Leveraging Machine Learning for the Analysis and Prediction of Influenza A Virus

Thesis submitted in accordance with the requirements of the University of Liverpool
for the degree of Doctor in Philosophy in *Artificial Intelligence* by

Yanhua Xu
March 2024

To my lovely family and friends.

Acknowledgements

I would first like to give special thanks to my first supervisor, Dominik Wojtczak, for the endless support and guidance during this academic journey. I sincerely appreciate your advice. I would also like to thank Neil French for contributing to my understanding of vaccination practices, pneumococci, and the spread of infectious diseases. Also, thanks to Roberto Vivancos for his contribution to the establishment of this project.

I would like to extend my sincere gratitude to the members of the review committee. Their expertise shone through in their advice and constructive feedback, which greatly improved the study. I am especially grateful to them for reviewing the manuscript and identifying the areas that need clarification on my original submission. Their dedication and expertise have truly impacted this work positively. I deeply value their time and expertise.

To my family, words fall short of expressing my gratitude. My heartfelt thanks go out to my parents, who have quietly supported my education and career endeavours while providing all the assistance they could offer. I also want to express my gratitude to my friends for their support and motivation, which have been crucial in helping me navigate through this journey. Although physical distance and circumstances separate us, their virtual presence, constant check-ins, and unwavering belief in me are a constant source of inspiration and motivation.

And finally, I would like to take this moment to thank all who have contributed to the efforts by coining in academia to provide all the technical assistance, early career encouragement and all the moral support in that sense. I could not have made it through this project without all of your help.

Abstract

Influenza, commonly known as flu, is a respiratory disease that poses a significant challenge to global public health due to its high prevalence and potential for serious health complications. The disease is caused by influenza viruses, among which influenza A viruses are of particular concern. These viruses are known for their rapid transmission, potential to cause severe health issues, and frequent mutations, which underscore the need for ongoing research and surveillance. A key aspect of managing influenza outbreaks includes understanding host origins, antigenic properties, and the ability of influenza A viruses to transmit between species, as this knowledge is critical in forecasting outbreaks and developing effective vaccines.

Traditional approaches, such as hemagglutination inhibition assays for antigenicity assessment and phylogenetic analysis to determine genetic relationships, host origins and subtypes, have been fundamental in understanding influenza viruses. These methods, while informative, often face limitations in terms of time, resources, and the ability to keep pace with the rapid evolutionary changes of viruses. To mitigate these limitations, this thesis uses advanced machine learning techniques to analyse critical protein sequence data from influenza A viruses, offering an alternative perspective for unravelling the complexities of influenza, and potentially opening new avenues for analysis without strict reliance on prior biological knowledge.

The core of the thesis is the application and refinement of predictive models to determine host origins, subtypes, and antigenic relationships of influenza A viruses. These models are evaluated comprehensively, considering factors such as the impact of incomplete sequences, performance across various host taxonomies and individual hosts, as well as the influence of reference databases on model performance. This evaluation illuminates the potential of machine learning to enhance our understanding of influenza A viruses in real-world scenarios, pointing out the ongoing importance of this research in public health.

Contents

List of Figures

List of Tables

Acronyms

Acc Accuracy.

AdaBoost Adaptive Boosting.

AI Artificial Intelligence.

ANN Artificial Neural Network.

AP Average Precision.

AUPRC Area Under Precision-Recall Curve.

AUROC Area Under the Receiver Operating Characteristic Curve.

BERT Bidirectional Encoder Representations from Transformers.

BiGRU Bidirectional Gated Recurrent Unit.

BiLSTM Bidirectional Long Short-Term Memory.

BLAST Basic Local Alignment Search Tool.

BLEU Bilingual Evaluation Understudy.

BO Bayesian Optimisation.

BV-BRC Bacterial and Viral Bioinformatics Resource Center.

CDC Centers for Disease Control and Prevention.

CI Confidence Interval.

CM Confusion Matrix.

CNN Convolutional Neural Network.

CV Cross-validation.

DL Deep Learning.

DNA Deoxyribonucleic Acid.

DPC Dipeptide Composition.

DT Decision Tree.

ESM Evolutionary Scale Modeling.

F_1 F_1 Score.

Flu Influenza.

FN False Negative.

FP False Positive.

FPR False Positive Rate.

GISAID Global Initiative on Sharing All Influenza Data.

GloVe Global Vectors for Word Representation.

GP Gaussian Process.

GPT Generative Pre-trained Transformer.

GRU Gated Recurrent Unit.

HA Hemagglutinin.

HI Hemagglutination Inhibition.

IAV Influenza A Virus.

IRD Influenza Research Database.

IVRs Influenza Vaccination Rates.

kNN k-Nearest Neighbors.

LOOCV Leave-One-Out Cross-Validation.

LR Logistic Regression.

LSTM Long Short-Term Memory.

M1 Matrix Protein 1.

M2 Matrix Protein 2.

MC-NN Multi-Channel Neural Network.

MCC Matthew's Correlation Coefficient.

ML Machine Learning.

MLP Multi-Layer Perceptron.

MSA Multiple Sequence Alignment.

NA Neuraminidase.

NATs Nucleic Acid-Based Tests.

NB Naïve Bayes.

NCBI National Center for Biotechnology Information.

NLP Natural Language Processing.

NN Neural Network.

NP Nucleoprotein.

NR NCBI's Non-Redundant Protein Sequence Database.

NS1 Non-Structural Protein 1.

NS2 Non-Structural Protein 2.

PA Polymerase Acidic.

PB1 Polymerase Basic 1.

PB2 Polymerase Basic 2.

PR Precision-Recall.

Pre Precision.

PSI-BLAST Position-Specific Iterative Basic Local Alignment Search Tool.

PSSM Position-Specific Scoring Matrix.

RBCs Red Blood Cells.

RBF Radial Basis Function.

Rec Recall.

ReLU Rectified Linear Unit.

RF Random Forest.

RNA Ribonucleic Acid.

RNN Recurrent Neural Network.

RNP Ribonucleoprotein.

ROC Receiver Operating Characteristic.

RUSBoost Random Undersampling Boosting.

Sias Sialic Acids.

SMOTE Synthetic Minority Oversampling Technique.

Spec Specificity.

SVM Support Vector Machine.

T5 Text-to-Text Transfer Transformer.

TN True Negative.

TP True Positive.

TPR True Positive Rate.

UniRef100 Universal Protein Resource Database with 100% sequence identity.

UniRef50 Universal Protein Resource Database with at least 50% sequence identity.

URI Upper Respiratory Infection.

VE Vaccine Effectiveness.

WE Word Embedding.

WHO World Health Organisation.

XGBoost Extreme Gradient Boosting.

Please note that certain abbreviations, such as DL (Deep Learning), ML (Machine Learning) and Flu (Influenza), are mentioned for clarity and standard recognition within the field. However, they will be used in their long form throughout this thesis, except in tables and figures where abbreviations may be used for brevity.

1 Introduction

The central concern of this thesis is influenza, a highly infectious disease that presents a global public health concern. The symptoms of influenza vary widely, with common signs such as cough, fever, sore throat, tiredness and body aches. In more severe cases, it may result in hospitalisation. On rare occasions, it can be fatal, especially in high-risk populations such as older adults, young children, and those with underlying chronic health conditions. Influenza is caused by the influenza viruses, which primarily infect the respiratory system, specifically the nose, throat, and lungs. There are four types of influenza viruses: A, B, C, and D, with type A being the most severe and responsible for epidemics and rare pandemics. Moreover, Influenza A Viruses (IAVs) can mutate quickly, requiring continuous vaccine adaptations to keep up with the changes. Therefore, the dynamism and adaptability of IAVs not only make them subjects of extensive research, but also present challenges that require innovative solutions.

The methodologies proposed in this thesis are inspired by the recent advancements in the field of machine learning. In simple terms, machine learning allows computers to "learn" from data, which holds promise in revealing complex patterns in large datasets. Thus, machine learning is widely used in various domains, including virology, as it can be used to extract meaningful information from large datasets to predict future trends, optimise the decision-making process, and tackle historically difficult-to-solve problems.

The development of machine learning offers the opportunity to view the world from a novel vantage point, one that is observed through the lens of data. This thesis takes advantage of machine learning, with a particular focus on their potential to analyse the complex nature of IAVs. The primary objective is to apply and further refine existing machine learning techniques, such as Convolutional Neural Network (CNN) and Transformer, to analyse and predict characteristics of IAVs. This

involves using machine learning to understand better virus-host origins, antigenic relationships, and subtypes, all of which are crucial for effective influenza surveillance and response strategies.

The data used in this thesis are protein sequence data, which can be gathered from public databases, and the meta-data does not contain private information. Analysing these protein data often involves the use of bioinformatics techniques. Thus, bioinformatics was applied in the first study mentioned in this thesis.

The first study centres on predicting the host origins of IAVs. This study uses two methods to encode protein sequences: a commonly used bioinformatics method known as Position-Specific Scoring Matrix (PSSM) and a popular Natural Language Processing (NLP) technique called word embedding. These encoded features are further processed using traditional machine learning and deep learning techniques, respectively. This study evaluates the performance of these different methods and examines how data quality affects modelling accuracy.

The subsequent study focuses on predicting hosts and antigenic types of IAVs using Multi-Channel Neural Network (MC-NN). It performs a detailed analysis of various machine learning models, with a specific emphasis on evaluating their performance across diverse datasets. This in-depth analysis is designed to enhance understanding of the models' adaptability and accuracy in various scenarios, including those involving datasets with incomplete sequences and inputs based on single sequences.

The research then shifts to comparing supervised and semi-supervised learning models in predicting antigenicity of IAVs. This study spans different ratios of labelled data, emphasising the influence of pre-trained model features on the models' performance. This study underscores the adaptability and effectiveness of machine learning in scenarios with varied data availability, particularly in relation to different influenza subtypes.

1.1 Motivation

This thesis is primarily motivated by the crucial interplay between healthcare disparities and the advanced achievements of machine learning, with a focus on managing influenza globally. Influenza remains a significant public health challenge, presenting varying degrees of severity worldwide. In countries like the UK, where healthcare systems are robust and vaccination programmes are extensive, the impact of the influenza is somewhat controlled. However, the story is quite different in many developing countries. These regions face challenges such as limited healthcare

infrastructure, limited access to vaccinations, and a general lack of public awareness, leading to more severe consequences during influenza outbreaks. For example, vaccination coverage for critical vaccines in wealthier nations often exceeds 90%, but this figure can fall below 50% in some low- and middle-income countries [1].

Furthermore, the World Health Organisation (WHO) estimates that vaccination prevents 2 - 3 million deaths annually, but this figure could be underestimated, indicating the profound impact of vaccines on global health management [1, 2]. The global Influenza Vaccination Rates (IVRs) also paints a concerning picture, with only 24.96% of the general population being vaccinated, which is significantly lower than that of specific groups, such as individuals with chronic diseases (41.65%), healthcare professionals (36.57%), and pregnant women (25.92%) [3]. This disparity becomes more evident when comparing high-income countries/regions to middle-income ones, such as the United States and India.

In the United States, during the 2020–21 influenza season, the Centers for Disease Control and Prevention (CDC) reported an influenza vaccination coverage of 58.6% among children aged 6 months to 17 years and 50.2% among adults aged 18 years and older. Overall, approximately half (52.1%) of the US population aged 6 months and older were vaccinated during this season [4]. In contrast, in India, influenza vaccinations are not commonly administered, primarily due to a lack of awareness, limited access, and high costs. The pandemic raised awareness and demand for influenza vaccines, but the cost of these vaccines, which ranged between 1,500 and 2,000 Indian rupees ($20 - $27), remains a hurdle for many, considering that the country's per capita income was $1,913 in 2020 [5, 6]. This disparity reflects the significant challenges in increasing IVRs in lower-income countries, where financial constraints and lack of infrastructure present major barriers to widespread immunisation.

Given the context that influenza vaccination coverage is low in low- and middle-income countries, primarily due to limited financial resources allocated toward vaccine development and distribution, it sets the stage for the significant role of machine learning. This disparity highlights the need for innovative approaches to predict and manage infectious diseases, such as influenza, in a cost-effective manner. Meep learning, with its shift from model-driven to data-driven approaches, offers promising solutions by allowing the analysis of large amounts of data to generate actionable insights.

The focus on predicting the host origins, subtypes, and antigenicity of Influenza A Viruses (IAVs) using protein sequence data is particularly relevant. Protein sequences are key to understanding the genetic makeup of viruses, offering insights

into their behaviour and evolution. Furthermore, the use of protein sequence data avoids privacy concerns that are often associated with clinical or demographic data. This aspect is particularly important for maintaining ethical research practices and public trust, especially in studies that span multiple countries and cultures. The non-invasive nature of collecting protein sequences, compared to some forms of clinical data, also eases the logistical and ethical challenges in research.

Understanding the origins of the hosts is crucial for early detection and containment of outbreaks, as different hosts can harbour viruses with varying potentials to cross species barriers. Predicting subtypes is vital for identifying strains that could potentially cause pandemics, helping health authorities to prepare and respond more effectively. Lastly, studying antigenicity is important due to the antigenic variations caused by mutations in the HA1 protein of IAVs. These variations can lead to viruses that escape antibodies generated by previous infections or vaccinations, making the prediction of these antigenic relationships essential for designing effective vaccines.

In this light, applying machine learning to predict these aspects of IAVs becomes a critical component of modern healthcare strategies. The interest in evaluating the performance of machine learning models under real-world conditions, such as with incomplete protein sequences, is driven by the reality that biological data is often imperfect. Incomplete sequences represent a significant challenge in bioinformatics and can lead to inaccurate predictions if not properly accounted for. By focusing on how machine learning models perform with incomplete data, the aim is to enhance their robustness and reliability. This is especially important in low-resource settings, where data might be more fragmented or difficult to obtain.

Therefore, this thesis aims to explore how machine learning models can be adapted and improved for real-world applications in influenza prediction and management. The ultimate goal is to contribute to the development of models that are not only theoretically sound, but also practically effective in diverse and challenging data environments.

1.2 Related Work

This thesis focuses on the application of machine learning to predict various aspects of IAVs, particularly focusing on host and subtype prediction. This section provides an overview of related work that has contributed to advancements in this field.

