to ensure the reliability and stability of the results.

2.6.0.1 Evaluation Metrics and Confusion Matrix

The models performance is evaluated using a set of standard classification metrics to provide a comprehensive and accurate assessment of model effectiveness. Accuracy is used to measure the proportion of correct predictions among all samples, Precision to evaluate the ability of the models to reduce false predictions, and Recall to measure the ability of the models to correctly identify relevant samples for each class. In addition, F1-score, which represents the harmonic mean of Precision and Recall, is also used. The Weighted Average values of Precision, Recall, and F1-score are adopted for presenting the results, in order to consider the number of samples within each class by assigning a weight proportional to the size of each class, making the evaluation more suitable for imbalanced datasets. In addition, the Confusion Matrix is used to provide a detailed analysis by illustrating the distribution of correct and incorrect predictions across all classes, allowing a better understanding of model's behavior.

2.6.0.2 Cross-Validation Strategy

Cross Validation is a technique used to evaluate the generalization capability of a model by dividing the dataset into several folds, where the model is trained on a portion of the data and its performance is evaluated on the remaining folds. This process is repeated several times, and the final performance is calculated by taking the average of the obtained results across all folds, providing a more reliable evaluation of model performance and helping reduce the overfitting problem.
Cross-validation is applied to the best-performing model by splitting the dataset into five folds. In each iteration, three folds are used for training, one fold for validation,

and one fold for testing. The process is repeated so that each fold is served in different roles, ensuring evaluation on unseen data at each iteration and providing a more robust evaluation. Finally, the overall performance of the model is obtained by averaging the results across all five folds.

2.6.0.3 Generalization Evaluation

A set of external data is used to evaluate the generalization capability of the models. Due to the difficulty of obtaining additional data for protein sequence classification, a set of protein sequences is collected from the National Center for Biotechnology Information (NCBI) Website[29] by searching for sequences belonging to the same classes present in the original dataset. Protein sequences belonging to the UniProt database are used and extracted in FASTA format, which is suitable for biological sequence processing. After the collection process, approximately 400 protein sequences belonging to 9 functional classes are obtained, while ensuring that there is no overlap with the original dataset to guarantee the reliability of the evaluation. These data are then used as an external test set to evaluate the performance of the best-performing model in this study and assess its generalization capability.

2.7 Conclusion

This chapter presents the methodology of protein sequence classification adopted in the study of protein sequence functional classification. It covers the stages of data preparation, sequence representation techniques, and feature extraction, as well as the different approaches used, including machine learning models, deep learning models, hybrid DL models, hybrid ensemble learning models, and transfer learning using advanced protein language models. The chapter also addresses the training and evaluation strategies adopted to ensure reliable and accurate performance. This methodology enables a fair comparison between all studied models and provides a reliable basis for evaluating their effectiveness in protein classification tasks. Following this evaluation strategy, the next chapter is dedicated to present the experimental results and discuss the performance of the proposed models in detail.

Chapter 3

Implementation Framework and Experimental Analysis

Chapter 3

Implementation Framework and Experimental Analysis

3.1 Introduction

This chapter presents the working environment and resources used for developing and training the models, in addition to presenting the results of the different models used and analyzing their performance in the protein sequence classification task.
It also includes the study of the impact of sequence representation on classification performance.
Furthermore, this chapter discusses the generalization ability of the models and evaluates them on external data, in addition to error analysis and the study of the impact of dataset composition on performance.
Finally, the chapter addresses the analysis of model complexity and computational cost, and presents a comprehensive comparison of the study results with previous works.

3.2 Experimental Design

All experiments are conducted using the same data preprocessing pipeline and the same training strategy. The same evaluation metrics are also adopted, in order to ensure a comprehensive and fair comparison between the different approaches used in the protein sequence classification task.

3.2.1 Development Environment

All experiments are developed and implemented using a unified programming environment, which is based on a set of specialized tools, libraries, and required computational resources, enabling the construction and training of different models.

3.2.1.1 Programming Language

In this study, **Python** is adopted as the primary programming language due to its simplicity, flexibility, and support for machine learning and deep learning domains, in addition to its wide range of specialized libraries that facilitate data processing, model building, and efficient performance evaluation.

3.2.1.2 Libraries and Frameworks

A variety of specialized libraries and frameworks are used, including :

- **Pandas and NumPy :** for data processing and organization, as well as facilitating numerical operations.
- **Scikit-learn :** for developing and evaluating traditional machine learning models.

- **TensorFlow and PyTorch :** for building and training deep learning models, in addition to **Keras** to facilitate model construction and development.
- **Hugging Face Transformers :** for running pretrained models and extracting representations.
- **Matplotlib and Seaborn :** for visualizing results and analyzing model performance.

3.2.1.3 Tools

Models are developed and implemented using **Google Colab** and **Kaggle**. Colab is an interactive cloud-based environment developed by Google, built on Jupyter Notebook, which allows code execution and model development with free GPU support. Kaggle is a platform dedicated to data science and machine learning, also providing a Jupyter Notebook-based execution environment with free computing resources such as GPUs. In this study, **Google Drive** is also used to store and organize data, models, and results, which facilitates file saving and synchronization across different working environments.

3.2.2 Hardware Specifications

Graphics Processing Units (GPUs) are used to accelerate the model training process in the Colab and Kaggle environments, such as Tesla T4 and P100. In addition, Random Access Memory (RAM), ranging between 12 and 16 GB, is utilized for efficient data processing and model execution.

3.3 Experimental Results

This section presents the results obtained on the test data using the different adopted approaches, in addition to providing an analysis of the results to evaluate the performance of the models.

3.3.1 Machine Learning Models Results

Table 3.1 presents the results of machine learning models developed in this study based on commonly used evaluation metrics. Figure 3.1 illustrates these results.

Table 3.1 : Performance Results of Machine Learning Models.

Model	**Accuracy (%)**	**Weighted Precision (%)**	**Weighted Recall (%)**	**Weighted F1-score (%)**
SVM	93.13	94.00	93.00	93.00
Logistic Regression	93.39	93.00	93.00	93.00
Naive Bayes	79.00	80.00	79.00	79.00
KNN	89.48	90.00	89.00	90.00
Decision Tree	87.94	88.00	88.00	88.00
Random Forest	93.72	94.00	94.00	94.00
XGBoost	86.85	88.00	87.00	87.00

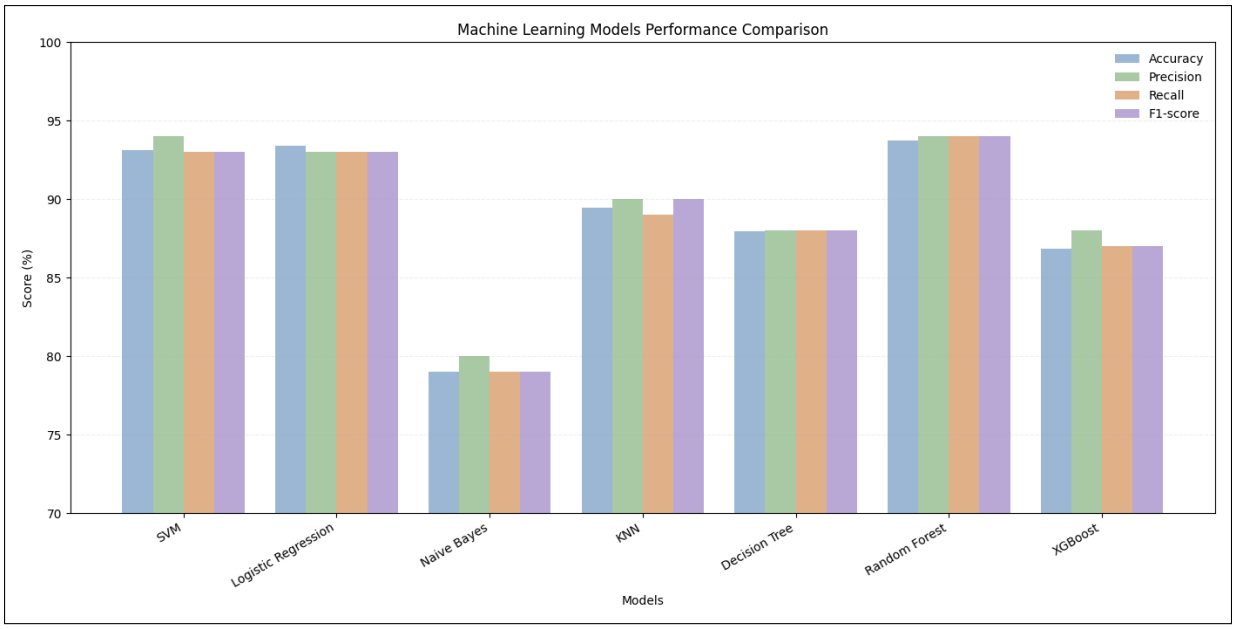


Figure 3.1 : Results of Machine Learning Models.

The obtained results shows clear differences in the performance of machine learning models for the protein sequence classification task. Among all evaluated models, Random Forest achieved the best performance with an accuracy of 93.72 %, in addition to obtaining 94 % for Precision, Recall, and F1-score. Furthermore, both Logistic Regression and SVM demonstrated competitive performance with close results, achieving accuracies of 93.39 % and 93.13 %, respectively, highlighting their ability to effectively exploit the extracted features. On the other hand, KNN, Decision Tree, and XGBoost achieved moderate performance compared to the previously mentioned models, while Naive Bayes exhibited the weakest performance with an accuracy of 79 %.
Overall, the results demonstrate the capability of traditional machine learning models to achieve good performance in protein sequence classification based on TF-IDF extracted features. The results also highlight the superiority of ensemble-based models such as Random Forest in handling these high-dimensional features in a more stable and effective manner.

3.3.2 Deep Learning Models Results

It represents the results of deep learning models obtained using One-Hot Encoding and K-mer representations.

3.3.2.1 Models Results Based on One-Hot Encoding Representations

The following Table 3.2. presents the performance results of the proposed CNN model using one-hot encoding for protein sequence representation.

Table 3.2 : Performance Results of CNN Model Based on One-Hot Encoding.

Model	**Loss**	**Accuracy (%)**	**Weighted Precision (%)**	**Weighted Recall (%)**	**Weighted F1-score (%)**
CNN	0.38	89.84	90.00	90.00	90.00

The CNN model based on the One-Hot Encoding representation achieved an accuracy of 89.84%, with a loss value of 0.38. The model also obtained identical values of 90% for Precision, Recall, and F1-score, reflecting its ability to extract local patterns from protein sequences using the One-Hot Encoding representation mthod and to reduce classification errors.

3.3.2.2 Models Results Based on K-mer Representations

The Table 3.3. presents the performance results of deep learning models based on k-mer representation.Figure 3.2 illustrates these results.

Table 3.3 : Performance Results of Deep Learning Models Based on k-mer Representation.

Model	**Loss**	**Accuracy (%)**	**Weighted Precision (%)**	**Weighted Recall (%)**	**Weighted F1-score (%)**
CNN	0.22	93.11	93.00	93.00	93.00
LSTM	0.30	91.77	92.00	92.00	92.00
BiLSTM	0.25	92.02	92.00	92.00	92.00
ETE	0.38	93.03	93.00	93.00	93.00

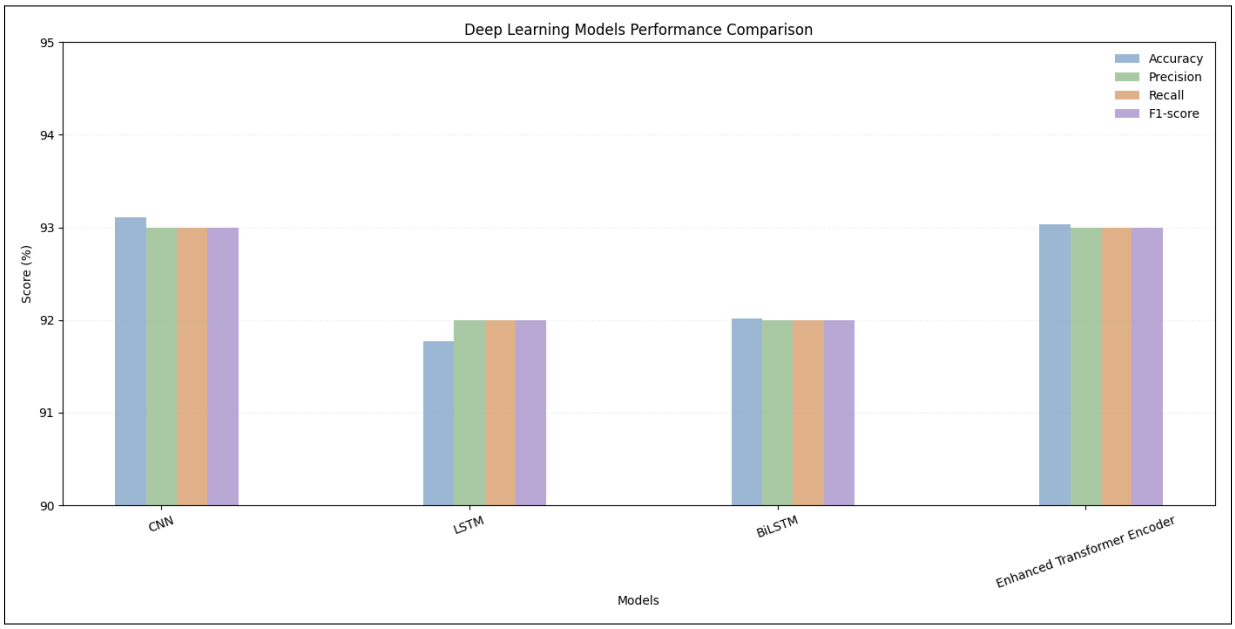


Figure 3.2 : Results of Deep Learning Models Based on k-mer Representation.

The results shows that the CNN model achieved the best performance among the deep learning models used, with an accuracy of 93.11% and the lowest test loss of 0.22, in addition to achieving 93% for Precision, Recall, and F1-score. The Enhanced Transformer Encoder also demonstrated competitive performance with an accuracy of 93.03%, while both BiLSTM and LSTM achieved relatively lower performances with accuracies of 92.02% and 91.77%, respectively. The results indicate the ability of deep learning models to extract complex features and benefit from representations based on the K-mer technique. In particular, the CNN model showed superior performance and stability, highlighting its ability to effectively capture local patterns and important features from protein sequences.

3.3.3 Hybrid Models Results

It represents the results of hybrid models obtained using raw-sequences and K-mer representations.

3.3.3.1 Models Results Based on Raw Sequences

Table 3.4 presents the performance results of the proposed hybrid learning models using raw sequences representation. Figure 3.3 illustrates these results.

Table 3.4 : Performance Results of Hybrid Learning Models Based on Raw Sequences Representation.

Model	Loss	Accuracy(%)	Weighted Precision (%)	Weighted Recall (%)	Weighted F1-score (%)
CNN-BiLSTM	0.30	91.53	92.00	92.00	92.00
CNN-BiGRU	0.35	90.38	91.00	90.00	91.00
CNN-BiLSTM-Attention	0.29	92.45	93.00	92.00	92.00
CNN-BiGRU-Attention	0.31	91.28	91.00	91.00	91.00

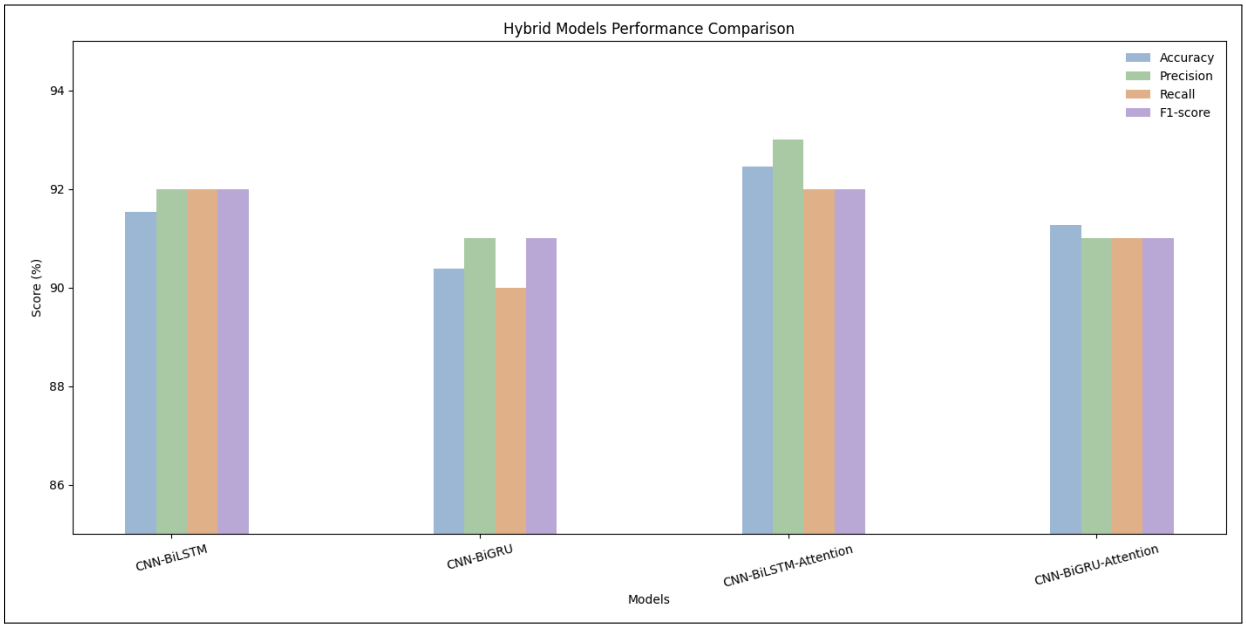


Figure 3.3 : Results of Hybrid Models Based on Raw Sequences Representation.

The results of the hybrid models based on raw sequence representation show differences in performance among the various architectures. The CNN-BiLSTM-Attention model achieved the best performance with an accuracy of 92.45% and the lowest loss value of 0.29. It also achieved 93% in Precision and 92% for both Recall and F1-score. In addition, both CNN-BiLSTM and CNN-BiGRU-Attention achieved similar performances with accuracies of 91.53% and 91.28%, respectively, while the CNN-BiGRU model recorded the lowest performance among the evaluated models, with an accuracy of 90.38% and a loss value of 0.35.
These results reflect the ability of hybrid models to effectively handle raw sequences, particularly the CNN-BiLSTM-Attention model, by benefiting from the strengths of each

architecture in extracting patterns and modeling sequential relationships among amino acids, in addition to improving the models ability to focus on the most important parts of the sequence.

3.3.3.2 Models Results Based on K-mer Representations

The following Table 3.5. presents the performance results of the proposed hybrid models using K-mer representation. Figure 3.4 illustrates these results.

Table 3.5 : Performance Results of Hybrid Learning Models Based on k-mer Representation.

Model	**Loss**	**Accuracy (%)**	**Weighted Precision (%)**	**Weighted Recall (%)**	**Weighted F1-score (%)**
CNN-BiLSTM	0.24	92.30	92.00	92.00	92.00
CNN-BiGRU	0.25	92.98	93.00	93.00	93.00
CNN-Self-Attention	0.22	93.15	93.00	93.00	93.00
CNN-BiLSTM-Attention	0.23	93.25	93.00	93.00	93.00
CNN-BiGRU-Attention	0.24	93.34	93.00	93.00	93.00

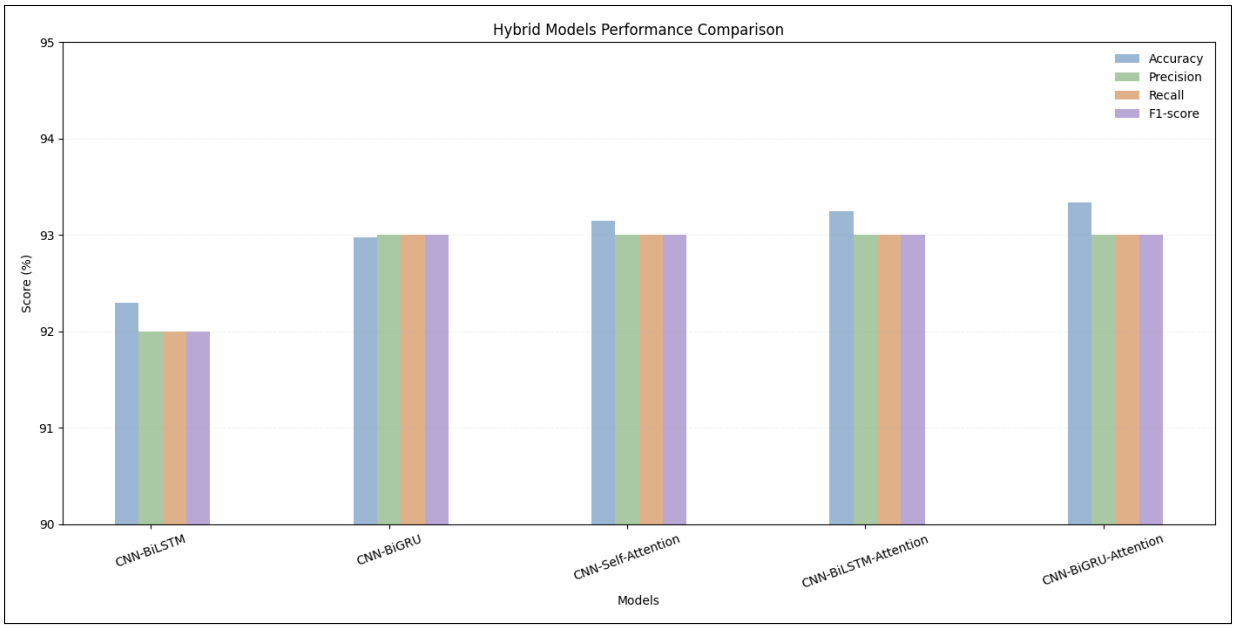


Figure 3.4 : Results of Hybrid Learning Models Based on K-mer Representation.