1.2.1 Encoding Techniques for Influenza Virus Sequence Representation

Traditional methods for detecting IAVs hosts and subtypes, such as Hemagglutination Inhibition (HI) assays and Nucleic Acid-Based Tests (NATs), are noted for being laborious and time-consuming [7, 8].

To save manpower and time, various machine learning and deep learning algorithms have been used in viral host prediction, particularly leveraging sequence data for model training. A pivotal prerequisite for training these machine learning models is not only the collection of high-quality sequence data, but also its conversion into numerical vectors. This step is essential because machine learning models inherently require numerical input to perform computations and extract patterns within the biological data.

Various methods have been implemented to convert biological sequences into numerical forms for computational analysis. Among the most straightforward of these are one-hot encoding or pre-defined binary encoding schemes [9–15]. Each amino acid within a sequence is translated into a binary vector in these methods. Specifically, one-hot encoding assigns a "1" to a position corresponding to a specific amino acid, and "0"s to all other positions. On the other hand, pre-defined binary encoding schemes assign a unique binary code to each amino acid. Another technique explored is k-mer frequency [12, 16], which calculates the occurrences of subsequences of length k within the biological sequences.

Furthermore, the ASCII-based encoding scheme [17] has been used to convert each nucleotide to its corresponding integer. For example, the ASCII codes for the nucleotides A, C, G, and T are assigned as 65, 67, 71, and 84, respectively. This method yields a format easily interpretable by computational models and offers a more data-dense alternative than one-hot encodings, making it particularly efficient for handling complex genomic data.

In addition to the aforementioned methods, the use of physicochemical properties to represent sequences numerically [18–26] has been a method of interest. This technique leverages the fundamental biological characteristics of proteins, thereby enhancing the capacity of machine learning models to discern patterns pertinent to biological functions and structures. By incorporating features such as hydrophobicity, charge, and molecular weight, the models can gain a deeper understanding of the data, which allows for a more complex understanding of protein functionality and interactions. Previous research has identified factors that strongly correlate with host prediction accuracy. These include van der Waals volume, polarisability, and

charge for Hemagglutinin (HA), as well as polarity for Neuraminidase (NA) [20].

Alongside this, the use of evolutionary information has proven beneficial in improving model performance, with recent studies [27] indicating that it can attain performance levels comparable to those of attention models. Therefore, physicochemical properties are a data representation that reflects the nature of biological processes.

However, using handcrafted feature sets or physicochemical features requires a feature selection process, which can be time-consuming and burdensome. Word2Vec [28, 29] is a neural network-driven approach for capturing semantic relationships between words. It can be further used to capture the context of protein sequences in a multidimensional vector space [30]. This method offers an alternative by automatically learning representations, which can encapsulate complex relationships in the sequence and inherently grasp underlying patterns that may correlate with physicochemical properties, without the explicit need for extensive feature engineering.

Expanding upon this, using pre-trained language models for feature extraction from sequences offers further innovation in the field [31]. These models are usually trained on large-scale biological sequence databases, enhance efficiency, and potentially provide more informative features for computational analysis.

1.2.2 Machine Learning for Influenza A Host and Subtype Prediction

Various supervised machine learning algorithms have been used in predicting IAV's hosts or subtypes. This includes, but is not limited to, Support Vector Machine (SVM) [13, 23, 30, 32], k-Nearest Neighbors (kNN) [19], Random Forest (RF) [10, 19, 20, 23, 33], Artificial Neural Network (ANN) [12], Decision Tree (DT) [13, 34–36], Naïve Bayes (NB) [20, 23], Convolutional Neural Network (CNN) [14, 17], and Long Short-Term Memory (LSTM) [11].

The application of RF combined with physicochemical features has sometimes shown superior performance in predicting hosts (avian, human, swine) compared to other methods like kNN, NB, SVM, and ANN [10, 19, 20]. Additionally, DT has been used for extracting meaningful association rules from each segment of viral protein, helping to clarify how different influenza genome segments contribute to the determination of host range [34].

Furthermore, Scarafoni et al. [14] suggest that CNN might be more suited for tasks that involve host tropism than RF and kNN. In a related development, Fabijańska and Grabowski developed a universal automatic virus subtyping tool,

VGDC [17], which uses a deep CNN. This tool can predict the most prevalent pure Influenza A subtypes. In their study, they performed a comparative analysis of its performance against other models such as CASTOR [37], COMET [38] and C-measure [39], discovering that specific models could not be feasibly compared. They also observed that the optimal depth of the CNN architecture varied with genome length. Specifically, shallower architectures proved sufficient for shorter genomes, such as IAVs. In these cases, employing just two convolutional/pooling layers pairs for feature extraction was adequate to achieve commendable classification outcomes. The additional convolutional layers did not significantly enhance accuracy but did result in increased training time.

Previous research has also attempted to transform biological sequences into image-like formats, which aligns with the initial design of CNN for image processing. For example, Ahsan and Ebrahimi [13] used deep CNN specifically for subtyping IAVs, converting protein sequences into binary image formats. Their research particularly focused on the H1, H3, H4, H5, H9, N1, N2, N6, and N8 subtypes. In a similar vein, Sara et al. [31] adopted an innovative approach by extracting features from a pre-trained model, TAPE [40], and then reshaping each vector to a 28×28 size to suit ProtConv, the proposed simplified Lenet-5 based CNN [41]. In addition to these applications, CNN has also been shown to improve the training efficiency of LSTM models [11].

1.2.3 Multi-channel Neural Networks

Regarding the use of multi-channel neural networks (MC-NNs), these have been applied in diverse fields, including but not limited to imaging processing [42, 43], relation extraction [44], relation recognition [45], emotion recognition [46, 47], entity alignment [48], sentiment classification [49], face detection [50], and haptic material classification [51].

A MC-NN typically refers to a type of Neural Network (NN) architecture that processes multiple types of input data simultaneously but through separate channels. For example, channels can handle text and images in a task that requires understanding both the content of a photograph and a corresponding description. The model can from multiple perspectives or modalities of input through this approach.

The design of MC-NNs varies depending on the data, time, and task specificity. In general, features learned by each channel are fused at some point before generating outputs, either at the early stage or later stage. Most prior research has combined

the features of each input channel following convolutional or LSTM layers to ensure that the neural network can learn distinct deep features from different inputs [44, 45, 47, 52–56]. Additionally, input features can be extracted using pre-trained models and merged after a down-sampling process [57]. Each channel may have the same layer structure or a completely different one to better accommodate the input type [51, 58, 59].

However, the application of MC-NNs in infectious diseases, particularly in the domain of influenza, remains relatively unexplored. In this context, this thesis introduces three distinct MC-NN architectures designed specifically for the simultaneous prediction of host origins and subtypes of IAVs. This approach contrasts with the more conventional method of training single task-specific models, offering a more integrated and potentially more insightful analysis for IAVs research.

1.2.4 Predicting Antigenicity of Influenza A Viruses Using Machine Learning

The prediction of IAVs's antigenicity using machine learning represents a significant stride towards improving influenza surveillance and vaccine design, given the virus's rapid antigenic evolution. Recent advancements in this domain highlight the versatility and efficacy of machine learning models in understanding and anticipating the antigenic properties of IAVs.

The introduction of the Context-Free Encoding Scheme (CFreeEnS) [22] provides a novel method for encoding protein sequences into a numeric matrix specifically designed for machine learning applications. This approach is free from subtype-specific selected features and has been proven to enhance the accuracy of predicting the antigenicity of IAVs.

Furthermore, advanced machine learning models have been employed to enhance the prediction of antigenic variations, such as the application of 2D CNN for forecasting antigenic variants in multiple influenza subtypes, including A/H1N1, A/H3N2, and A/H5N1 [60]. In addition, a sophisticated method combining CNN with Bidirectional Long Short-Term Memory (BiLSTM) networks [15], focusing on predicting antigenic variants of the A/H3N2 subtype. This method leverages the strengths of both CNN for feature extraction and BiLSTM for capturing temporal dependencies.

Further research has led to the development of specific models aimed at understanding antigenic relationships within these subtypes. For instance, the PREDAC-H1pdm model [61] focuses on post-2009 pandemic H1N1pdm viruses, along with

PREDAC-H1 [62], illustrating the utility of machine learning in tracking antigenic clusters and their evolution over time. In addition to the PREDAC-H1pdm model, other models have been specifically designed for predicting antigenic variants of the A/H1N1 virus, including the use of a stacking model [63]. This approach leverages three techniques for extracting features: based on residues, regional bands, and epitope regions, to evaluate its efficacy and stability. The findings revealed that this stacking model surpasses single prediction models, using various classifiers, including Logistic Regression (LR), SVM, NB, NN, and kNN.

Following advancements in models for A/H1N1, predictive models for the A/H3N2 subtype have been further developed. For example, the attribute network embedding technique [64] has been proposed for forecasting antigenic distances among influenza A/H3N2 virus strains, using protein sequence representations as node attributes and antigenic distances as edges. A joint RF method [65] has also been applied to predict A/H3N2 antigenicity.

Beyond subtype-specific models, there are also universal models designed to offer broad applicability across various IAVs' subtypes, such as Univ-Flu [26], which uses novel 3D structure-based descriptors combined with RF, and PREDAV-FluA [66], which was built on regional bands to predict antigenic variation across different HA subtypes.

1.3 Contributions of the Thesis and Outline

This thesis first applies machine learning to predict the host origins of IAVs and adopts the Position-Specific Scoring Matrix (PSSM) to extract informative evolutionary features from protein sequences, based on the characteristics of protein data. This method involves sequence alignment, a commonly used method in prior studies [12, 14, 20, 67, 68], which increases the computational cost and complicates the process of building the model. Thus, a sequence alignment-free method is also proposed, which uses word embedding techniques to enable the model to learn features from sequences autonomously. It was found that both methods exhibit comparable performance, thereby suggesting that sequence alignment is not a necessity. It also indicates that deep learning can learn the physicochemical features from data, reducing the need for manual feature extraction that previously used in [9–12, 12–26].

Most previous studies [10, 12, 14, 19, 20, 25, 30, 33, 34, 69] have focused on predicting host origin at a higher classification level (i.e., human, swine, and avian), primarily because their research focus is on host tropism or zoonotic risk prediction.

In contrast, this thesis predicts host origin not only at this higher classification level, but also at a lower classification level (i.e., subdividing the avian class further).

The thesis introduces novel architectures of end-to-end MC-NNs specifically designed for predicting hosts and antigenic types of IAVs. Furthermore, it conducts rigorous evaluation and validation of the proposed models under mimicked real-world scenarios to demonstrate the practical utility of these models.

1.3.1 Published Works

Most of the work presented in this thesis has been published in the following publications. For the most recent versions of these papers, please refer to the corresponding arXiv versions:

Conference Paper

1. Xu, Y. and Wojtczak, D., 2021. Predicting influenza A viral host using PSSM and word embeddings. In *2021 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)* p.1-10. IEEE [70]. arXiv:2201.01140. (Chapter 4)

2. Xu, Y. and Wojtczak, D., 2022. End-to-End Multi-channel Neural Networks for Predicting Influenza A Virus Hosts and Antigenic Types. In *Proceedings of the 14th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management - KDIR*, p.40-50 [71]. arXiv:2206.03823. (Chapter 5)

Journal Paper

1. Xu, Y. and Wojtczak, D., 2022. Dive into machine learning algorithms for influenza virus host prediction with hemagglutinin sequences. *Biosystems*, 220, p.104740 [72]. arXiv:2207.1384. (Chapter 4)

2. Xu, Y. and Wojtczak, D., 2023. Predicting Hosts and Antigenic Types of Influenza A Virus using End-to-End Multi-channel Neural Networks. *Springer Nature Computer Science*, 4(5), p.435 [73]. arXiv:2306.05587. (Chapter 5)

1.3.2 Structure of the Thesis

This thesis undertakes a comprehensive exploration of machine learning methodologies tailored for addressing challenges in IAVs' research, with a specific focus on

predicting host origins and understanding antigenic relationships. It begins with a foundational overview of influenza and machine learning concepts (Chapter 2), followed by an introduction to sequence data representation (Chapter 3). Subsequent chapters include an extensive investigation into predicting the host origins of the IAVs using bioinformatics and machine learning (Chapter 4) and an exploration of a MC-NNs approach for identifying hosts and antigenic types of IAVs (Chapter 5). The thesis also examines the role of semi-supervised learning in predicting IAVs's antigenicity (Chapter 6). The final chapter (Chapter 7) consolidates the main conclusions and discusses potential avenues for future research.

2

Preliminaries of Influenza and Machine Learning

In this chapter, a foundational understanding of both influenza and machine learning is established, providing the necessary background for the thesis's focus on the prediction of Influenza A Viruses (IAVs) using machine learning techniques. Some key algorithms and techniques used in the experiments throughout this thesis are also introduced to avoid redundancy in later sections.

The exploration begins with influenza, delving into its biological aspects and how it differs from similar viruses like the common cold. This includes a discussion on the key proteins that define influenza viruses, their evolutionary mechanisms, and the factors contributing to their pathogenicity. Understanding the host range of IAVs and the techniques like Hemagglutination Inhibition (HI) assay used in their study is crucial for grasping the complexity of predicting virus behaviour and spread.

Then, the chapter transitions into the fundamentals of machine learning, a field that has become instrumental in analysing complex biological data. This section elucidates what constitutes machine learning, how it differs from and relates to deep learning and the paradigms that underpin it. By outlining the typical workflow of machine learning projects, including various algorithms and techniques such as nested k-fold Cross-validation (CV) and Bayesian Optimisation (BO), the chapter sets the stage for understanding how these tools can be applied to the nuanced challenges of influenza research. The rationale behind using specific evaluation metrics to assess the performance of machine learning models is also discussed to highlight their importance in validating the effectiveness of these computational approaches.

2.1 Foundations of Influenza

2.1.1 Influenza, Influenza Viruses and the Common Cold

Difference between Influenza and the Common Cold

Influenza is a respiratory illness caused by influenza viruses. This viral infection triggers annual epidemics, affecting approximately 1 billion people worldwide each year [74]. These recurrent outbreaks are significant, leading to around half a million deaths worldwide annually. It is important to distinguish between the flu and common colds, as they are often mistakenly thought to be the same due to overlapping symptoms.