The results of the hybrid models show that the best performance for protein sequence classification was achieved by the CNN-BiGRU-Attention model, with an accuracy of 93.34% and a test loss of 0.24. It is followed by the hybrid models CNN-BiLSTM-Attention and CNN-Self-Attention, which achieved close performances with accuracies of 93.25% and 93.15%, respectively. Meanwhile, both CNN-BiGRU and CNN-BiLSTM recorded relatively lower performances with accuracies of 92.80% and 92.30%, respectively.
These results reflect the effectiveness of hybrid models in protein sequence classification, particularly the CNN-BiGRU-Attention model. This is achieved by combining the ability

of CNN to extract local features from sequences represented using K-mer, the capability of BiGRU to model sequential dependencies, and the effectiveness of the Attention mechanism in focusing on the most important parts of the sequence.

3.3.4 Hybrid Ensemble Learning Models Results

Table 3.6 below presents the the performance results of the hybrid ensemble learning models using K-mer representation. Figure 3.5 illustrates these results.

Table 3.6 : Performance Results of Hybrid Ensemble Learning Models.

Model	**Accuracy (%)**	**Weighted Precision (%)**	**Weighted Recall (%)**	**Weighted F1-score (%)**
CNN-BiLSTM-Attention + Random Forest	94.04	94.00	94.00	94.00
CNN-BiLSTM-Attention + XGBoost	93.96	94.00	94.00	94.00
CNN-BiGRU-Attention + Random Forest	94.00	94.00	94.00	94.00
CNN-BiGRU-Attention + XGBoost	93.84	94.00	94.00	94.00

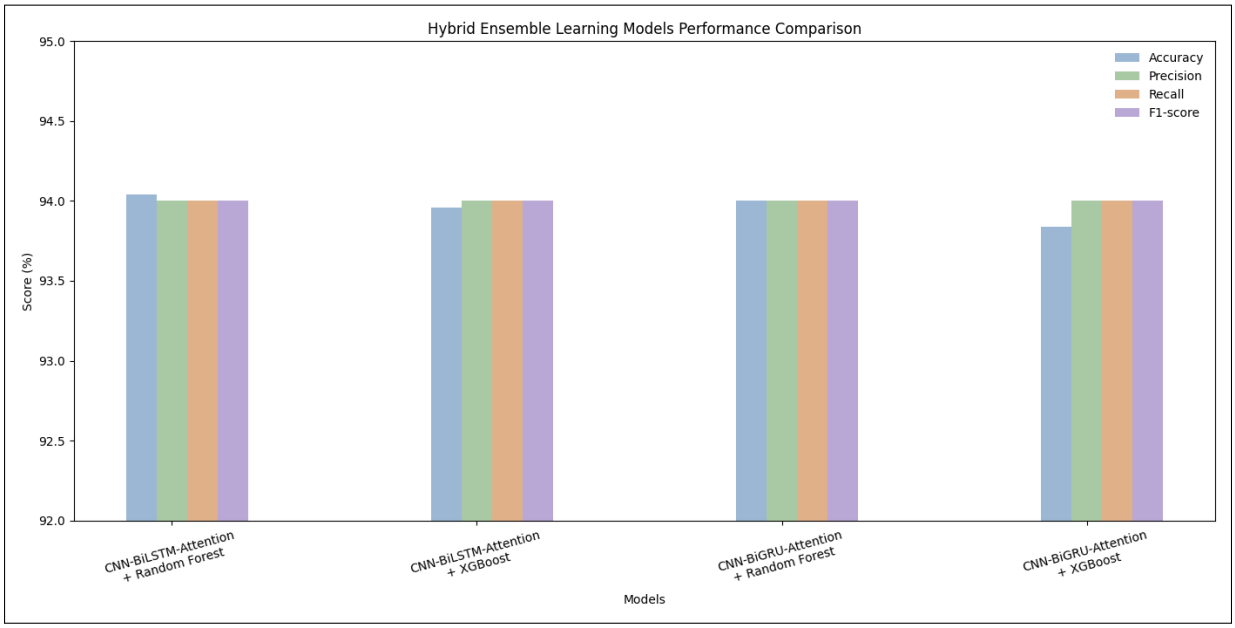


Figure 3.5 : Results of Hybrid Ensemble Learning Models .

The results of the hybrid ensemble models highlights the superiority of the CNN-BiLSTM-Attention + Random Forest model, which achieved an accuracy of 94.04%, along with 94% for Precision, Recall, and F1-score. In addition, the CNN-BiGRU-Attention + Random Forest model achieved a very close performance with an accuracy of 94.00%. In contrast, both CNN-BiLSTM-Attention + XGBoost and CNN-BiGRU-Attention + XGBoost achieved accuracies of 93.96% and 93.84%, respectively, with identical values for the remaining evaluation metrics.
These results demonstrate the ability of ensemble learning models to improve protein sequence classification performance, particularly the CNN-BiLSTM-Attention + Random

Forest model. This improvement is achieved through the contribution of hybrid deep learning models in extracting discriminative features and important relationships within biological sequences. In addition, machine learning models exploit these extracted features to further enhance the final classification process.

3.3.5 Transfer Learning Models Results

It presents the results of transfer learning models based on both feature extraction and fine-tuning approaches using the ProtBERT and ESM models.

3.3.5.1 Results Using ProtBERT Embeddings

The following Table 3.7 presents the results of the machine learning models based on ProtBERT embeddings. To illustrate these results, the corresponding Figure 3.6. shows a comparison of the performance of the hybrid learning models.

Table 3.7 : Performance Results of Machine Learning Models Using ProtBERT Embeddings.

Model	**Accuracy (%)**	**Weighted Precision (%)**	**Weighted Recall (%)**	**Weighted F1-score (%)**
SVM	79.27	81.00	79.00	79.00
Logistic Regression	78.95	81.00	79.00	79.00
Naive Bayes	38.97	49.00	39.00	37.00
KNN	88.88	89.00	89.00	89.00
Decision Tree	79.41	80.00	79.00	79.00
Random Forest	88.36	88.00	88.00	88.00
XGBoost	92.00	92.00	92.00	92.00

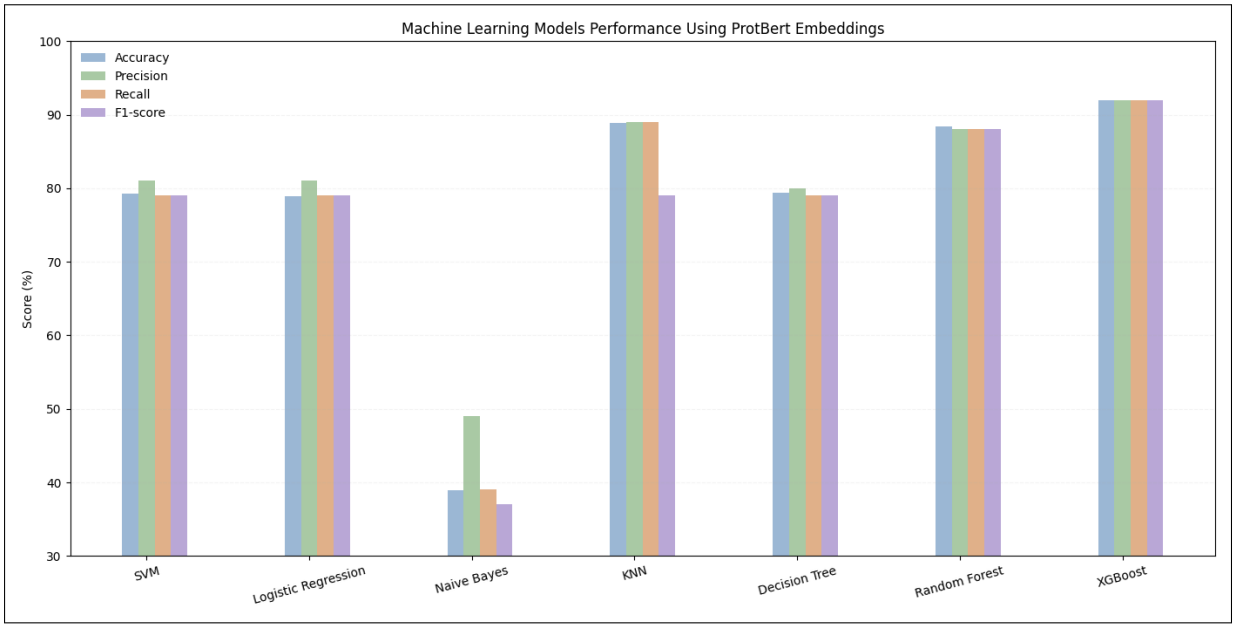


Figure 3.6 : Results of Machine Learning Models Performance based on ProtBERT Embeddings.

The results obtained using ProtBERT embeddings reveal differences in the performance of machine learning models. The XGBoost model achieved the best performance with an

accuracy of 92%, while also obtaining 92% for Precision, Recall, and F1-score. Meanwhile, both KNN and Random Forest achieved similar results, with accuracies of 88.88% and 88.36%, respectively. In contrast, Logistic Regression, SVM, and Decision Tree exhibited relatively similar performance, with accuracies ranging between 78% and 79%. However, Naive Bayes recorded the weakest performance with an accuracy of 38.97%.
These results reflect the effectiveness of ProtBERT representations in extracting rich features from protein sequences and capturing relevant biological information. Models based on boosting and ensemble learning techniques, particularly XGBoost, tend to be more capable of exploiting these representations and converting them into accurate classification decisions compared to the other models.

3.3.5.2 Results Using ESM Embeddings

Table 3.8 presents the results of the machine learning models based on ESM embeddings. To illustrate these results, the corresponding Figure 3.7. shows a comparison of the performance of the hybrid learning models.

Table 3.8 : Performance Results of Machine Learning Models Using ESM Embeddings.

Model	**Accuracy (%)**	**Weighted Precision (%)**	**Weighted Recall (%)**	**Weighted F1-score (%)**
SVM	81.12	80.00	79.00	81.00
Logistic Regression	82.76	82.00	81.00	83.00
Naive Bayes	48.02	52.00	55.00	48.00
KNN	92.54	91.00	91.00	92.00
Decision Tree	90.60	90.00	89.00	91.00
Random Forest	94.84	95.00	94.00	95.00
XGBoost	94.38	94.00	93.00	94.00

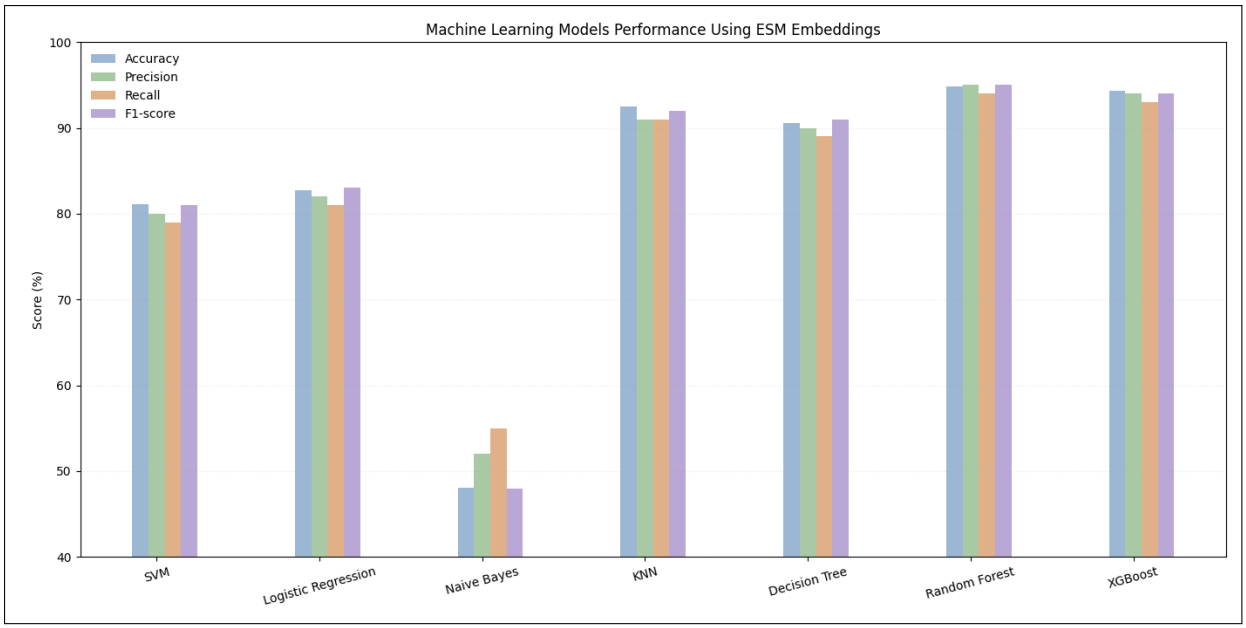


Figure 3.7 : Results of Machine Learning Models Performance based on ESM Embeddings.

The results obtained using ESM embeddings show variations in the performance of machine learning models. The Random Forest model achieved the best performance with

an accuracy of 94.84%, along with 95% for Precision and F1-score, and 94% for Recall. It was followed by the XGBoost model with a close performance, achieving an accuracy of 94.38%. Meanwhile, both KNN and Decision Tree achieved accuracies of 92.54% and 90.60%, respectively. In contrast, Logistic Regression and SVM showed lower performance with accuracies of 82.76% and 81.12%, respectively. Naive Bayes recorded the weakest performance with an accuracy of 48.02%.
The results indicate the effectiveness of machine learning models in handling protein features extracted using ESM, particularly ensemble and boosting methods such as Random Forest and XGBoost, which effectively exploit these representations and improve classification performance, with Random Forest achieving the best performance in protein sequence classification.

3.3.5.3 ESM Fine-Tuned Model Results

The corresponding Table 3.9 presents the results of the ESM model after fine-tuning technique.

Table 3.9 : Performance Results of the Fine-Tuned ESM Model.

Model	**Loss**	**Accuracy (%)**	**Weighted Precision (%)**	**Weighted Recall (%)**	**Weighted F1-score (%)**
ESM	0.22	94.95	95.00	95.00	95.00

The results obtained using the ESM model after fine-tuning show that the model achieved a classification accuracy of 94.95% with a loss value of 0.22, in addition to obtaining 95% for Precision, Recall, and F1-score.
These results indicate the model's ability to learn directly from protein sequences and capture biological relationships within them, allowing it to extract more specialized and accurate representations compared to fixed embeddings, which positively impacts protein functional classification performance.

3.4 Comparative Analysis and Discussion

This section aims to provide a comprehensive comparative analysis and discussion of the obtained experimental results. Through the evaluation of the studied models and the interpretation of the achieved results, it highlights the key factors influencing model effectiveness and reliability.

3.4.1 Intra-Study Model Comparison

The comparison among the different models investigated in this study revealed variations in performance across the different approaches. The corresponding Figure 3.8 shows a comparison of the best performing models from the different studied methods.

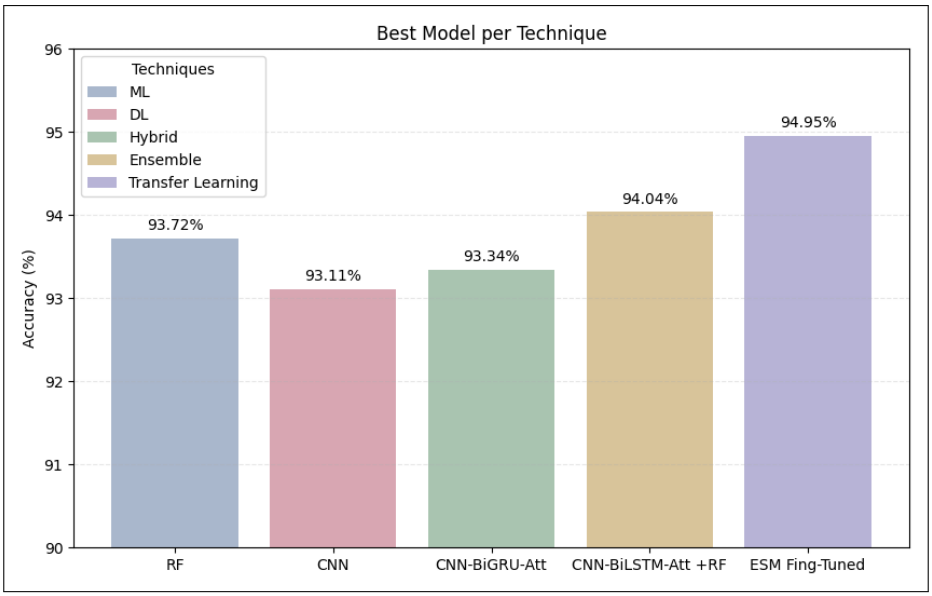


Figure 3.8 : Comparison of the Best Models Across Different Proposed Methods.

Among the machine learning models, Random Forest achieved the best performance with an accuracy of 93.72%, while the CNN model achieved the highest accuracy among the deep learning models, reaching 93.11% when using the K-mer technique with Integer Encoding. Regarding the hybrid models, the CNN-BiGRU-Attention model achieved the best performance when using K-mer representations, with an accuracy of 93.34%. Hybrid ensemble learning models also demonstrated progressive improvement compared to the previous approaches, with the highest accuracy of 94.04% achieved by the CNN-BiLSTM-Attention + Random Forest model. For transfer learning approaches, ProtBERT representations combined with XGBoost achieved an accuracy of 92%, while ESM representations combined with Random Forest outperformed them, reaching 94.84% accuracy. In addition, the ESM model after Fine-Tuning further improved the results, achieving the best performance in this study with an accuracy of 94.95%. Overall, these comparisons confirm the progressive improvement in protein sequence classification performance when moving from one approach to another, with transfer learning-based models showing superior performance, particularly the ESM Fine-Tuning model.

3.4.2 Interpretation of Results

The results showed that performance differences is directly related to the architectural design of the models and the tuning of hyperparameters, in addition to each model's ability to represent the biological properties embedded within protein sequences. These sequences are characterized by complex relationships between amino acids, making performance improvement highly dependent on how well each model can capture and represent these interactions.

The superiority of the Random Forest model among machine learning methods reflects the ability of ensemble decision trees to handle nonlinear relationships within sequences, leading to improved classification stability, unlike Naive Bayes, which relies on the assumption of feature independence, an assumption that does not align with the interconnected nature of amino acids in protein sequences.

In deep learning models, CNN achieved better performance due to its ability to extract

local motifs that represent important biological signals related to protein function. The Enhanced Transformer Encoder also demonstrated the effectiveness of attention in modeling long-range dependencies within protein sequences, in addition to improving the encoder architecture to enhance contextual interactions between amino acids. BiLSTM and BiGRU models were able to capture long-term sequential dependencies, enabling better contextual understanding.
Hybrid models benefited from the combination of different architectures, particularly through the Attention mechanism, which helped focus on the most biologically relevant regions within protein sequences, especially those related to functional roles.
In ensemble learning, combining deep hybrid models with machine learning algorithms such as Random Forest and XGBoost led to improved performance compared to standalone deep hybrid models, by effectively leveraging extracted features to enhance classification accuracy.
Finally, in transfer learning, protein language models showed superior performance due to their ability to learn deep biological representations. ProtBERT provided contextual representations of protein sequences, which were effectively utilized by the XGBoost model to improve the classification process through its ability to handle such features. ESM outperformed ProtBERT due to differences in the representations learned by each model, as ESM was trained with a more protein-specific objective, enabling it to better capture complex amino acid relationships, evolutionary patterns, and structural dependencies, which explains its superior performance in this study.