Influenza is not a (severe) cold. The common cold, primarily affecting the upper respiratory system, is often referred to by physicians as Upper Respiratory Infection (URI). A common misconception is that a sudden drop in temperature is a primary trigger for the onset of the common cold. However, the common cold actually arises from an imbalance between the body's immune defences and the normal viral flora present in areas such as the nasal cavity, rather than being directly caused by cold weather or a foreign pathogen invasion. While exposure to cold weather can influence susceptibility, common colds can occur at any time throughout the year, particularly when individuals are subjected to conditions that may weaken immune responses. In contrast, influenza is specifically caused by an influenza virus infection and is known for its seasonal outbreaks, often linked to specific strains of the virus circulating during a given period [75].

The development of vaccines for the common cold remains unfeasible for several reasons [75]. Firstly, the common cold typically results from an interaction between the body's immune defences and the endogenous viral population in our respiratory system, which generally does not elicit a novel immune response. Furthermore, the common cold can be caused by over 200 different viruses, presenting a significant challenge in developing a universal vaccine. Additionally, the cold is typically a self-limiting disease that usually resolves within a few days through the innate immune system's primary action, thereby reducing the need for adaptive immunity intervention.

Regarding its prevalence, adults may experience the common cold two to five times annually, while school-aged children are more susceptible, contracting it approximately seven to ten times each year [76, 77]. Although common colds are widespread and can cause discomfort, they are generally less severe than the influenza. It's important to note that antibiotics are not effective in treating either

the influenza or the common cold, as both are viral infections and antibiotics target bacterial pathogens.

Influenza Viruses

Influenza viruses belong to the *Orthomyxoviridae* family, which are major human and animal pathogens categorised into four serotypes: A, B, C, and D [78]. Specifically, influenza A, B, and C viruses are major human pathogens. Influenza B and C are known to infect only mammals. Influenza A Viruses (IAVs) are characterised by their single-stranded Ribonucleic Acid (RNA) genomes, which consist of eight segments of negative polarity [78], varying in length from approximately 890 to 2,341 nucleotides. The simplified representation of the genomic structure of influenza virus is presented in Fig. 2.1. These segments encode at least 11 proteins: segment 1 encodes for Polymerase Basic 2 (PB2), segment 2 for Polymerase Basic 1 (PB1) and PB1-F2, segment 3 for Polymerase Acidic (PA), segment 4 for Hemagglutinin (HA), segment 5 for Nucleoprotein (NP) segment 6 for Neuraminidase (NA), segment 7 for matrix proteins M1 and M2, and segment 8 for nonstructural proteins NS1 and NS2. Among these, the surface proteins HA and NA play crucial roles in virus attachment, and cell fusion, and are primary targets of the host immune system. The serotypes of IAVs are distinguished based on HA and NA subtypes, with 18 HA (named H1 to H18) and 11 NA (labelled N1 through N11) have been reported to date [79, 80]. Further details about the functions of these proteins are discussed in Section 2.1.2.

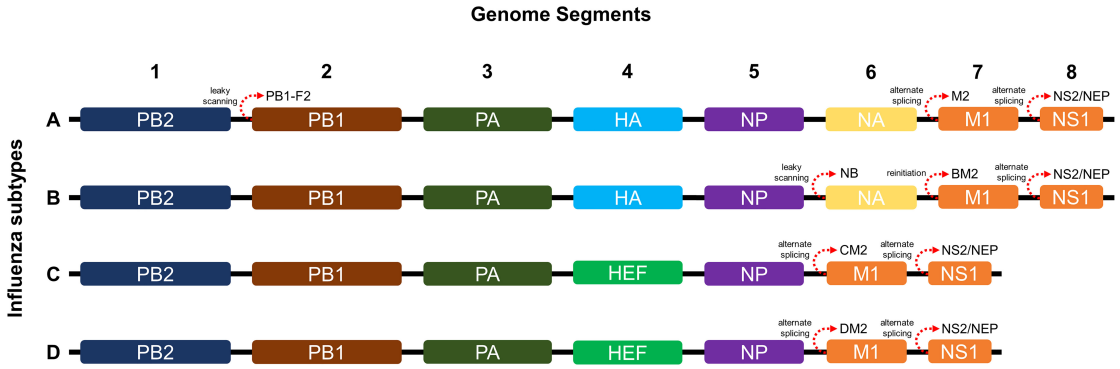


Figure 2.1: Schematic representation of influenza virus genome segments: the diagram simplifies the genomic structure for visual clarity, whereas in reality, the influenza genome is composed of single-stranded RNA segments that are not arranged in a linear fashion as illustrated. Image was sourced from [81, 82].

Table 2.1 presents an overview of the various HA and NA subtypes along with the species in which they have been detected, and Fig. 2.2 illustrates the

distribution of influenza virus subtypes A/H1, A/H3, A/H5, and Type B in the top 15 countries with the highest prevalence.

Table 2.1: Distribution of Hemagglutinin (H) and Neuraminidase (N) subtypes across various species [83].

Species Detected In	H/N Subtypes
Human, Poultry, Swine, Bat or Other Animals	H3
Human, Poultry, Swine	H1, H2, H5, H9; N1, N2
Swine, Bat or Other Animals	H4
Human, Poultry or Other Animals	H7; N7, N8
Human, Poultry	H6, H10; N6, N9
Poultry	H8, H11 - H16; N3 - N5
Bat	H17, H18; N10, N11

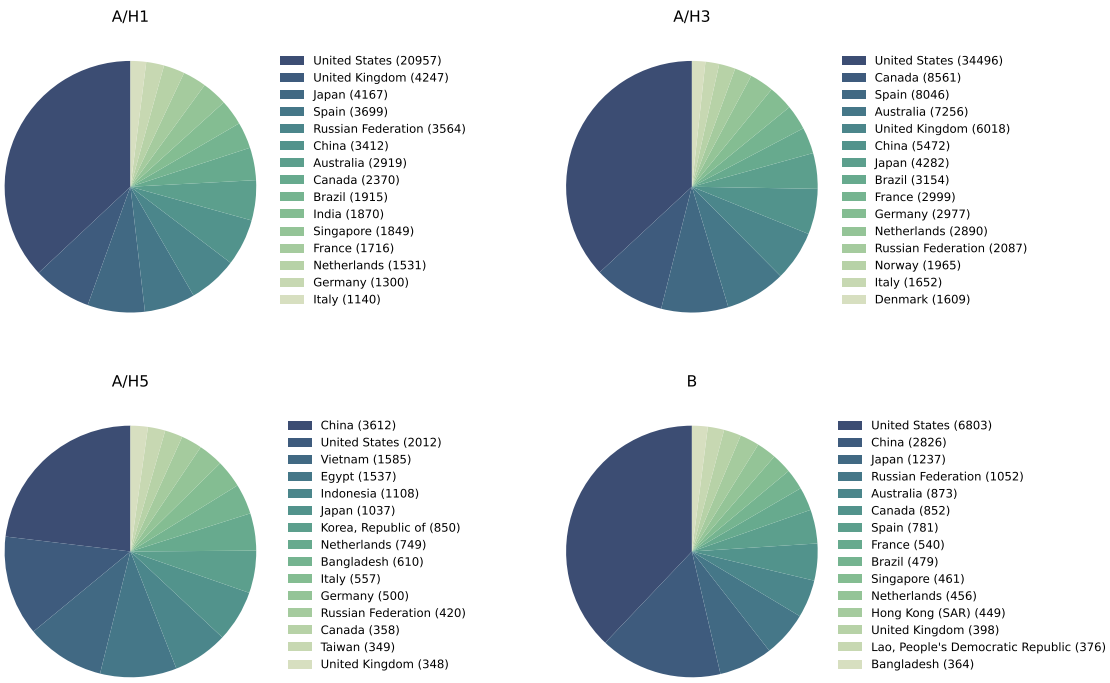


Figure 2.2: Distribution of influenza virus subtypes A/H1, A/H3, A/H5, and Type B in the top 15 countries with the highest prevalence, as of 25 March 2023 [84].

The persistence and spread of influenza viruses are greatly affected by environmental factors, with temperature playing a particularly significant role [85–87]. It has been observed that the influenza virus demonstrates greater stability at lower temperatures, indicating that temperature is a vital factor in its transmission and prevalence [88].

2.1.2 Key Proteins of Influenza Viruses

Influenza viruses, while structurally simple, exhibit complexity in their functional mechanisms, primarily due to the presence of several key proteins that play pivotal roles in the infection process. Among these, HA and NA are the most critical for the virus's life cycle. A brief introduction to these proteins will be outlined in this section.

HA is a major surface glycoprotein of influenza A and B viruses, it facilitates the virus's attachment to susceptible cells in the host's respiratory tract, marking one of the initial steps in the infection process, which enables the virus to enter the cell. Additionally, HA is crucial in mediating the fusion between viral and endosomal membranes, a process vital for the delivery of the virus's genome into the cell. This fusion mechanism has positioned influenza's HA as a paradigm for virus-cell fusion processes, offering a framework for understanding similar mechanisms in other enveloped viruses. Moreover, HA is a critical antigen for inducing protective antibody responses in the host, making it the primary target for the humoral immune response during influenza infection.

NA plays a complementary role to HA in the influenza virus lifecycle. While HA facilitates entry into the host cell, HA assists in releasing newly formed virus particles post-replication. NA's receptor-destroying activity, which recognises Sialic Acids (Sias), is essential for the virus's propagation within the host. The intricate balance in Sias recognition between HA and NA is crucial for a successful viral infection.

In addition to HA and NA, the influenza virus comprises other vital proteins. Matrix proteins (M1 and M2) contribute to the virus's structure and stability, with M2 playing a role in aiding the virus to enter the host cell. NP binds to the virus's RNA genome, providing protection and organisation. It serves as a structural and functional unit within the Ribonucleoprotein (RNP) complex.

The non-structural proteins (NS1 and NS2), play critical roles in evading the host's immune response. The polymerase proteins (PA, PB1, PB2) are important for replicating the virus's RNA genome, crucial for generating new virus particles inside the host cell. PB1 is responsible for the polymerase activity, PB2 serves as the cap-binding protein, and PA is associated with proteolytic activity.

The structure and lifecycle of the IAV are shown in Fig. 2.3 and Fig. 2.4. For a more comprehensive understanding of these key proteins and their roles in the influenza virus lifecycle and host interactions, please refer to [78, 89, 90].

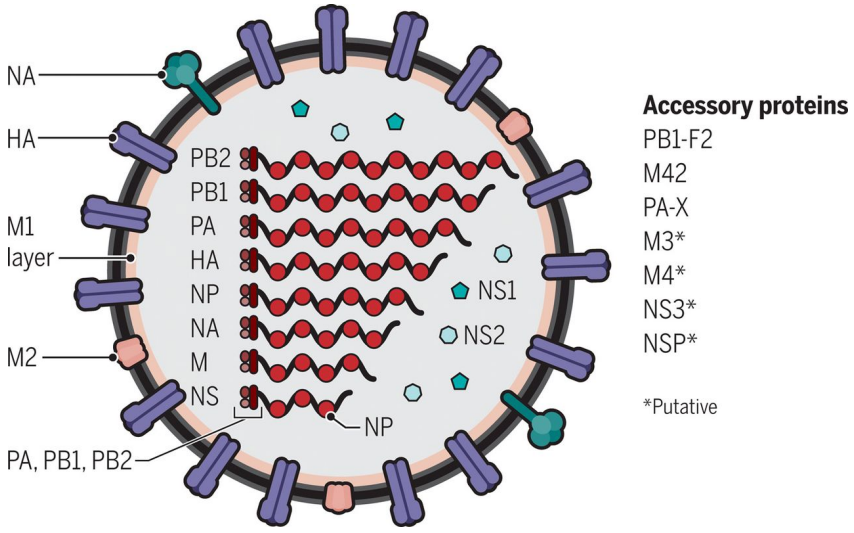


Figure 2.3: Structure of influenza A viruses, sourced from [91].

2.1.3 Evolutionary Mechanisms and Pathogenicity of Influenza Viruses

Influenza viruses are transmitted to hosts through respiratory droplets from infected individuals and exhibit significant virulence primarily due to two major surface glycoproteins: HA and NA. As mentioned in Section 2.1.2, HA facilitates the virus's attachment to respiratory epithelial cells. In addition, NA cleaves Sias to release new virions from host cells and reduces the mucous film's viscosity in the respiratory tract, aiding in the spread of the virus. This rapid viral spread leads to widespread infection and damage to the respiratory epithelium, rendering it vulnerable to secondary bacterial infections or superinfections.

HA and NA are not only vital for the virus's propagation but also the primary targets of the host immune response. To evade this immune detection, these proteins undergo constant evolutionary changes. Two primary evolutionary processes that influenza undergoes are *antigenic drift* and *antigenic shift*. Antigenic drift is characterised by the gradual accumulation of point mutations. These mutations give rise to distinct influenza strains (i.e., antigenic variants), which are responsible for annual epidemics. The newly evolved strains of the influenza virus exhibit sufficient genetic divergence from those of previous years, thereby evading the host's memory responses developed from prior infections or vaccinations. This is the primary reason for the necessity of frequent updates to influenza vaccines, to ensure they remain effective against these continuously evolving and circulating strains.

Antigenic shift, in contrast to antigenic drift, represents a more radical change, resulting in the introduction of entirely new subtypes of the virus into the human

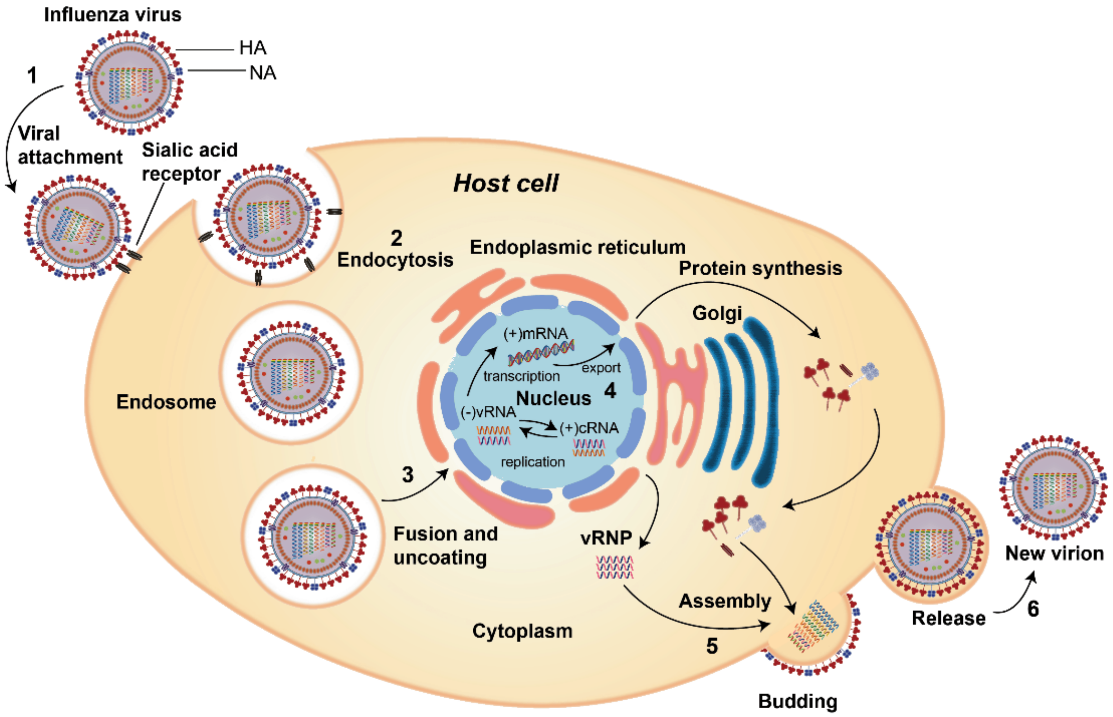


Figure 2.4: The influenza A virus lifecycle can be described through the following stages: `(1) viral attachment`: the virus uses HA to latch onto sialic acid on the surface of a host cell; `(2) endocytosis`: the virus is then taken into the host cell through a process known as endocytosis; `(3) fusion and uncoating`: inside the host cell, the virus merges with cell membranes and releases its genetic material; `(4) replication`: this genetic material is transported into the cell's nucleus, where it is copied and new viral proteins are made; `(5) assembly`: the new viral components are assembled into complete viruses at the cell's membrane. `(6) release`: these new viruses exit the host cell to infect more cells. The image was sourced from [90].

population, as shown in Fig. 2.5. The antigenic shift can occur through genetic reassortment between different influenza virus strains, potentially in a host infected with both human and non-human influenza strains. Due to the segmented nature of the influenza virus genome, this reassortment is readily facilitated. This mechanism is one of the contributors to infrequent but severe worldwide pandemics, as they introduce novel and dangerous strains against which the entire population lacks pre-existing immunity.

Antigenic drift and antigenic shift pose significant challenges for public health, particularly in vaccine development and influenza control strategies. Continuous surveillance and research are essential to understand and mitigate the impacts of these changes, especially considering the heightened risk groups, such as pregnato whichomen, who face more significant complications from influenza due to immune system adaptations during pregnancy. These adaptations can weaken the maternal

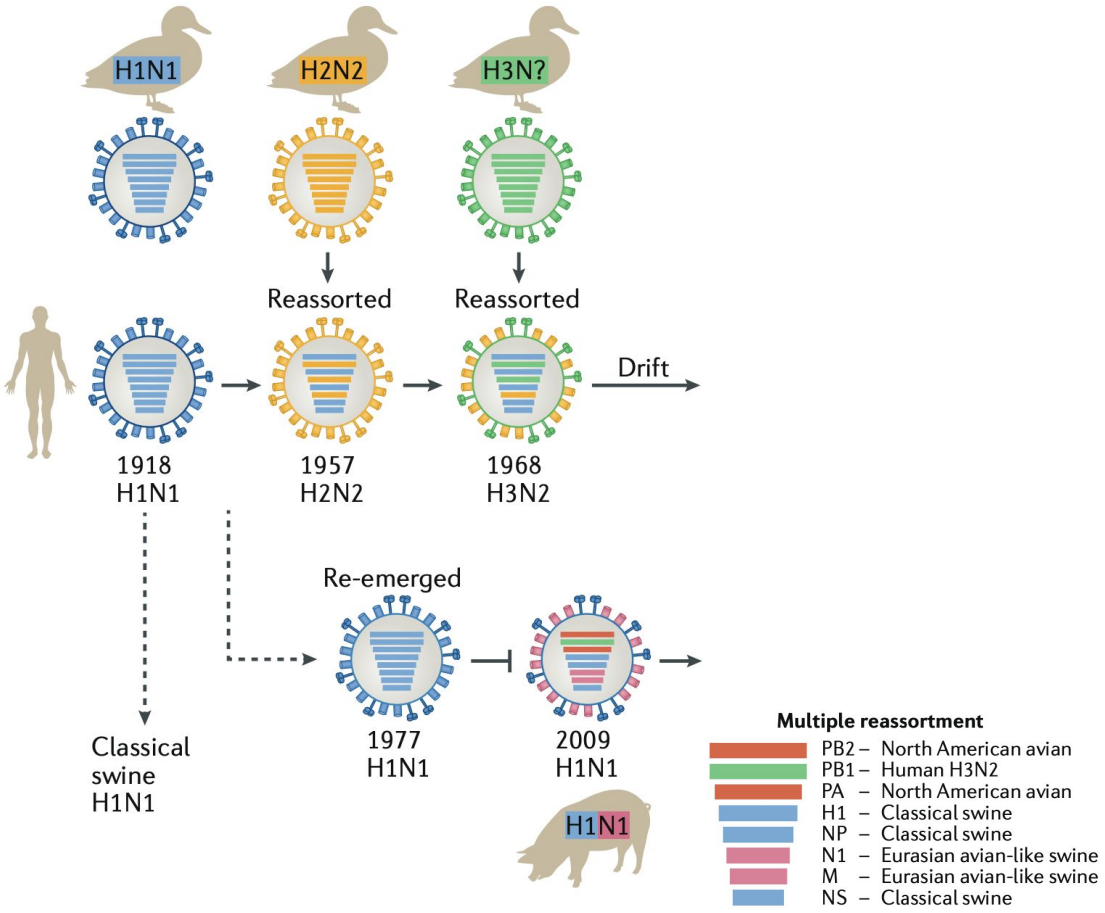


Figure 2.5: Genetic reassortment in Influenza A Virus: Antigenic drift is the minor variations in the surface proteins of the influenza virus, resulting from errors made by the virus's polymerase, allowing the virus to evade immune responses accumulated within the population. Antigenic shift, on the other hand, involves larger-scale genetic reassortment, leading to the acquisition of new genetic material by the virus, which can trigger influenza pandemics. Pandemic viruses typically emerge following antigenic shift, carrying entirely new combinations of antigens that the population lacks immunity, thus enabling rapid spread. Once a pandemic subsides, these viruses often persist within the population as seasonal influenza, undergoing antigenic drift and gradually adapting to human hosts. The image was sourced from [92].

response to infections like influenza, underlining the need for targeted prevention and treatment strategies in such vulnerable populations.

For a more comprehensive understanding of the evolutionary mechanisms and pathogenicity of influenza viruses, please refer to [78, 90, 92, 93].

2.1.4 Host Range of Influenza A Viruses

Influenza viruses use multivalent interactions of their HA with sialyl oligosaccharide moieties on cellular glycoconjugates to attach to target cells. This interaction, crucial in determining the virus's host range and tissue tropism, is mediated by sialic acids, which are commonly found on the surfaces of most avian and mammalian cells. As a result, influenza viruses can bind to a variety of cell types, leading to significant biological responses, including the stimulation of inflammatory responses. The host range of influenza viruses encompasses a variety of species, ranging from humans to various animal groups such as avian, swine, equine, and canine [78], as shown in Fig. 2.6.

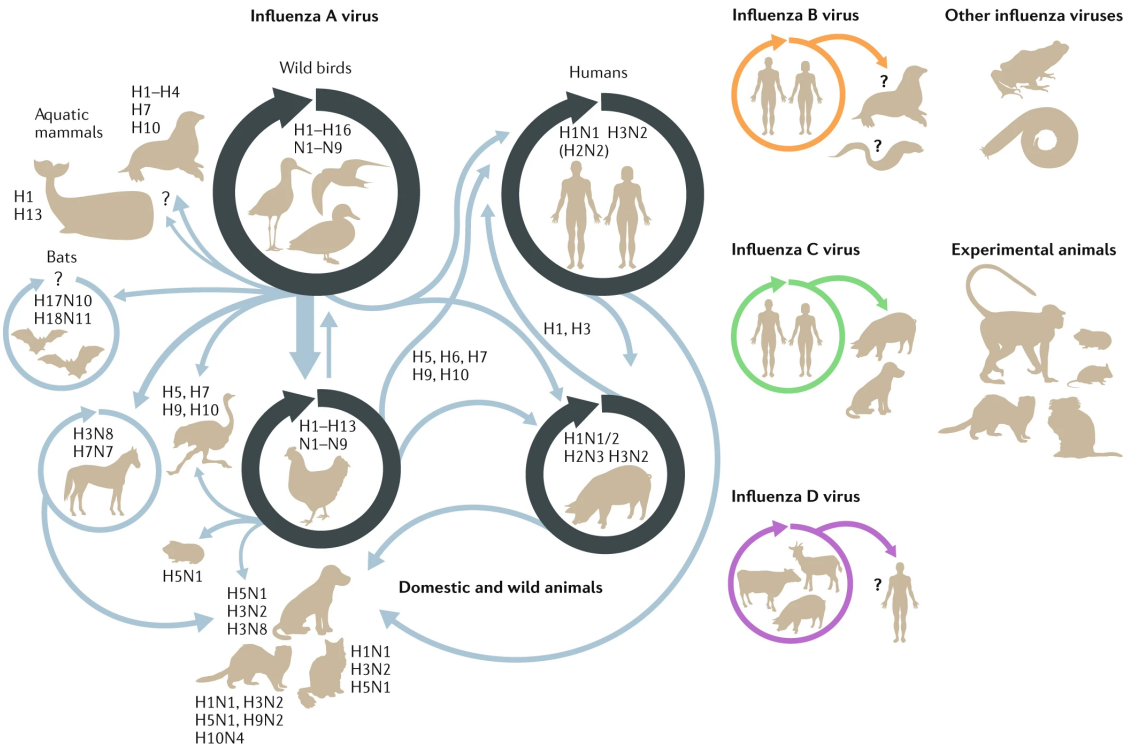


Figure 2.6: Host range of influenza viruses: an overview of the transmission pathways and subtypes of influenza A, B, C, and D viruses among humans, various animals, and birds. The image was sourced from [92].

Influenza A Viruses (IAVs) demonstrate the broadest host range among influenza viruses, distinguished by their capacity to infect many species. This attribute contributes to the widespread occurrence of both endemic and epidemic infections across diverse populations. Wild aquatic birds are the primary natural reservoirs for IAVs. These viruses sporadically transmit to a diverse range of species, such as sea mammals, swine, land-based poultry, horses, and humans, leading to infections of varying severity. Typically, following transmission to a new host species, these viruses

tend to die out due to inadequate adaptation. However, on rare occasions, they can adapt effectively, resulting in their prolonged presence and the establishment of stable, host-specific viral lineages. All recognised lineages of IAVs present in terrestrial birds and mammals are originally derived from strains hosted by wild aquatic birds [78, 92].

2.1.5 Hemagglutination Inhibition Assay

The Hemagglutination Inhibition (HI) assay, also termed the HI test, is a widely used method for quantifying antibodies specific to influenza viruses. It is based on the principle that these antibodies can inhibit agglutination of erythrocytes (red blood cells) [94], as shown in Fig. 2.7. This agglutination (i.e., hemagglutination) occurs when the virus's Hemagglutinin (HA) protein binds to Sialic Acids (Sias) on the surface of erythrocytes, effectively "gluing" them together to form a lattice. Thus, the HI assay quantifies the ability of antibodies to inhibit this agglutination (i.e., hemagglutination inhibition, which aids in detecting the antigenic characteristics of the virus.

In the HI assay, antibodies are typically sourced from animals without prior exposure to influenza viruses or related vaccines, with ferrets being the most common choice. The influenza viruses tested are usually collected from human infections. Additionally, Red Blood Cells (RBCs) used in the HI assay are commonly sourced from animals, such as turkeys or guinea pigs. In this assay, a mixture of antibodies, influenza virus, and red blood cells is placed into the wells of a microtiter plate. The wells of these microtiter plates are arranged in rows and columns, labelled with letters and numbers on the plate, respectively. The rows enable the testing of various influenza viruses with a consistent antibody set, while the columns help in distinguishing between the dilutions of antibodies, as illustrated in Fig. 2.8. As previously mentioned, hemagglutination in the HI test indicates that antibodies fail to recognise and attach to influenza viruses. However, if the vaccine-induced antibodies bind to the circulating virus isolated from respiratory specimens of ill patients, it indicates that the circulating virus is antigenically similar to the vaccine strain. Therefore, the HI assay is instrumental in selecting candidate vaccine strains [97, 98], aiding in vaccine licensing [97–102], evaluating vaccine effectiveness [103], conducting antigenic cartography [97, 98], and supporting seroepidemiologic studies to assess population immunity levels [104], among other applications.

However, the efficacy of the HI assay, particularly its predictive accuracy concerning Vaccine Effectiveness (VE), has been a subject of critique [105–107].