3.4.3 Impact of Feature Representation

The experimental results of this study showed that the method used to represent protein sequences plays a fundamental role in protein sequence classification performance. Different sequence representations provided different levels of biological information to the evaluated models. The corresponding Figure 3.9 compares feature representation methods for the best proposed model.

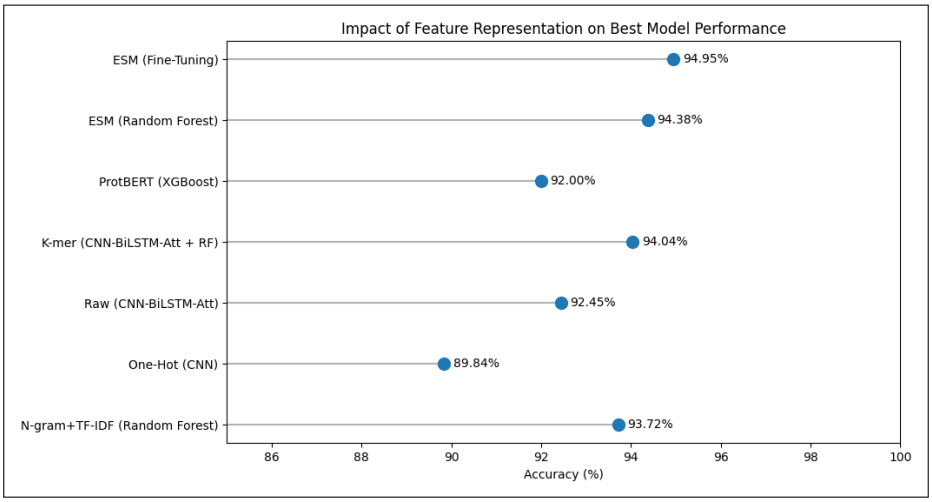


Figure 3.9 : Comparison of the Best Models Results Across Different Representation Methods.

The N-gram representation combined with TF-IDF demonstrated that frequency-based sequence patterns can still provide useful discriminatory information.

In addition, One-Hot Encoding showed the ability to convert protein sequences into a simple numerical representation that preserves the identity of amino acids directly, allowing models to treat each amino acid as an independent unit within the sequence.
In contrast, the K-mer representation with Integer Encoding helped preserve part of the sequential information, allowing models to detect recurrent local patterns within protein sequences. This positively influenced model performance, particularly hybrid and ensemble models that benefited from these patterns to extract features more closely related to protein function. Meanwhile, the Raw Sequences representation preserved the complete biological information of protein sequences in their original form, enabling models to learn representations directly from the data. However, its effectiveness remained dependent on the ability of the model architecture to learn meaningful features from the available data.
In addition, ProtBERT and ESM representations provided a higher level of feature representation by leveraging pretrained protein language models trained on large protein datasets, contributing to the generation of rich biological features that reflect the biological context of amino acids. Furthermore, ESM representations captured deeper biological and structural information, in addition to evolutionary relationships within protein sequences.
Finally, the use of ESM with Fine-Tuning achieved the best overall results by adapting the internal representations of the pretrained model according to the characteristics of the target dataset, enabling the extraction of more specialized and task relevant features for classification.
Overall, the transition from traditional representations to representations derived from protein language models led to a progressive improvement in protein classification performance, confirming the importance of protein sequence representation in building effective classification models.

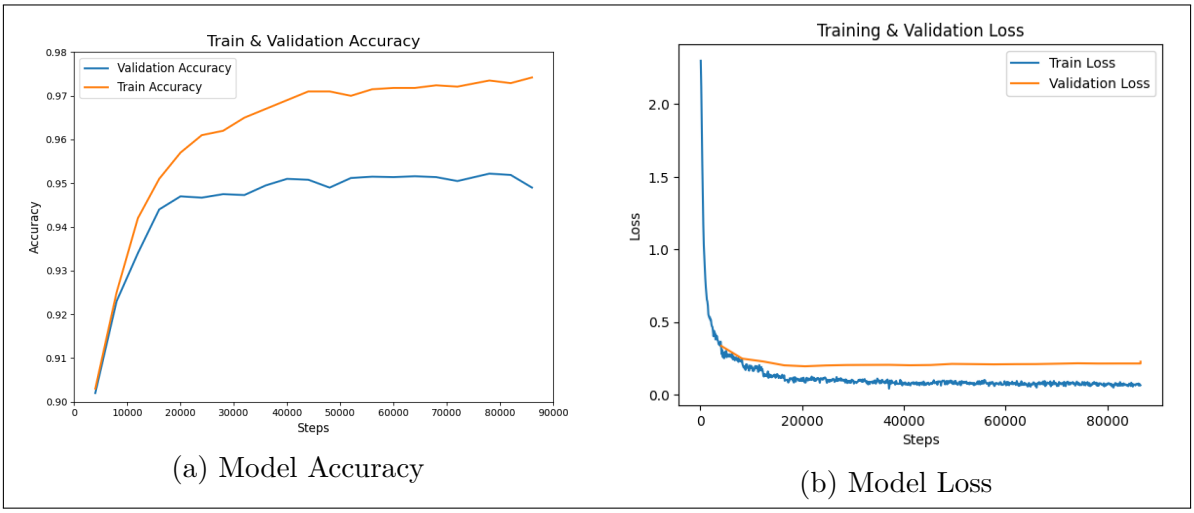
3.4.4 Overfitting Analysis and Generalization

This section aims to analyze the overfitting phenomenon and the generalization ability of the proposed ESM fine-tuning model in order to evaluate its performance on previously unseen data.

3.4.4.1 Overfitting Analysis

The training and validation curves presented in Figure 3.10 showed that the training dynamics of the ESM Fine-Tuning model exhibited stable learning behavior throughout the optimization process. The model effectively learned the underlying patterns in the data during the early stages of training, as both training and validation accuracy increased simultaneously while the loss values decreased significantly, confirming the model's ability to capture important features associated with biological functions. As training progressed, the training accuracy continued to increase gradually, reaching approximately 97.4%, while the validation accuracy stabilized around 95.21% with only minor fluctuations. Meanwhile, the training loss continued to decrease, whereas the validation loss stabilized at approximately 0.20.
These results indicate the presence of a small gap between training and validation performance, reflecting the model's ability to learn additional details from the training data, particularly during the later stages of training. However, this difference remained limited,

as the stability of the validation accuracy and the absence of any significant decline, together with the stable validation loss without a continuous increase, indicating that the model did not suffer from severe overfitting.



(a) Model Accuracy

(b) Model Loss

Figure 3.10 : Training and Validation Curves of the Fine-Tuned ESM Model Showing Accuracy and Loss Evolution.

3.4.4.2 Cross Validation Results

The corresponding Table 3.10 presents the test evaluation results of the Fine-Tuned ESM model using cross-validation.

Table 3.10 : Cross-Validation Results of the Fine-Tuned ESM Model.

Fold	**Loss**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-score (%)**
Fold 1	0.18	94.82	94.82	94.82	94.82
Fold 2	0.18	94.70	94.68	94.70	94.69
Fold 3	0.19	94.72	94.70	94.72	94.71
Fold 4	0.16	94.91	94.88	94.91	94.89
Fold 5	0.17	94.95	94.93	94.95	94.94
Average	0.17	94.82	94.80	94.82	94.81

The results obtained through Cross-Validation showed that the ESM Fine-Tuning model achieved stable performance across different folds, with classification accuracy ranging from 94.70% to 94.95%, and an average accuracy of 94.82%. Furthermore, the Precision, Recall, and F1-score values remained highly consistent across all folds, indicating a balanced classification performance without any noticeable bias. On the other hand, the loss values remained low and stable across all folds, with an average value of 0.18, confirming the model's effectiveness in minimizing classification errors. Moreover, the small variations observed between fold results indicate that the model did not depend on a specific data split but rather maintained consistent performance throughout all experiments. Overall, these results confirm the ability of the ESM model to generalize and maintain stable performance on unseen data that were not used during training.

3.4.4.3 External Dataset Evaluation

The corresponding Table 3.11 presents the evaluation results of the fine-tuned ESM model on external test data.

Table 3.11 : Evaluation Results of the Fine-Tuned ESM Model on External Test Data.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
ESM	96.30	96.00	96.00	96.00

Through the evaluation results obtained on the external dataset, it is evident that the ESM Fine-Tuning model achieved stable and balanced performance across different evaluation metrics. The model reached an accuracy of 96.30%, while Precision, Recall, and F1-Score each achieved 96.00%, indicating the model's strong ability to correctly classify samples. In addition, the model demonstrated the ability to extract rich biological features from protein sequences, which contributed to improving classification accuracy and enhancing generalization performance.

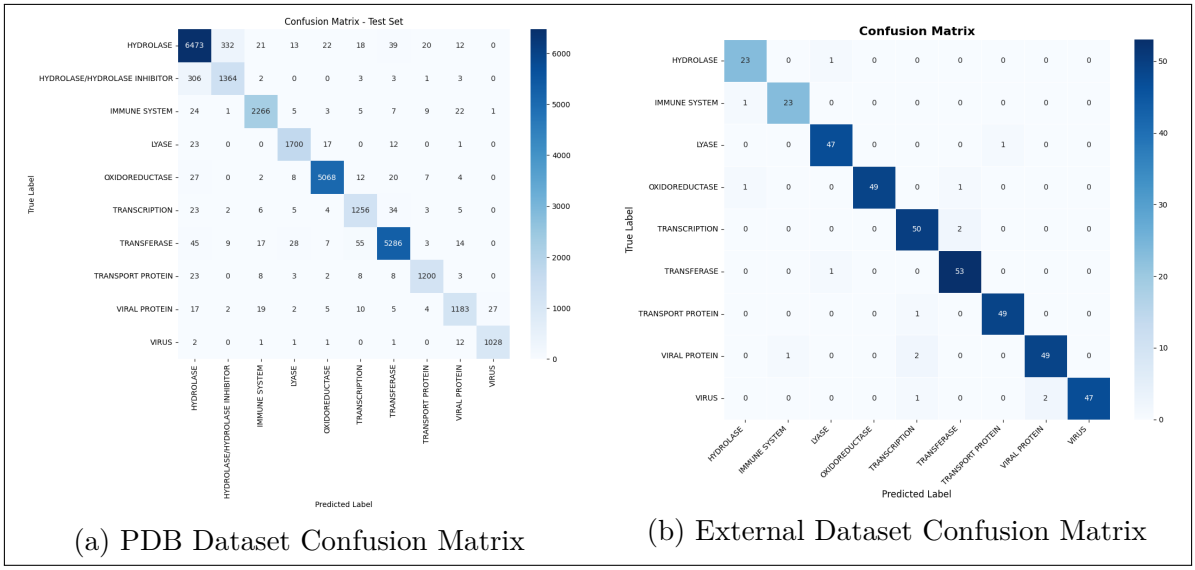
3.4.4.4 Generalization Discussion

The obtained results demonstrate the ability of the ESM model to generalize across different evaluation stages. The training and validation results showed no clear indications of overfitting, while the Cross-Validation results confirmed the stability and consistency of the model's performance. Furthermore, the evaluation on an external dataset demonstrated the model's ability to effectively handle previously unseen protein sequences without losing its classification capability. The model's generalization effectiveness can be attributed to its ability to learn rich biological representations from protein sequences, capturing the complex functional and structural relationships among amino acids. In addition, the Fine-Tuning process contributed to adapting these representations to the characteristics of the target dataset, enabling the model to acquire more task specific features relevant to the classification task.