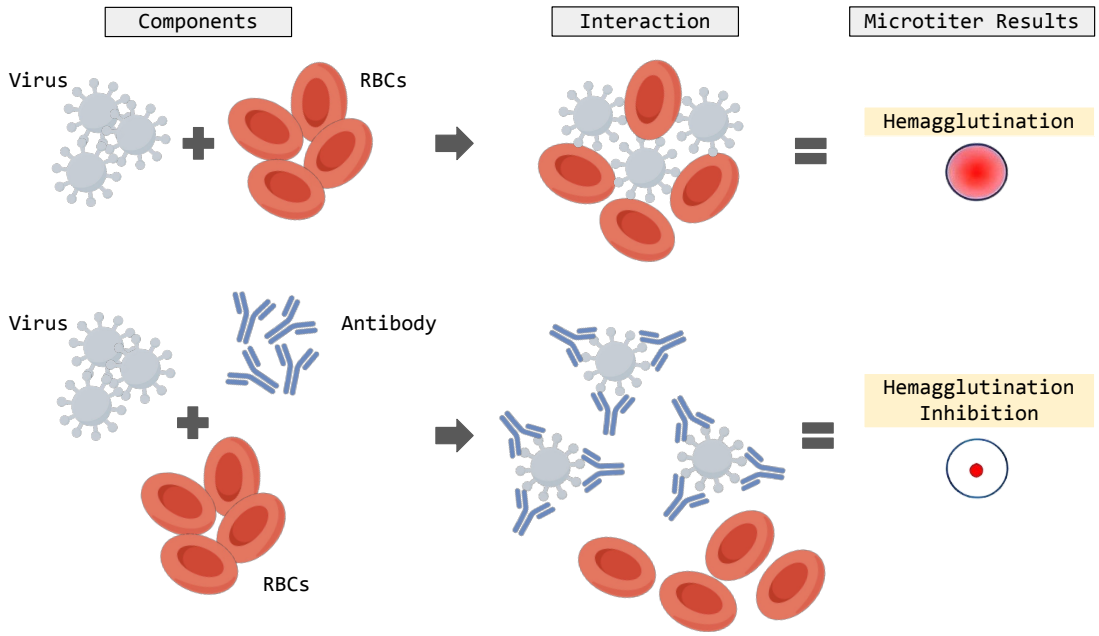


Figure 2.7: Hemagglutination inhibition assay: in a microtiter plate well, Red Blood Cells (RBCs) will settle at the bottom in the absence of influenza viruses. However, when introducing influenza viruses, they bind to the RBCs, causing a lattice formation or hemagglutination, visible as a diffuse layer. This interaction is prevented by antibodies with a strong affinity for the viral Hemagglutinin (HA), which bind to the viruses and inhibit their ability to cause hemagglutination. This results in the RBCs settling as a small red dot at the base of the well, indicating hemagglutination inhibition. The image was adapted from [95] and generated using `Figdraw` [96].

Several factors can influence HI titers (i.e., the results of HI assay), which are not directly connected to antigenic differences. These include the virus strain's propensity to trigger antibody production [108], its affinity for red blood cells, and specific experimental conditions such as temperature and pH. Additionally, antibodies specific to heterologous strains may occasionally be present in low concentrations, implying that slight variations in these concentrations can introduce measurement inaccuracies in the HI titers.

Given the significant variability observed in HI titer measurements across independent assays, it is recommended that estimations of antigenic differences be based on multiple replicated measurements to improve accuracy and reliability [109]. This approach helps mitigate the intrinsic inconsistencies of the HI assay and ensures more robust conclusions about the antigenic characteristics of influenza virus strains.

For more details of HI, please refer to [7, 95, 110, 111].

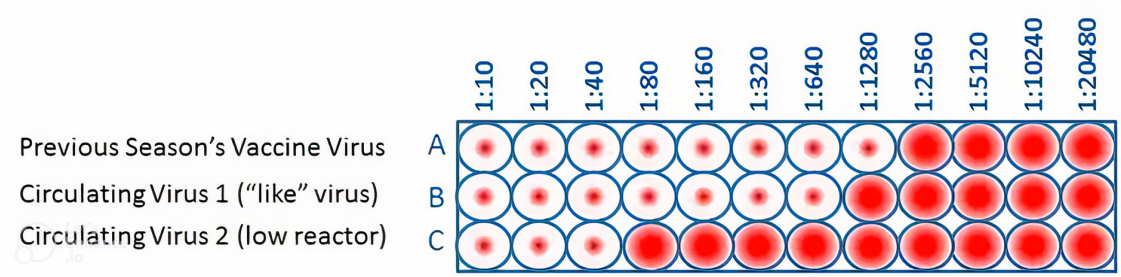


Figure 2.8: Using the HI test to determine antigenic differences between circulating influenza viruses and the previous season's vaccine strain. The microtiter plate shown contains rows and columns of wells with different mixtures of RBCs, influenza virus, and antibodies that were developed against a comparison virus, such as a vaccine virus. The antibody dilutions are marked across the top of the plate and serve as a scale for evaluating antigenic similarity and immune response. The HI test measures how effectively antibodies at higher dilutions can prevent hemagglutination. For example, circulating virus 1 has an HI titer of 640, representing the highest antibody dilution that still inhibits hemagglutination. Viruses are considered antigenically similar if their HI titers are within two dilutions (a four-fold difference) of each other. Thus, circulating virus 1 is considered antigenically similar to the vaccine virus from the previous season, whereas circulating virus 2 is not. The image was sourced from [95].

2.2 Foundations of Machine Learning

2.2.1 What is Machine Learning?

Artificial Intelligence (AI) is a broad domain focused on developing machines that can mimic intelligent human behaviour. Its goal is to devise systems capable of undertaking complex tasks that mirror human problem-solving techniques. The ambition of AI is to craft computational models that are capable of recognising visual scenarios, understanding natural language, and performing actions in the physical world.

Within this broad AI spectrum, machine learning emerges as a crucial sub-field that diverges from traditional programming methods. In traditional programming, the logic and rules are explicitly defined by programmers. However, in machine learning, the model identifies patterns and makes predictions based on the input data it is fed, without explicitly being programmed for each possible situation, as depicted in Fig. 2.9. This "learning" is not a one-off process but rather an ongoing cycle where the model continuously improves its accuracy and efficiency with more data and occasional human guidance to fine-tune its algorithms.

Therefore, the essence of machine learning is the utilisation of data, whether it be numbers, images, or text. This training data is the foundation upon which a machine learning model is built. The accuracy of a model often improves with

the richness of the dataset, provided the data is of high quality, diverse, and relevant to the task at hand, and the model is appropriately complex for the task it is designed to perform. This paradigm shift from traditional, instruction-based programming to a data-driven learning process marks a significant evolution in how we approach problem-solving in computing.

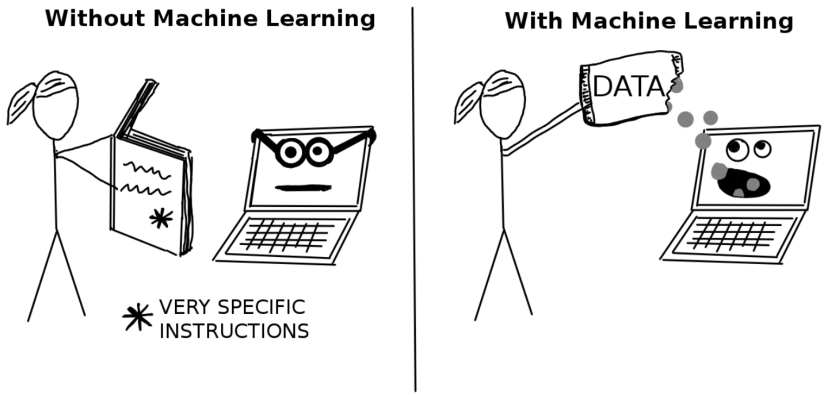


Figure 2.9: Traditional programming vs. machine learning: from explicit instructions to data-driven decisions. The image was sourced from [112].

To evaluate the model's prowess, some data are reserved and not used during training. This evaluation data gauges the model's accuracy on unseen information. Once trained and validated, the model can be used on diverse datasets in the future. This process is known as holdout validation. Another model evaluation method is Cross-validation (CV), detailed in Section 2.2.6.

Machine learning is adept at addressing various problems, including classification (whether binary or multi-class) and regression. Classification problems focus on categorising data into predefined classes, while regression aims to predict a continuous value based on input variables. The models are typically learned by estimating parameters, such as weights, or by learning structures like trees, with their progress and accuracy being guided by minimising a score or loss function. The nature of a model's output delineates its task: if the output is categorical, it's a classification problem; if it's a continuous numerical value, then it's a regression classification. An overview of the standard machine learning problem types, along with their associated loss functions commonly used in deep learning, is presented in Table 2.2.

In the broader landscape, machine learning is a cornerstone of data science. Machine learning algorithms have varied roles: they can be descriptive, offering explanations based on data; predictive, forecasting future events; or prescriptive, suggesting potential courses of action [113]. By employing statistical techniques,

Problem Types	Output Unit	Loss Function
Binary Classification	Logistic	Cross-entropy Loss
Multi-class Classification	Softmax	Categorical Cross-entropy Loss
Regression	Linear	Mean Squared Error (MSE) Loss

Table 2.2: Classic problem types of machine learning.

these algorithms make predictions, categorise data, and unearth pivotal insights. Such revelations guide strategic decisions and can potentially influence critical growth indicators.

2.2.2 Common Machine Learning Paradigms

Machine learning enables computers to learn from data, and it can be widely classified by the type and extent of supervision they need during training.

Supervised Learning Supervised machine learning is a paradigm where models are trained using labelled data sets, meaning each training sample is paired with the correct output so that we already know the outcome of interest. This allows the algorithm to gradually enhance its accuracy by learning from the provided data. Supervised learning aims to understand the relationship between inputs and outputs (i.e., learning a model that can map input features to outputs), aiming to make accurate predictions on unfamiliar data (i.e., unseen data).

Fig. 2.10 illustrates an example of supervised learning, in which the goal is to predict the target y based on input features X. The learner is trained on labelled data to build a predictive model that can make predictions $Y_{predicted}$ based on new input features X_{new}.

Unsupervised Learning Contrary to supervised learning, which relies on labelled data, unsupervised learning is trained on unlabelled data (i.e., target y is not provided). It depends on the algorithm's capacity to autonomously discern patterns and structures within the input data without explicit guidance. The most common unsupervised learning method is clustering, where the algorithm tries to group similar data. This method has practical applications in scenarios such as anomaly detection, including fraud identification. Another common method is dimensionality reduction, where the algorithm identifies and extracts the most important features from the data, thereby preserving the essence of the original dataset's properties. This technique is particularly useful for data visualisation and optimising the data

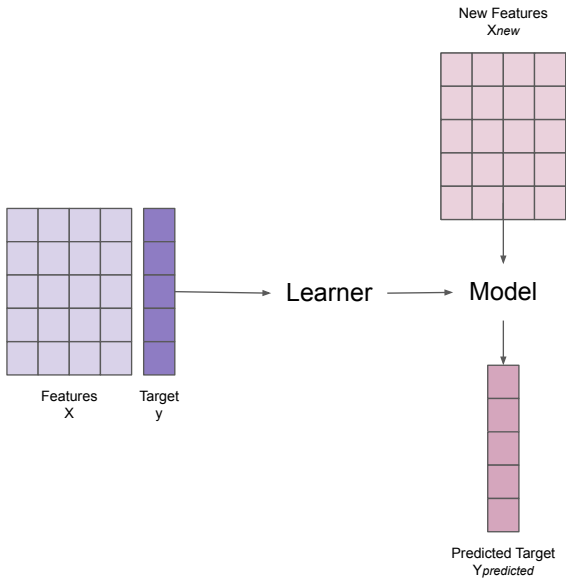


Figure 2.10: An example of supervised learning, adapted from [112].

for subsequent algorithmic processing.

Semi-supervised Learning Semi-supervised learning, also known as weak supervision, lies between supervised and unsupervised learning and is particularly useful in situations where obtaining labelled data is challenging, and the process of labelling data is impractical. In this approach, the model is trained on a blend of a relatively small amount of labelled data complemented by a larger amount of unlabelled data. The main idea behind semi-supervised learning is that the model can leverage the unlabelled data to better understand the overall data distribution and thus improve its performance on the labelled data.

Self-supervised Learning As the name suggests, self-supervised learning enables models to be supervised by "itself". Unlike traditional unsupervised learning, which uses unlabelled data, self-supervised learning involves generating labels from the input data, thereby transforming the learning task from unsupervised to supervised. The essence of self-supervised learning lies in its ability to discern hidden parts of the input from its unhidden counterparts. It leverages the inherent structure of the data to generate various supervisory signals across data sets without depending on external labels. These signals serve as integral feedback during the training process. In self-supervised learning, the model is trained to solve a specific task (i.e., "pretext" task). Once trained on this pretext task, the model can be fine-tuned for a downstream task.

An example is a model trained to predict the next word in a sentence. In

this scenario, the model is trained on many sentences where one word is removed, and its task is to predict the missing word. This approach allows the model to learn important features about the data unsupervised, which is helpful for other tasks like classification or regression.

Reinforcement Learning Reinforcement learning differs from the previously mentioned learning methods. Unlike supervised or unsupervised learning, reinforcement learning does not necessitate a supervisor or pre-labelled data. Instead, it gains its training data through experience, acquired by interacting with its environment and observing the responses to its actions. The goal of reinforcement learning is to develop an optimal strategy or policy that maximises cumulative reward over time. Cumulative reward refers to the total sum of rewards an agent collects, where each reward is a numerical feedback signal by the agent after executing an action. This approach is beneficial in scenarios where the correct decision or action is not predetermined but can be deduced from the rewards or penalties following an action. Applications of reinforcement learning are diverse, including game playing, robotics, and specific optimisation problems, where it excels in determining the best course of action in complex and dynamic environments.

2.2.3 Machine Learning and Deep Learning

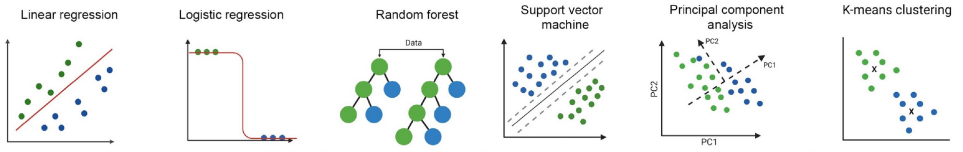
It's not uncommon to encounter confusion or misinterpretations regarding the terms AI, machine learning, and deep learning. In simple terms, machine learning is a subfield within artificial intelligence, and deep learning is a subset of machine learning.

Meep learning models are supplied with features (i.e., input data) and corresponding labels (i.e., output data). During the training phase, the model learns to map these inputs to the desired outputs. This learning process is directed by a loss function, which quantifies the divergence between the model's predictions and the actual labels. Additionally, model parameters are refined, and the model's generalisation capabilities are evaluated using a validation set.