3.4.5 Error Analysis

The confusion matrix of the ESM Fine-Tuning model presented in Figure 3.11(a) showed that most samples were correctly classified, as reflected by the high values along the main diagonal of the matrix. Most classes, including OXIDOREDUCTASE, TRANSFERASE, and IMMUNE SYSTEM, achieved a large number of correct predictions, confirming that the model successfully learned discriminative patterns that enabled efficient separation of functional protein classes. However, some misclassification cases were still observed. The HYDROLASE/HYDROLASE INHIBITOR class exhibited the weakest performance compared to the other classes. This can be attributed to the composite nature of this class, which contains sequences belonging to both HYDROLASE proteins and HYDROLASE INHIBITORS, while the dataset simultaneously includes a separate class dedicated to HYDROLASE proteins. Consequently, this overlap and similarity between the two classes increased the difficulty of distinguishing them and led to a higher level of confusion. The confusion matrix also revealed a limited number of misclassifications between other classes, such as TRANSFERASE and TRANSCRIPTION, which may be explained by shared biological patterns or structural and functional similarities between these categories.

Furthermore, the confusion matrix of the external dataset in Figure 3.11(b) demonstrated the model's ability to effectively distinguish between classes, particularly in the absence of the composite HYDROLASE/HYDROLASE INHIBITOR class. This contributed to reducing class confusion and improving classification performance. These findings suggest that most of the errors observed in the test set were primarily related to the nature of certain classes and the overlap between them, rather than limitations in the model itself. Overall, the results confirmed the effectiveness of the ESM Fine-Tuning model in distinguishing between different functional protein classes, while highlighting some challenges associated with the biological complexity of the data, which had only a limited impact on the models overall performance.



(a) PDB Dataset Confusion Matrix

(b) External Dataset Confusion Matrix

Figure 3.11 : Confusion Matrices of the Fine-Tuned ESM Model on PDB and External Datasets.

3.4.6 Impact of Dataset Composition

Through error analysis, it was observed that the composition of the dataset has a direct impact on the performance of the evaluated models, particularly regarding the presence of biologically related or composite classes. This was clearly illustrated by the HYDROLASE/HYDROLASE INHIBITOR class, which combines hydrolase proteins and their inhibitors within a single category, while a separate independent class exists specifically for hydrolase proteins. This overlap led to a high level of confusion between the two classes, making it difficult for the models to distinguish between them.

To further investigate the impact of this class on model performance, additional experiments were conducted by removing the HYDROLASE/HYDROLASE INHIBITOR class and replacing it with the SIGNALING PROTEIN class, which contains 6,469 protein sequences. Due to the high computational cost and long training time required for both the ESM Fine-Tuning model and the other models, these experiments were performed using the CNN model and the CNN-BiLSTM-Attention model.

The obtained results showed a noticeable improvement in performance after replacing the composite class. The accuracy of the CNN model increased to 94.92%, compared to 93.11% previously, while the CNN-BiLSTM-Attention model achieved an accuracy of 95.26%, compared to 93.25% previously. As shown by the confusion matrix of the

CNN-BiLSTM-Attention model in c, the classification performance is robust, with only rare errors in distinguishing between similar classes. These findings confirm that strong similarity and overlap between protein classes can increase the difficulty of learning discriminative sequence patterns and directly affect the model performance. Therefore, dataset composition is a critical factor in protein sequence classification, significantly influencing both model generalization and stability.

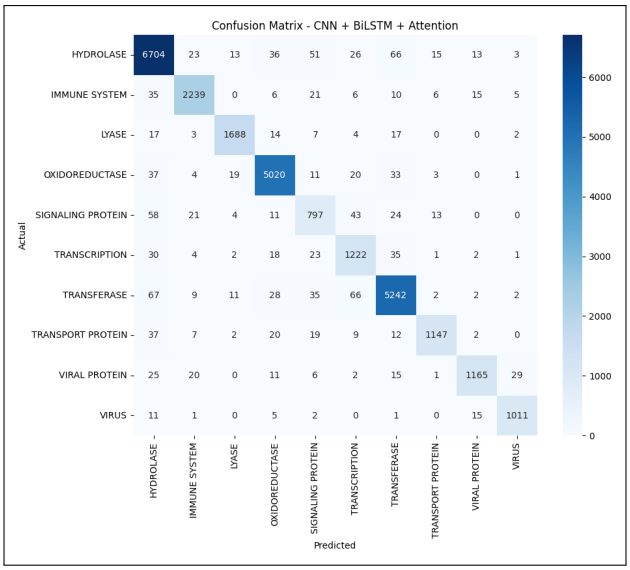


Figure 3.12 : Confusion Matrices of the CNN-BiLSTM-Attention Model.

3.4.7 Model Complexity and Computational Cost

The complexity of the models varied significantly depending on their architectural design. Machine learning models were considered the simplest due to their reliance on predefined features and representations, whereas deep learning models introduced higher complexity through the use of multiple neural layers such as convolutional and recurrent layers. Hybrid models further increased this complexity by combining several deep learning architectures, leading to deeper and more complex structures. Similarly, ensemble models increased complexity by integrating hybrid models with machine learning models, enhancing the models representational power and computational depth. Regarding transfer learning models, they represented the most complex category in this study due to their reliance on transformer architectures, particularly the self-attention mechanism, as well as their large number of parameters, making them the most advanced in terms of structure and resource consumption.

From a computational perspective, machine learning models required moderate training time and did not require significant GPU acceleration, maintaining relatively low computational cost even when handling large datasets such as the one used in this study (over 188,000 protein sequences). In contrast, deep learning models required moderate computational resources with the need for GPU acceleration due to matrix operations and backpropagation, and they required considerable training time, particularly the Enhanced Transformer Encoder model. Hybrid models exhibited higher computational cost with increased memory usage and training time, especially those incorporating attention mechanisms. Ensemble models were the most computationally expensive in terms of memory usage and training time, due to the integration of hybrid models with machine learning

models, significantly increasing computational complexity. Regarding protein language models, ProtBERT required higher computational resources and GPU memory, resulting in a long feature extraction time using batch processing. In contrast, feature extraction using the ESM model consumed more resources but was faster than ProtBERT. Meanwhile, ESM fine-tuning was the most GPU-intensive and required a very long training time, despite achieving the best performance.
Overall, the results demonstrate a positive correlation between model complexity, computational cost, and performance effectiveness. As model complexity and resource requirements increase, performance also improves, as clearly demonstrated by the ESM fine-tuning model.

3.4.8 Comparison with Related Work

The comparison with previous studies in Table 3.12 showed that the results obtained in this study were competitive with various approaches proposed in the literature. The use of models similar to those reported in previous studies, including machine learning and deep learning models, achieved higher performance in many cases due to the selection of appropriate feature representation methods, architectural improvements, and careful hyperparameter tuning, all of which contributed to enhancing classification performance.

The results also demonstrated that adopting more advanced approaches that were not employed in most previous studies, such as Enhance Transformer Encoder, hybrid models (CNN-BiGRU, CNN-BiLSTM-Attention, CNN-BiGRU-Attention), ensemble learning models (CNN-BiLSTM-Attention + Random Forest, CNN-BiLSTM-Attention +XGBoost, CNN-BiGRU-Attention+ Random Forest, CNN-BiGRU-Attention+ XGBoost) , and protein language models (ProtBERT, ESM), led to further improvements in performance. In particular, the ESM Fine-Tuning model achieved the highest accuracy of 94.95%, outperforming most previous studies based on machine learning, deep learning, and even some transformer-based approaches. This superiority can be attributed to the model's ability to learn rich biological representations that capture complex contextual and functional relationships within protein sequences, as well as the contribution of the fine-tuning process in adapting these representations to the target classification task, thereby enhancing the model's generalization capability.

The only exception among the compared studies was the work of Sekhar et al., in which an LSTM model achieved an accuracy of 96 % using the same PDB database. However, this difference should be interpreted in light of the different experimental settings. Their study relied on only 10,000 protein sequences, whereas the present study used a substantially larger dataset. Furthermore, the study did not provide details regarding the distribution of classes, which is considered an important factor influencing classification performance.

Table 3.12 : Comparison with Related Works.

Study	**Dataset**	**Data Size**	**Representation Technique**	**Best Model (Accuracy)**
Dongardive and Abraham [39]	Swiss-Prot	717	3-gram	KNN 84.51%
Vazhayil et al. [40]	Swiss-Prot	40,433	3-gram	LSTM 78.40%
Naveenkumar et al. [41]	Swiss-Prot	40,433	Keras Embedding	LSTM 91.24%
Siddha [42]	PDB	ND	TF-IDF +N-gram	Decision Tree 78.71 %
Sekhar et al. [43]	PDB	10,000	Count Vectorizer	LSTM 96.00%
Wang [44]	PDB	36,411	K-mer + Word2Vec	CNN 83.00%
Tasnim et al. [45]	PDB	226,693	TF-IDF + Count Vectorizer	SVM 92.00%
Perveen and Weeds [46]	PDB	ND	ProtBert Embeddings	ProtBert (Fine-Tuning) 77.00%
Chakraborty and Bhargava [47]	PDB	ND	5-mer + Integer Encoding	CNN 91.80%
This Study	**PDB**	**188,353**	**ESM-based Representation** Kmer+Integer Encoding	ETE 93.03% CNN-BiGRU-Attention 93.34% CNN-BiLSTM-Attention+RF 94.04% **ESM(Fine-Tuning) 94.95**%

Overall, this comparison confirms that the improvements proposed in the present study, whether at the level of sequence representation methods, architectural design, or the incorporation of additional models, particularly modern protein language models contributed to achieving advanced performance and placed the obtained results among the best reported outcomes in protein function classification studies.