Meep learning includes both traditional machine learning and deep learning approaches. Traditional machine learning relies on manually crafted features and established algorithms, such as decision trees or support vector machines, primarily suited for structured data. In contrast, deep learning uses multi-layered neural networks to extract features from raw data autonomously, resulting in more sophisticated data representations and a deeper understanding of underlying patterns and complexities.

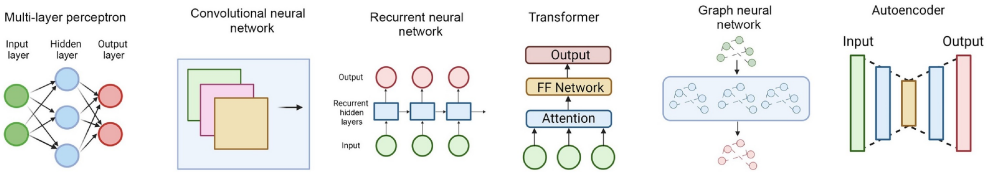
Fig. 2.11 presents a comparative overview of some popular algorithms in both traditional machine learning and deep learning [114]. Traditional machine learning algorithms are usually simpler and less computationally intensive compared to deep learning, which requires substantial computational power and extensive data for effective performance, as shown in Fig. 2.12. However, deep learning has demonstrated exceptional performance in areas such as image recognition, natural language processing, and complex decision-making tasks.

(A) Machine learning algorithms



Model objective	Model the relation between two classes using a line of best fit	Discriminate between two classes of binary variables	Classify variables into groups using decision trees	Discriminate classes by maximizing a hyperplane	Dimensionality reduction	Cluster points around centroids based on mean distance
Complexity	+	+	++	+	+	+
Performance	+	+	+	++	++	+

(B) Deep learning algorithms



Model objective	Pass information among neurons to make predictions	Perform convolutions on filters across the data	Use temporal data to make predictions	Using contextual awareness to make predictions	Capture networks structures within a graph	Learn embedding of the data
Complexity	++	+++	+++	++++	+++	+++
Performance	++	++++	++++	++++	+++	++

Figure 2.11: Comparative overview of traditional machine learning and deep learning algorithms, sourced from [114].

Deep learning applies complex Neural Network (NN) architectures to identify intricate patterns in data. Fig. 2.13 provides an example of a fully connected NN. These neural networks are computing systems inspired by the structure of the human brain, consisting of layers of nodes, often referred to as neurons, interconnected by edges or weights. Data enters the network through an input layer, gets processed in one or more hidden layers, and ultimately generates an output from the output layer. In a fully connected (or dense) NN, each node in one layer is linked to all nodes in the following layer.

2.2.4 Machine Learning Workflow

The machine learning workflow typically includes the following steps:

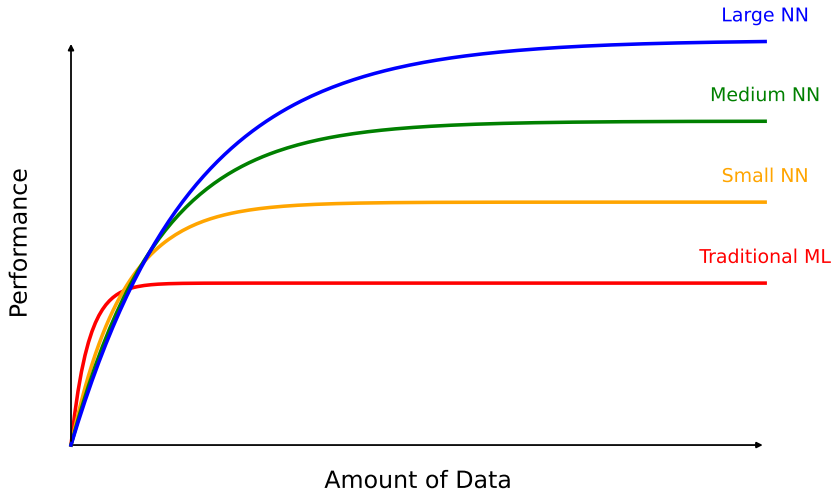


Figure 2.12: The impact of data volume on model performance varies with the complexity of machine learning models: traditional machine learning models initially benefit from increased data volumes but quickly reach a performance ceiling due to complexity constraints. In contrast, shallow neural networks use a few hidden layers to handle larger datasets more efficiently, thereby extending their performance limit. Medium neural networks have more hidden layers that make them better at capturing complex patterns, thus increasing their capacity to learn from more datasets. Deep neural networks have even more layers, allowing them to process complex data and benefit significantly from larger datasets.

Data Preparation The foundation of any machine learning model lies in its data – the more, the better. Data should be collected, meticulously cleaned, and rigorously preprocessed. This preprocessing may encompass normalisation, addressing missing values, and transforming the data into a suitable format for training. While data quality can affect model performance, this does not always hold. Data can also influence the choice of subsequent methods to be employed. For instance, if the data consists of a small number of labelled samples and many unlabelled data, then we can consider using semi-supervised learning techniques.

Model Design This step involves choosing suitable machine learning algorithms. If selecting the deep learning models, then it involves the designation of NN architecture, regarding the number of layers, the number of neurons in each layer, and the type of layers (dense, convolutional, recurrent, etc.) that are made at this stage. A suitable optimisation method for hyperparameter tuning can also be chosen at this stage.

Training To mitigate the risk of overfitting, preprocessed data is usually divided into distinct sets for training, validation, and testing. Once the model is designed,

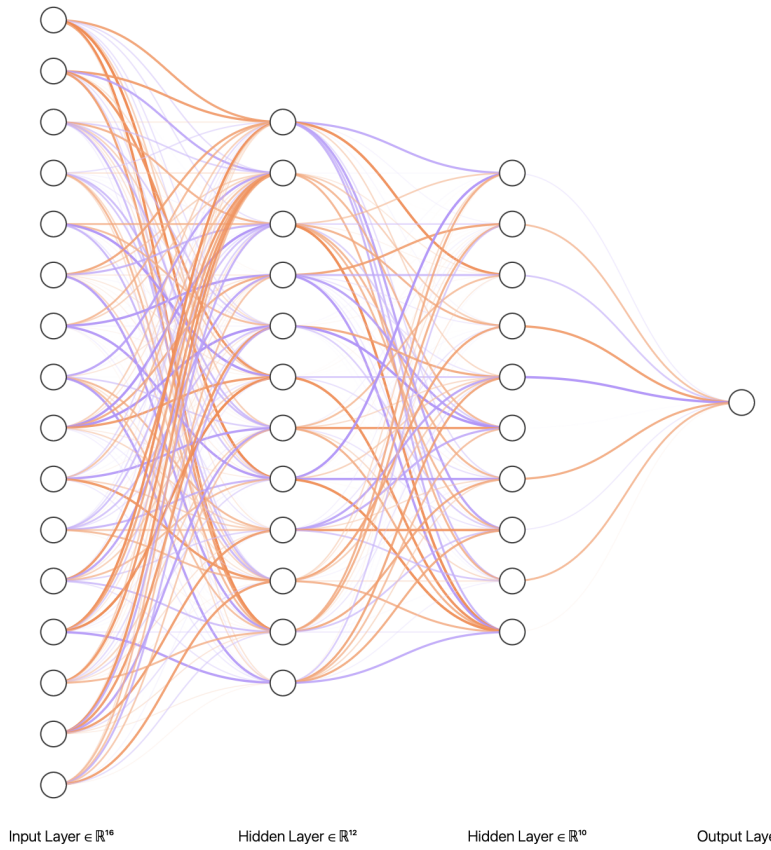


Figure 2.13: An example of a fully connected neural network: this neural network has an input layer with 16 nodes, two hidden layers with 12 and 10 nodes, respectively, and an output node that is suitable for binary classification or regression tasks. In binary classification, this node would typically output a probability score for one class versus another, while in regression, it would output a continuous value. Conversely, networks with multiple output nodes are better suited for multi-class classification tasks, where each node corresponds to a distinct class, and the network outputs a probability distribution across these classes. The image was generated by `NN-SVG` [115].

it is trained using the training dataset. During this phase, the deep learning models primarily focus on adjusting internal weights through a learning process. In contrast, classic machine learning models use a variety of techniques tailored to their specific algorithms. These techniques include, but are not limited to, generating decision-making rules, identifying optimal hyperplanes for data separation, and in some cases, straightforwardly storing data for subsequent reference. Each technique represents a unique approach to "learning" from training data, enabling the model to make informed predictions or classifications based on the learned patterns.

When optimisation algorithms are employed for hyperparameter tuning, the validation set is used to evaluate the model's performance during the training process, facilitating the fine-tuning of hyperparameters to enhance overall performance.

Evaluation After training, the model's performance is usually evaluated on a test set to ensure it generalises well to new, unseen data. The validation and test sets should not be used interchangeably due to their distinct functions in the model development process. The test set is used in the final evaluation, while the validation set is primarily used for hyperparameter tuning and model evaluation during the training phase.

Figure 2.14 illustrates the general process flow for evaluating machine learning or deep learning models. It shows how the model processes input data to produce predictions. These predictions are then compared with the true labels to generate a Confusion Matrix (CM) and Receiver Operating Characteristic (ROC) curve for evaluating performance metrics. The validation set, although not explicitly mentioned, is typically utilised immediately after the model is trained on the training data. It aids in tuning hyperparameters and validating the model's performance iteratively. Once the model is fully trained and the hyperparameters are optimised, the test set is introduced. It provides an unbiased evaluation of the model's performance. This is depicted in the latter part of the figure, where true labels are compared against predicted labels to construct the CM and calculate the metrics represented by the ROC curve.

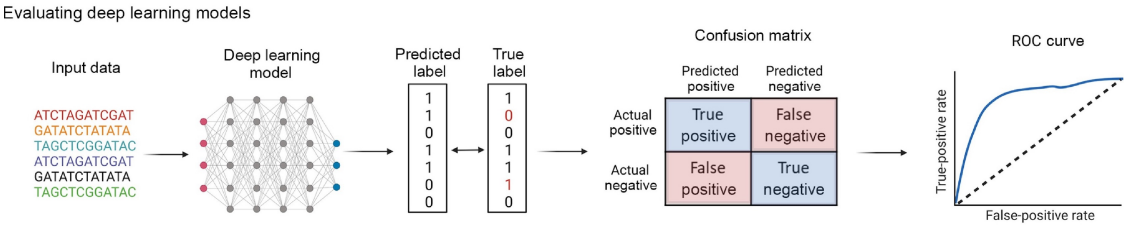


Figure 2.14: Process flow for evaluating deep learning models. The image was sourced from [114].

2.2.5 Machine Learning Algorithms

This section introduces the machine learning algorithms used in this thesis.

Random Forest

The Random Forest (RF) algorithm [116] represents an advanced ensemble learning technique that combines the principles of bagging (i.e., bootstrap aggregating) and decision trees to enhance model performance. Bagging effectively reduces the variance of predictive models by creating multiple datasets from the original dataset through bootstrapping (sampling with replacement) and then aggregating

the models trained on these datasets. Decision trees, in contrast, typically have higher variance but lower bias. By combining bagging with decision trees, RFs achieve a balance, resulting in a robust model with improved accuracy and stability.

In a RF, each decision tree within the ensemble is exposed to only a subset of features when determining where to split at each node, rather than considering all features. This strategy introduces an element of randomness, thereby increasing the diversity among the trees and helping to reduce overfitting, a common problem with deep decision trees. In contrast to boosting-based ensembles, which sequentially build models with a focus on correcting the errors of previous models, bagging-based ensembles like RF construct trees independently and allow them to grow deep. This independence and depth lead to more complex models but also contribute to the model's robustness and accuracy.

However, while bagging-based ensembles such as RF might require more computational resources and time due to their complexity, they often obviate the need for a separate validation dataset to estimate generalisation performance. This is because the aggregation of multiple, diverse decision trees inherently provides a measure of validation, offering a reliable estimate of the model's performance on unseen data. Through this internal validation mechanism, RFs can provide insights into their expected accuracy and stability without additional validation steps.

Extreme Gradient Boosting

Extreme Gradient Boosting (XGBoost) [117] represents a scalable and efficient implementation of gradient boosting algorithms. Gradient boosting is an advanced ensemble technique that sequentially builds models to correct the errors of previous ones, and uses gradient descent to optimise a differentiable loss function. This approach is different from and more sophisticated than earlier ensemble methods such as Adaptive Boosting (AdaBoost), which primarily focus on reweighting misclassified instances.

A key strength of XGBoost is its versatility and robustness in tackling complex challenges in data science. It is engineered to manage missing values automatically and efficiently process sparse data, common issues in real-world datasets. This capability is due to its innovative handling of sparse structures, which significantly reduces memory use and improves computational efficiency.

Moreover, XGBoost distinguishes itself by including built-in mechanisms for regularisation, both L_1 and L_2 forms, which help prevent overfitting, a frequent issue with standard gradient boosting methods. This integration of regularisation

terms into the model's objective function enhances its robustness and predictive performance across a variety of datasets.

Another salient feature of XGBoost is its parallel processing capability. Unlike traditional gradient boosting, which processes components sequentially, XGBoost leverages modern computational architectures to parallelise the construction of decision trees. This parallelism, coupled with advanced optimisation techniques such as cache-aware access patterns and out-of-core computing, significantly expedites the model training process, making XGBoost exceptionally efficient for large-scale and complex datasets.

In essence, XGBoost embodies a versatile, robust, and highly efficient solution for both classification and regression problems, equipped with unique features such as automatic handling of missing data, advanced regularisation, and efficient parallel training. These attributes make XGBoost a preferred choice for data scientists aiming to tackle challenging predictive modelling tasks with precision and speed.

Random Undersampling Boosting

Addressing class imbalance in machine learning is a critical challenge, often addressed through data sampling techniques and boosting algorithms. Data sampling methods, such as oversampling (which augments minority class instances) and undersampling (which reduces instances of the majority class), are commonly employed strategies to achieve a more balanced dataset. The Random Undersampling Boosting (RUSBoost) algorithm [118] integrates the concept of undersampling with boosting techniques to handle class imbalance effectively.

RUSBoost combines the advantages of random undersampling of the majority class with the iterative model building of boosting, thereby enhancing the focus on minority class examples during the learning process. This integration helps address the imbalance and contributes to building a more robust predictive model. Compared to other oversampling methods like SMOTEBoost [119], which is based on Synthetic Minority Oversampling Technique (SMOTE), RUSBoost is recognised for its computational efficiency and efficacy. It offers a cost-effective solution by reducing the training time and resource consumption, making it a viable option for large data sets where class imbalance is a significant concern.

Support Vector Machine

Support Vector Machine (SVM) is one of the most commonly used supervised learning algorithms [120]. A key strength of SVMs lies in their ability to classify not

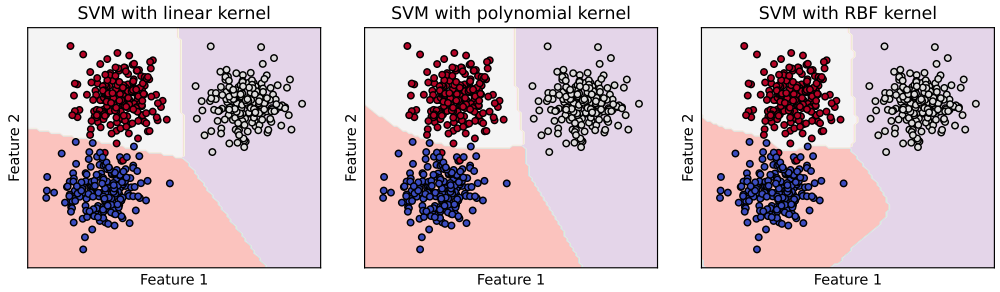


Figure 2.15: Comparison of support vector machine classification boundaries using linear, polynomial, and Radial Basis Function (RBF) kernels.

only linearly separable data but also non-linearly separable data, courtesy of the "kernel trick". This technique enables SVMs to process high-dimensional, and even infinite-dimensional data effectively. It achieves this by mapping lower-dimensional data into a higher-dimensional space, thus facilitating the separation of complex data sets without the need for explicit data transformation.

A key feature of SVMs is its ability to identify the optimal hyperplane that distinctly separates different classes within the feature space. This is achieved by maximising the margin between the data points of different classes, ensuring a clear and definitive classification boundary. Additionally, SVM's versatility is further enhanced by using various kernel functions, such as the Gaussian kernel, which allows for the construction of a multi-class SVM capable of tackling complex, multi-dimensional data sets.

Fig. 2.15 illustrates SVM classification boundaries using different kernel functions. This comparison showcases how the choice of the kernel can significantly influence the SVM's ability to delineate between classes.

Multi-Layer Perceptron

The Multi-Layer Perceptron (MLP) is a type of feed-forward neural network that addresses the limitations of single-layer perceptrons, particularly their inability to solve problems that are not linearly separable [121]. A MLP typically comprises three layers: an input layer, one or more hidden layers, and an output layer. Each layer consists of neurons, with connections flowing from the input to the output layer without any cycles, characterising its feed-forward nature.

Fig. 2.16 illustrates a standard five-layer fully connected MLP architecture, including the one input layer, three hidden layers and one output layer. The number of neurons in the input layer corresponds to the number of features present in the input data, ensuring that each feature is adequately represented. Similarly, the

number of neurons in the output layer is determined by the number of classes in the dataset, facilitating the classification or regression tasks. Each neuron in a given layer is connected to every neuron in the subsequent layer, forming a dense network of connections that enables complex data processing and pattern recognition. This architecture allows the MLP to learn and model relationships in the data, making them suitable for a wide range of applications from simple binary classification to complex multi-class classification tasks.

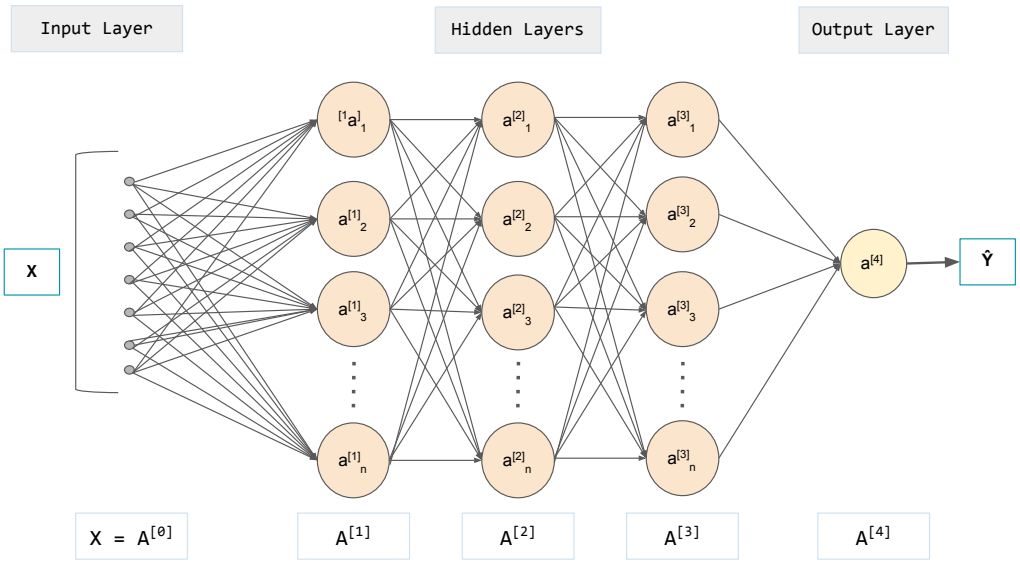


Figure 2.16: Example of a fully connected MLP architecture, adapted from [122].

Convolutional Neural Network

Convolutional Neural Networks (CNNs) are widely used in a variety of fields, including facial recognition, object recognition, and autonomous vehicles. CNNs were initially trained on images but have since expanded their applicability to encompass diverse data types, including time series, text, and audio. In contrast to traditional machine learning techniques, CNNs are designed to learn and extract features at each hidden layer autonomously. CNNs also differ from standard fully connected neural networks, such as the one depicted in Fig. 2.16, regarding their sparse layer connectivity and reduced number of parameters that must be learned. This distinction not only enhances efficiency but also improves computational feasibility, making CNNs more adept at handling complex data processing tasks.

A typical CNN architecture consists of three core layers: the convolutional layer, which is responsible for learning spatial features; the activation layer, which

activates significant features; and the pooling layer, which downsamples features to reduce dimensionality. In many CNN models, the number of filters tends to increase as the network delves deeper. This approach is based on the rationale that as the spatial resolution of feature maps decreases due to pooling layers or strided convolutions, having a greater number of channels becomes beneficial to capture more complex and high-level features.

However, variations exist, such as in some protein-related prediction tasks or when CNNs are used as encoders. In these scenarios, a decreasing or constant number of filters in convolutional layers can be observed. Such configurations are designed to capture local patterns in the initial layers, which are then integrated into more abstract representations in the deeper layers. This demonstrates the adaptability and versatility of CNN architectures in handling a wide range of complex tasks.

Transformers

The Transformer model represents a breakthrough in neural network design and forms the basis for advanced applications such as Bidirectional Encoder Representations from Transformers (BERT) and the Generative Pre-trained Transformer (GPT) series. This model has shown superior performance in language translation tasks compared to Recurrent Neural Network (RNN), with higher Bilingual Evaluation Understudy (BLEU) scores [123]. RNNs, which processes words sequentially, faces limitations in training speed and struggles with long sequences due to vanishing and exploding gradient problems.

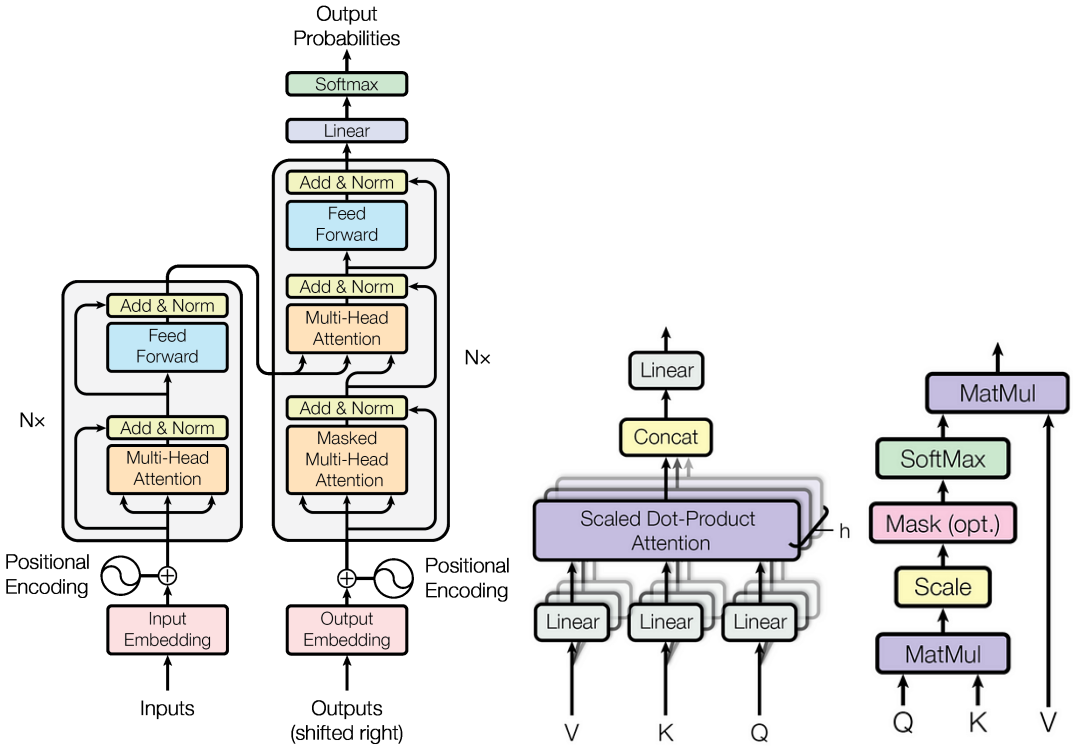
Transformers, however, move away from the recurrent approach and use a self-attention mechanism. This mechanism allows the model to focus on specific parts of the input data. The key feature of the Transformer is its scaled dot-product attention, which is defined by the formula:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.1}$$

where d_k is the dimension of the key, so $1/\sqrt{d_k}$ is the scaling factor. Q, K and V denote the query vector, key vector and value vector, respectively. The softmax function converts the attention score to attention distribution. The core idea behind the dot-product attention is that the dot product is higher between similar sequences than in dissimilar ones.

Another important innovation of the Transformer is the multi-head attention. This feature combines several scaled dot-product attentions, leading to a more diverse and effective model. This multi-faceted approach prevents the model

from excessively focusing on specific words, thereby yielding more robust results compared to single-head attention. However, like the bag-of-words model, attention mechanisms inherently lack sequential order information. To address this, the original design added positional encoding to the input embeddings. Fig. 2.17 shows the architecture of the Transformer, including the scaled dot-product attention and the multi-head attention mechanisms. More detailed information about the Transformer is available in [123].



(a) Transformer architecture **(b)** Multi-head attention **(c)** Scaled dot-product attention

Figure 2.17: The architecture of the Transformer model, sourced from [123].

Bidirectional Recurrent Neural Networks

Within the domain of neural networks tailored for sequential data processing, such as text or time series data, two advanced variations of bidirectional RNNs stand out for their effectiveness in various applications: Bidirectional Long Short-Term Memory (BiLSTM) and Bidirectional Gated Recurrent Unit (BiGRU).

LSTM is an advanced type of RNN that solves the vanishing gradient problem seen in standard RNNs [124, 125]. A typical LSTM uses three gates: an input gate, which stores new information from the current input and selectively updates the cell state; a forget gate, responsible for removing irrelevant data; and an

output gate, which determines what information gets passed to the next state. Bidirectional LSTM, or BiLSTM, combines a forward and a backward LSTM. It processes data in both directions to better understand context and is more effective than unidirectional LSTM [126, 127].

Gated Recurrent Unit (GRU) is another RNN variant similar to LSTM, but it has only two gates: a reset gate and an update gate [128]. These gates help the GRU decide how much of the past information to keep and how much new information to add. GRUs are generally faster in training than LSTMs because they have simpler operations. Similar to BiLSTM, bidirectional GRU also works in both forward and backward directions, enabling the network to capture a broader context. Fig. 2.18 shows an architectural comparison of RNN, LSTM and GRU.

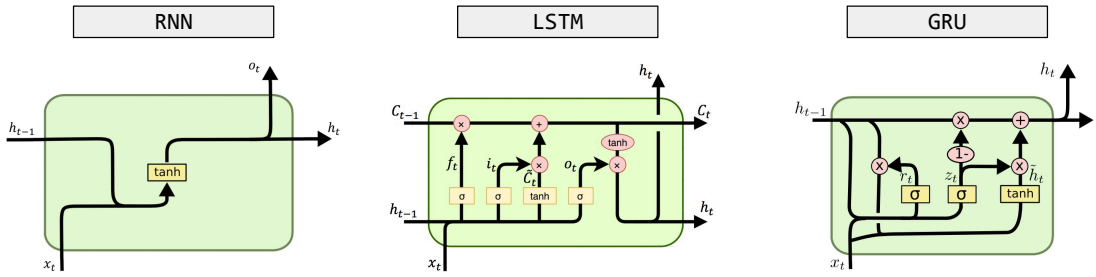


Figure 2.18: Architectural comparison of recurrent neural networks.

2.2.6 Nested k-fold Cross-validation

As mentioned in Section 2.2.4, the data is usually split into separate subsets for training, validation, and testing during model training. This is a crucial step to ensure that the model not only learns from the data but also generalises well to new, unseen data. One of the most robust methods to achieve this balance is through Cross-validation (CV).

CV is primarily used for preventing overfitting, a common issue where a model performs well on its training data but poorly on new, unseen data. Fig. 2.19 displays three different outcomes of model fitting: underfitting, where the model is too simple and does not capture the complexity of the data; optimal fitting, where the model balances complexity and simplicity, accurately reflecting the true distribution without being influenced by the noise in the data; and overfitting, where the model is too complex and captures the noise in the training data, which impairs its performance on new data.