3.4.9 Final Summary

Figure 3.13 summarizes the workflow of the proposed methodology developed in this study, which addressed the limitations identified in previous state-of-the-art approaches and improve protein functional classification performance.

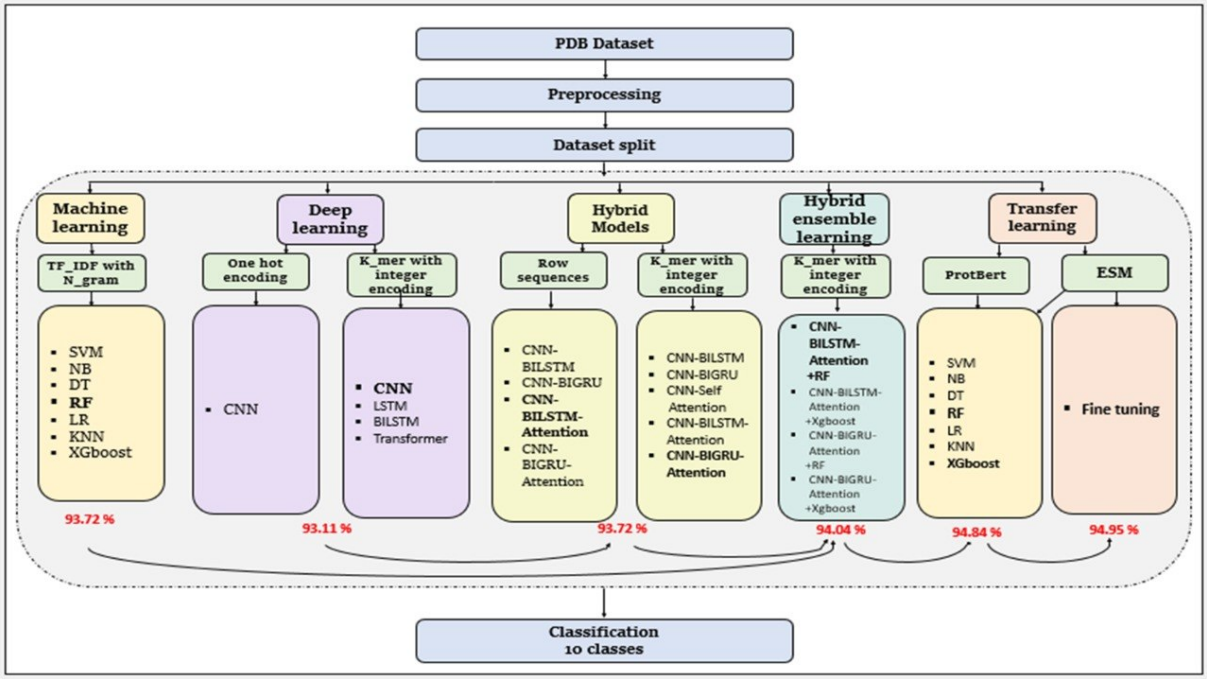


Figure 3.13 : Final Summary.

The overall results demonstrate a strong and consistent performance across all evaluated learning paradigms for protein functional classification. They demonstrate a progressive improvement in protein functional classification performance across different learning paradigms. Random Forest achieves the best performance among traditional Machine Learning models (93.72%), while CNN with K-mer representation effectively captures local sequence patterns (93.11%). The Hybrid model (CNN-BiGRU-Attention) further improves accuracy (93.34%) by combining feature extraction, sequential modeling, and attention mechanisms. Integrating this hybrid architecture with Random Forest in an Ensemble framework enhances robustness and achieves 93.70%. The highest performance is obtained by the fine-tuned ESM Transfer Learning model (94.95%), highlighting the effectiveness of pretrained protein language models in learning rich biological representations. Overall, the progression from ML to Transfer Learning leads to improved accuracy and better generalization, with ESM Fine-Tuning emerging as the most effective approach for protein functional classification. Moreover, the comparison with previous studies shows that the proposed methodology achieves competitive and often superior performance due to improved feature representations, optimized model architectures, and the integration of advanced learning strategies such as hybrid, ensemble, and transfer learning approaches.

3.5 Conclusion

This chapter presented the working environment and resources used to develop the different models, in addition to presenting and evaluating the obtained results, along with providing analyses and comparisons of model performance and studying the impact of sequence representation on classification performance.
The models' generalization ability was evaluated, along with error analysis and clarification of the impact of dataset composition on performance.
Moreover, the relationship between model complexity, computational cost, and achieved

performance was examined in this study, together with a comparison of the obtained results with previous works and a final summary of the different approaches and methods used.

General Conclusion

This study addressed the problem of protein function classification, which represents one of the major challenges in the field of bioinformatics. A diverse set of Machine Learning, Deep Learning, Hybrid, Ensemble, and Protein Language Models was developed. In addition, various protein sequence representation methods were adopted across different approaches. The results demonstrated the effectiveness of the proposed approaches in improving classification performance, particularly the ESM model, which achieved the best performance after fine-tuning technique, reaching an accuracy of 94.95%.

The results also showed that model performance was significantly influenced by the sequence representation method used. Features extracted using protein language models contributed to improved results compared with some traditional methods. Furthermore, the findings revealed variations in model performance depending on the adopted approach, highlighting the importance of protein sequence representation and feature extraction in improving classification systems.

The study also investigated the generalization capability of the models. The ESM model, after fine-tuning, demonstrated strong effectiveness in generalizing to both the target dataset and external datasets, although the results were influenced by the composition and quality of the data.

Although these results are promising, several limitations were encountered, including limited data availability, a restricted number of functional classes compared to the vast diversity of protein functions, and data imbalance. Moreover, many models, particularly protein language models, require high computational resources, which may limit their large-scale applicability.

In future work, this study can be extended by using larger and more diverse datasets, exploring a greater number of protein functional classes, and improving hybrid and ensemble models to enhance both accuracy and generalization. Additionally, more efficient and less resource-intensive protein language models can be explored. Furthermore, integrating structural data with sequence-based protein information could enrich biological representations and further improve classification accuracy.

Overall, this work confirms the effectiveness of artificial intelligence techniques in improving protein classification performance. Moreover, the integration of modern protein sequence representations with advanced deep learning techniques represents a promising approach for enhancing the accuracy and reliability of protein function classification systems, thereby supporting advancements in bioinformatics research and contributing to the development of future applications in the field.

Bibliography

[1] Sanvictores T, Farci F. Biochemistry, Primary Protein Structure. [Updated 2025 Dec 13]. In : StatPearls [Internet]. Treasure Island (FL) : StatPearls Publishing ; 2026 Jan-. https://www.ncbi.nlm.nih.gov/books/NBK564343/

[2] Jose, S. (2022). Protein importance and their function in the human body. J Biomol Res, 11, 194. https://www.walshmedicalmedia.com/open-access/protein-importance-and-their-function-in-the-human-body.pdf

[3] Clark, M. A., Douglas, M., & Choi, J. (2018). Biology 2e. https://openstax.org/books/biology-2e/pages/3-4-proteins

[4] Morris, R., Black, K. A., & Stollar, E. J. (2022). Uncovering protein function : from classification to complexes. Essays in biochemistry, 66(3), 255–285. https://doi.org/10.1042/EBC20200108

[5] Stollar, E. J., & Smith, D. P. (2020). Uncovering protein structure. Essays in biochemistry, 64(4), 649–680. https://doi.org/10.1042/EBC20190042

[6] Hamamsy, T., Barot, M., Morton, J. T., Steinegger, M., Bonneau, R., & Cho, K. (2023). Learning sequence, structure, and function representations of proteins with language models. bioRxiv : the preprint server for biology, 2023.11.26.568742. https://www.biorxiv.org/content/10.1101/2023.11.26.568742v1.full

[7] McDonald, A. G., Boyce, S., & Tipton, K. F. (2015). Enzyme classification and nomenclature. eLS, 1-11. https://gyansanchay.csjmu.ac.in/wp-content/uploads/2022/05/Classification-of-Enzymes_3.pdf

[8] Gaudet, P., Škunca, N., Hu, J. C., & Dessimoz, C. (2017). Primer on the Gene Ontology. Methods in molecular biology (Clifton, N.J.), 1446, 25–37. https://doi.org/10.1007/978-1-4939-3743-1_3

[9] The Gene Ontology knowledgebase in 2026. Nucleic Acids Research, 2026, vol. 54, no D1, p. D1779-D1792. https://doi.org/10.1093/nar/gkaf1292

[10] Korf, I., Yandell, M., & Bedell, J. (2003). Blast. " O'Reilly Media, Inc.".

[11] Ma, Y., Chen, H., Kang, J., Guo, X., Sun, C., Xu, J., Tao, J., Wei, S., Dong, Y., Tian, H., Lv, W., Jia, Z., Bi, S., Shang, Z., Zhang, C., Lv, H., Jiang, Y., & Zhang, M. (2025). The hidden Markov model and its applications in bioinformatics analysis. Genes & diseases, 13(1), 101729. https://doi.org/10.1016/j.gendis.2025.101729

[12] Jeffery, C. J. (2023). Current successes and remaining challenges in protein function prediction. Frontiers in Bioinformatics, 3, 1222182. https://doi.org/10.3389/fbinf.2023.1222182

[13] Zhang, H., Shi, Y., Wang, Y., Yang, X., Li, K., Im, S.-K., & Han, Y. (2025). Biological Sequence Representation Methods and Recent Advances : A Review. Biology, 14(9), 1137. https://doi.org/10.3390/biology14091137

[14] Lin, B., Luo, X., Liu, Y., & Jin, X. (2024). A comprehensive review and comparison of existing computational methods for protein function prediction. Briefings in Bioinformatics, 25(4), bbae289. https://doi.org/10.1093/bib/bbae289

[15] Kühl, N., Schemmer, M., Goutier, M. et al. Artificial intelligence and machine learning. Electron Markets 32, 2235–2244 (2022). https://doi.org/10.1007/s12525-022-00598-0

[16] Journal, I. R. J. E. T. IRJET- Machine Learning Algorithms : A Review. https://www.academia.edu/39652752/IRJET_Machine_Learning_Algorithms_A_Review

[17] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning : concepts, CNN architectures, challenges, applications, future directions. Journal of big data, 8(1), 53. https://doi.org/10.1186/s40537-021-00444-8

[18] Dastres, R., & Soori, M. (2021). Artificial neural network systems. International Journal of Imaging and Robotics (IJIR), 21(2), 13-25. https://hal.science/hal-03349542v1/document

[19] Vakalopoulou, M., Christodoulidis, S., Burgos, N., Colliot, O., & Lepetit, V. (2023). Deep learning : basics and convolutional neural networks (CNNs). Machine learning for brain disorders, 77-115. https://www.ncbi.nlm.nih.gov/books/NBK597497/

[20] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (2002). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324. https://ieeexplore.ieee.org/document/726791

[21] Abdel-Jaber, H., Devassy, D., Al Salam, A., Hidaytallah, L., & EL-Amir, M. (2022). A Review of Deep Learning Algorithms and Their Applications in Healthcare. Algorithms, 15(2), 71. https://doi.org/10.3390/a15020071

[22] Salman, D., Direkoglu, C., Kusaf, M. et al. Hybrid deep learning models for time series forecasting of solar power. Neural Comput & Applic 36, 9095–9112 (2024). https://doi.org/10.1007/s00521-024-09558-5

[23] Mishra, R. K., Reddy, G. S., & Pathak, H. (2021). The understanding of deep learning : A comprehensive review. Mathematical Problems in Engineering, 2021(1), 5548884. https://doi.org/10.1155/2021/5548884

[24] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

[25] Staudemeyer, R. C., & Morris, E. R. (2019). Understanding LSTM–a tutorial into long short-term memory recurrent neural networks. arXiv preprint arXiv :1909.09586. https://doi.org/10.48550/arXiv.1909.09586

[26] Cho, K., Van Merriënboer, B., Gulçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014, October). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1724-1734). https://arxiv.org/pdf/1406.1078

[27] Shiri, F. M., Perumal, T., Mustapha, N., & Mohamed, R. (2023). A comprehensive overview and comparative analysis on deep learning models : CNN, RNN, LSTM, GRU. arXiv preprint arXiv :2305.17473. https://doi.org/10.48550/arXiv.2305.17473

[28] Galassi, A., Lippi, M., & Torroni, P. (2020). Attention in natural language processing. IEEE transactions on neural networks and learning systems, 32(10), 4291-4308. https://doi.org/10.1109/TNNLS.2020.3019893

[29] National Center for Biotechnology Information https://www.ncbi.nlm.nih.gov/

[30] Rainio, O., Teuho, J., & Klén, R. (2024). Evaluation metrics and statistical tests for machine learning. Scientific reports, 14(1), 6086. https://www.nature.com/articles/s41598-024-56706-x

[31] Hannan, S. A., Pushparaj, M. J. K., Kumar, A., & Kaur, T. (2024). Detection of brain disorders using artificial neural networks. J AutonIntell, 7(5), 1-17. https://www.researchgate.net/publication/380604049_Detection_of_brain_disorders_using_artificial_neural_networks

[32] Das, S., Ahsan, S. M. M., Rahman, M., & Karim, M. S. (2024). A voting approach for heart sounds classification using discrete wavelet transform and CNN architecture. SN Computer Science, 5(2), 251. https://doi.org/10.1007/s42979-023-02580-9

[33] Hoa, P. V., Binh, N. A., Hong, P. V., An, N. N., Thao, G. T. P., Hanh, N. C., ... & Bui, D. T. (2024). One-dimensional deep learning driven geospatial analysis for flash flood susceptibility mapping : A case study in North Central Vietnam. Earth Science Informatics, 17(5), 4419-4440. https://doi.org/10.1007/s12145-024-01285-8

[34] Peng, K., & Yan, G. (2021). A survey on deep learning for financial risk prediction. Quantitative Finance and Economics, 5(4), 716. https://www.aimspress.com/article/doi/10.3934/QFE.2021032

[35] Naik, D., & Jaidhar, C. D. (2022). A novel Multi-Layer Attention Framework for visual description prediction using bidirectional LSTM. Journal of Big Data, 9(1), 104. https://doi.org/10.1186/s40537-022-00664-6

[36] Mohsen, S. (2023). Recognition of human activity using GRU deep learning algorithm. Multimedia Tools and Applications, 82(30), 47733-47749. https://doi.org/10.1007/s11042-023-15571-y

[37] Wang, X., Xie, G., Zhang, Y., Liu, H., Zhou, L., Liu, W., & Gao, Y. (2025). The Application of a BiGRU Model with Transformer-Based Error Correction in Deformation Prediction for Bridge SHM. Buildings, 15(4), 542. https://doi.org/10.3390/buildings15040542

[38] Ali, N., Neagu, D., & Trundle, P. (2019). Evaluation of k-nearest neighbour classifier performance for heterogeneous data sets. SN Applied Sciences, 1(12), 1559. https://doi.org/10.1007/s42452-019-1356-9

[39] Dongardive, J., & Abraham, S. (2015, December). Protein sequence classification based on N-gram and K-nearest neighbor algorithm. In Computational Intelligence in Data Mining—Volume 2: Proceedings of the International Conference on CIDM, 5-6 December 2015 (pp. 163-171). New Delhi : Springer India. https://link.springer.com/chapter/10.1007/978-81-322-2731-1_15

[40] Vazhayil, A., & KP, S. (2018). DeepProteomics : Protein family classification using shallow and deep networks. arXiv preprint arXiv :1809.04461. https://arxiv.org/abs/1809.04461

[41] Naveenkumar, K. S., Mohammed Harun, B. R., Vinayakumar, R., & Soman, K. P. (2018). Protein Family Classification using Deep Learning. bioRxiv, 414128. https://doi.org/10.1101/414128

[42] Siddha, S. S. (2020). Protein sequence classification using machine learning and deep learning (Doctoral dissertation, Dublin, National College of Ireland). https://norma.ncirl.ie/4472/1/shravaneeshekharsiddha.pdf

[43] Sekhar, S. M., Siddesh, G. M., Raj, M., & Manvi, S. S. (2021). Protein class prediction based on Count Vectorizer and long short term memory. International Journal of Information Technology, 13(1), 341-348. https://link.springer.com/article/10.1007/s41870-020-00528-3

[44] Wang, A. (2022). Deep learning methods for protein family classification on PDB sequencing data. arXiv preprint arXiv :2207.06678. https://arxiv.org/abs/2207.06678

[45] Tasnim, F., Habiba, S. U., Mahmud, T., Nahar, L., Hossain, M. S., & Andersson, K. (2024). Protein sequence classification through deep learning and encoding strategies. Procedia Computer Science, 238, 876-881. https://doi.org/10.1016/j.procs.2024.06.106

[46] Perveen, H., & Weeds, J. (2025). Protein sequence classification using natural language processing techniques. Discover Artificial Intelligence, 5(1), 66. https://link.springer.com/article/10.1007/s44163-025-00304-x

[47] Chakraborty, P., & Bhargava, A. (2025). XAI-Driven Deep Learning for Protein Sequence Functional Group Classification. arXiv preprint arXiv :2511.13791. https://arxiv.org/abs/2511.13791

[48] Zhao, Y., Wang, X., Meng, F., & Chen, X. (2026). AGConvLSTM : An Adaptive Graph Convolutional LSTM Network for Multi-Station Water Quality Classification. Water, 18(9), 1073. https://doi.org/10.3390/w18091073

[49] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019, June). Bert : Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics : human language technologies, volume 1 (long and short papers) (pp. 4171-4186). https://aclanthology.org/N19-1423/

[50] Uskaner Hepsağ, P., Özel, S. A., Dalcı, K., & Yazıcı, A. (2024). Using BERT models for breast cancer diagnosis from Turkish radiology reports. Language Resources and Evaluation, 58(3), 981-1012. https://doi.org/10.1007/s10579-023-09669-w

[51] Ahmad, S., Bano, N., & Raza, K. (2025). RCSB Protein Data Bank : revolutionising drug discovery and design for over five decades. Med. Data Min, 8(2), 7-11. https://www.tmrjournals.com/public/articlePDF/20250220/23eeff3cc1890ab75e547508f407cd54.pdf

[52] Structural protein sequences. https://www.kaggle.com/shahir/protein-data-set.

[53] Wang, X., Ding, Z., Wang, R., & Lin, X. (2023). Deepro-Glu : combination of convolutional neural network and Bi-LSTM models using ProtBert and handcrafted features to identify lysine glutarylation sites. Briefings in Bioinformatics, 24(2), bbac631. https://doi.org/10.1093/bib/bbac631

[54] Yang, Q., Yu, J., & Zheng, J. (2025). A survey of down stream applications of evolutionary scale modeling protein lan guage models. Quantitative biology (Beijing, China), 14(1), e70013. https://doi.org/10.1002/qub2.70013

[55] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30. https://doi.org/10.65215/nxvz2v36

[56] UniProt.https://www.uniprot.org/uniprotkb?query=reviewed:true

[57] RCSB Protein Data Bank. https://www.rcsb.org/

[58] Elnaggar, A., Heinzinger, M., Dallago, C., Rehawi, G., Wang, Y., Jones, L., ... & Rost, B. (2021). ProtTrans : toward understanding the language of life through self-supervised learning. IEEE transactions on pattern analysis and machine intelligence, 44(10), 7112-7127. https://doi.org/10.1109/TPAMI.2021.3095381

[59] Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297. https://link.springer.com/article/10.1007/Bf00994018

[60] Rish, I. (2001, August). An empirical study of the naive Bayes classifier. In IJCAI 2001 workshop on empirical methods in artificial intelligence (Vol. 3, No. 22, pp. 41-46). https://www.researchgate.net/publication/228845263_An_Empirical_Study_of_the_Naive_Bayes_Classifier

[61] Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. IEEE transactions on information theory, 13(1), 21-27. https://doi.org/10.1109/TIT.1967.1053964

[62] Quinlan, J. R. (1986). Induction of decision trees. Machine learning, 1(1), 81-106. https://link.springer.com/article/10.1007/BF00116251

[63] Cox, D. R. (1958). The regression analysis of binary sequences. Journal of the Royal Statistical Society Series B : Statistical Methodology, 20(2), 215-232. https://doi.org/10.1111/j.2517-6161.1958.tb00292.x

[64] Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32. https://link.springer.com/article/10.1023/A:1010933404324

[65] Chen, T., & Guestrin, C. (2016, August). Xgboost : A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794). https://doi.org/10.1145/2939672.2939785

[66] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout : a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), 1929-1958.