Cross-validation includes a variety of methods with their approach to partitioning data and evaluating model performance. For example, Leave-One-Out Cross-

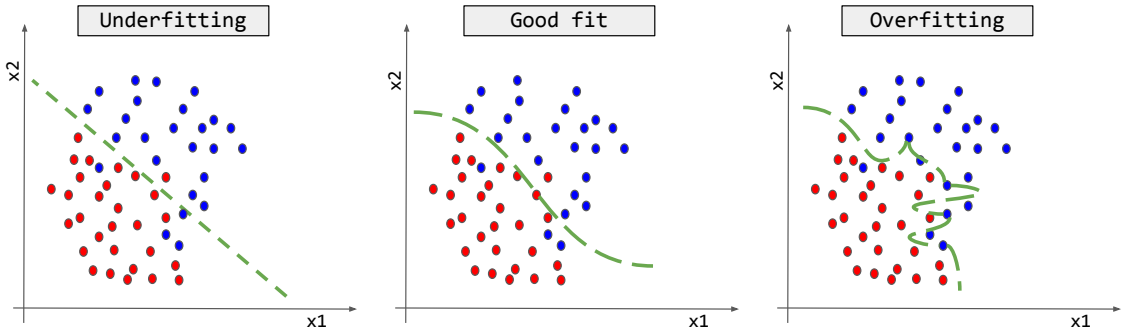


Figure 2.19: Comparative visualisation of model fitting in machine learning, adopted from [122].

Validation (LOOCV) uses a single observation as the validation set for each iteration; the holdout method simply splits the data into a single training and test set; and repeated random sub-sampling validation, also known as Monte Carlo CV, repeatedly divides the data at random into validation and training sets; and k-fold CV partitions the data into k equally-sized folds and systematically uses each as a validation set once. Amongst these, k-fold CV, particularly when stratified and nested, is the preferred choice for this thesis. It is a well-balanced validation process, using all available data points across multiple training and testing iterations. Therefore, our subsequent discussions will be dedicated to detailing k-fold CV, with a particular focus on stratified k-fold CV and nested k-fold CV.

Stratified k-fold CV preserves the proportion of classes in the training set that is almost the same as that of the test set, maintaining class distribution consistency across folds, irrespective of whether the approach is nested or not. Nested k-fold CV differs from the standard k-fold CV in that it adds an inner loop. The standard k-fold CV divides the data into k parts and performs k rounds of evaluations. In each evaluation, one part is selected as the testing set, ensuring that every part of the data is used for testing exactly once during the entire process. In contrast, nested k-fold CV introduces an additional layer within each outer fold, where the remaining data is further divided into k smaller subsets for the inner loop. This inner loop is used for hyperparameter tuning, ensuring that the model selection is unbiased and that the evaluation of the outer loop is based on a completely independent dataset [129]. This method provides a more rigorous evaluation of the model's predictive performance. The example of nested 5-fold CV is shown in Fig. 2.20.

Hyperparameter tuning is typically conducted within the inner loop of nested k-fold CV. This crucial step can be achieved using a variety of methods, with Bayesian optimisation (BO) being one such example. The pseudo-code detailing the

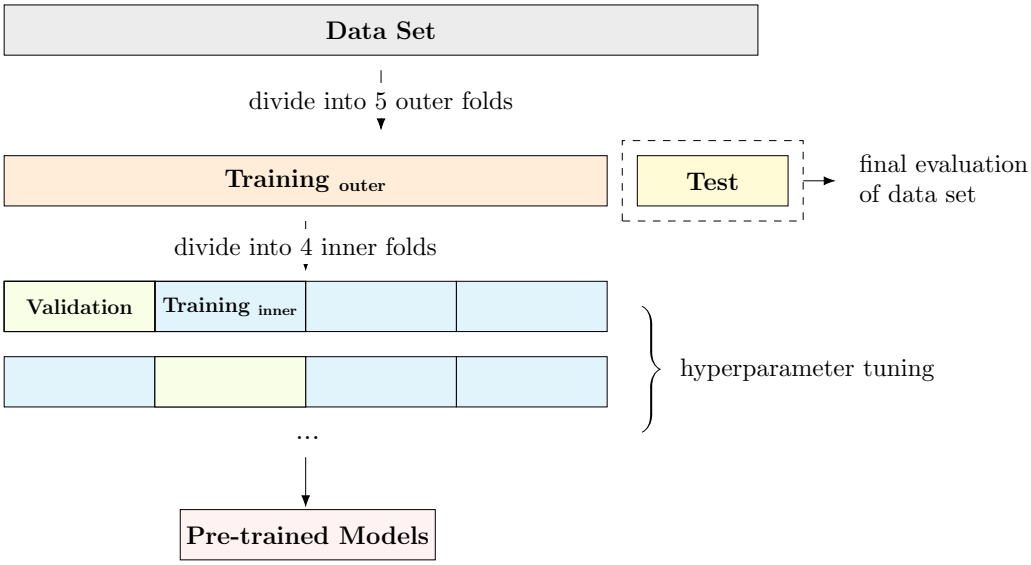


Figure 2.20: Example of nested k-fold cross-validation ($k_{outer} = 5$ and $k_{inner} = 4$).

implementation of nested k-fold CV with such optimisation techniques is presented in Fig. 2.21. While nested k-fold CV provides a thorough evaluation, it demands significant computational resources and time. In situations with constraints on time or computational capacity, alternative validation techniques like random splitting may be more appropriate.

2.2.7 Bayesian Optimisation

Modern machine learning algorithms usually have many hyperparameters, and their performance can vary significantly depending on the specific set of hyperparameters chosen. Therefore, it's essential to select appropriate hyperparameter values for the effective application of these algorithms. Bayesian Optimisation (BO) is a frequently used technique for hyperparameter tuning and optimisation of any black-box function. This optimisation approach centres on a key question: "Considering what we currently know, which data point should we evaluate or explore next?" This involves finding a trade-off between exploration and exploitation through acquisition functions, which are heuristics indicating the desirability of evaluating a point based on the current model.

In BO, a model is maintained to represent estimates and uncertainties at each point. The model is typically a surrogate model, such as a Gaussian Process (GP), to model the objective function and define its prior distribution. Then, the posterior is obtained using Bayes' rule based on observations (i.e., function evaluations). The following sample point is guided by an acquisition function, which depends

Algorithm 1: Nested k-fold Cross-validation with Bayesian Optimisation

Data: Data set with features X and labels y, $\mathcal{D} = \{X, y\}$
input : Number of inner folds k_{inner}, number of outer folds k_{outer}
Maximum number of steps of Bayesian Optimisation n_{iter}
Maximum tuning time $MaxTime$
output: Generalisation error E_g of the model

Shuffle $\mathcal{D}$;
Stratified split $\mathcal{D}$ into k_{outer} folds;
for $i = 1$ **to** k_{outer} **do**
  Take i^{th} fold of $\mathcal{D}$ as test set $\mathcal{D}^i_{test}$ and remaining as training set $\mathcal{D}^i_{train}$;
  Stratified split $\mathcal{D}^i_{train}$ into k_{inner} folds;
  for $k = 1$ **to** n_{iter} **do**
    for $j = 1$ **to** k_{inner} **do**
      Take j^{th} fold of $\mathcal{D}^i_{train}$ as validation set $\mathcal{D}^j_{val}$ and remaining as training set $\mathcal{D}^j_{train'}$;
      Train the model on $\mathcal{D}^j_{train'}$ with hyperparameter set $\mathcal{P}_k$;
      Compute validation error $E^{k,j}_{val'}$ of the model on $\mathcal{D}^j_{val}$;
    Compute average validation error E^k_{val}, where $E^k_{val} = average(E^k_{val'})$;
    if *elapsed time* $> MaxTime$ **then**
      Stop tuning
  Select optimal hyperparameter set $\mathcal{P}_{opt}$ with the lowest E_{val};
  Fit the model on $\mathcal{D}^i_{test}$ with $\mathcal{P}_{opt}$;
  Compute the test error E^i_{test} of the model on $\mathcal{D}^i_{test}$;
Compute generalisation error E_g of the model, $E_g = average(E_{test})$;

Figure 2.21: Pseudo code of stratified nested k-fold cross-validation.

on the posterior. Following this, the new data point is added to the existing observations, and the posterior is updated accordingly. This iterative process of adding new data and updating the model continues until either convergence is achieved or available resources are exhausted.

While parameters in machine learning models are learned from data, hyperparameters are pre-set and require careful selection. For less complex models or when training is not costly, a grid search can effectively find optimal hyperparameters. However, grid search becomes unfeasible for cost-intensive functions like training extensive neural networks. BO thus can be a more efficient solution to quickly find hyperparameter settings that yield good performance. While BO is efficient, it can sometimes get stuck in local minima, potentially overlooking better solutions.

We used BO for hyperparameter tuning across all models. For traditional ma-

chine learning models, `skopt.BayesSearchCV` [130] was the chosen implementation, whereas `keras_tuner.BayesianOptimization` [131] was applied for deep learning models, both using their default acquisition functions.

Further details about BO can be found in [132–134].

2.2.8 Evaluation Metrics

Confusion Matrix The Confusion Matrix (CM) is commonly used for summarising a classification model's predictions on test data, in a matrix format. It is suitable for both binary and multi-class classification scenarios. The matrix is square-shaped, with its columns and rows equalling the total number of classes. Thus, the CM is an $n \times n$ matrix, where n is the number of classes. The CM for binary classification is illustrated in Fig. 2.22. In this matrix, the columns represent the predicted classes, and the rows indicate the actual classes. This matrix provides insights into which classes the model may incorrectly interpret. Furthermore, the CM facilitates the calculation of key metrics such as Accuracy (Acc), Recall (Rec), Precision (Pre), and Specificity (Spec).

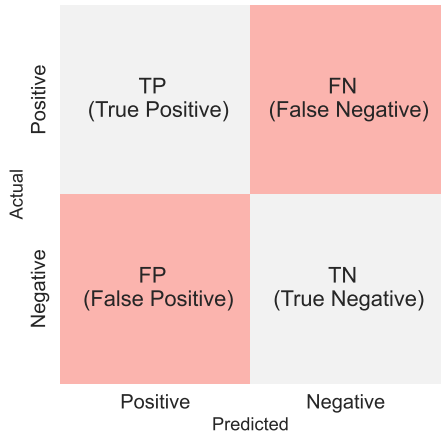


Figure 2.22: The confusion matrix for binary classification.

True Positive and True Negative True Positive (TP) and True Negative (TN) represent the number of correct predictions (the diagonal of CM). Specifically, TP corresponds to correctly predicted positive instances, while TN relates to correctly classified negative instances. For example, for a binary classification problem, such as determining a patient's health status, where "positive" denotes being unhealthy and "negative" indicates healthiness, TP quantifies the correctly

diagnosed patients as unhealthy. In contrast, TP corresponds to the number of patients correctly discerned as healthy.

False Positive and False Negative False Positive (FP) and False Negative (FN) denote the number of incorrect predictions (the off-diagonal of CM). FP is the number of negative instances incorrectly classified as positive, while FN is vice versa. For example, FP refers to the instances where healthy patients (negative class) are erroneously diagnosed as unhealthy (positive class). In contrast, FN represents the number of patients misdiagnosed as healthy when they are unhealthy.

Accuracy Accuracy (Acc) is the ratio of correctly classified instances to the total number of instances within the test dataset. This metric ranges between 0 and 1, where 1 represents the perfect prediction of all positive and negative samples, and 0 indicates a complete failure to correctly predict any positive or negative samples.

Although accuracy is a commonly used metric for evaluating model performance, it may not always provide a reliable indication, particularly in cases of imbalanced data sets. A model might achieve high accuracy in such scenarios by predominantly classifying instances into the more frequent class. As a result, it becomes crucial to consider additional metrics, such as recall, precision, and the F_1 score. These metrics offer a more thorough understanding of a model's performance across various classes, particularly when there is a significant disparity in class distribution.

The formula for calculating accuracy is expressed as:

$$\mathrm{Acc} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}} \tag{2.2}$$

Recall The Recall (Rec), also referred to as Sensitivity or the True Positive Rate (TPR), is a measure of the proportion of positive samples that the model correctly classifies. It is calculated as the ratio of the correctly classified positive samples (true positives) to the total number of actual positive samples. The recall value lies within the range of $[0, 1]$, where 1 represents the correct prediction of all samples in the positive class, and 0 indicates a complete failure in correctly predicting any sample in the positive class. recall is particularly crucial in medical research and diagnostics, where the aim is to minimise the number of missed positive cases, thereby necessitating a high recall value.

The formula for recall is given by:

$$\mathrm{Rec} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{2.3}$$

Precision Precision (Pre) measures the accuracy of positive predictions. It is calculated as the ratio of correctly identified positive instances (TPs) to the total number of instances predicted as positive, including both correct (TPs) and incorrect (FPs) predictions. Essentially, precision answers the question: "Of all the instances classified as positive, how many were actually positive?" The value of precision ranges between 0 and 1, where 1 indicates perfect precision (all instances predicted as positive are indeed positive), and a value of 0 means none of the instances predicted as positive are actually positive.

The formula for calculating precision is as follows:

$$\text{Pre} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.4}$$

Specificity Specificity (Spec), often considered the counterpart of recall for the negative class, measures the proportion of actual negatives that are correctly identified. It is computed as the ratio of correctly classified negative samples to the total number of actual negative samples. The range of specificity lies between 0 and 1, where 1 represents the perfect classification of all negative class samples, and 0 indicates a complete misclassification of negative class samples.

The formula for specificity is:

$$\text{Spec} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{2.5}$$

F_1 score F_1 score (F_1) represents the harmonic mean of precision and recall. Its key characteristic is the penalisation of extreme values in either precision or recall, ensuring a balanced consideration of both metrics. The F_1 score's value is influenced by the designated positive and negative classes, thereby it is asymmetric between these classes. In situations with a predominant positive class and a classifier biased towards it, the F_1 score tends to be high due to a high number of true positives. However, if the negative class becomes the majority and the classifier's bias shifts accordingly, the F_1 score may decrease, demonstrating its sensitivity to class definitions, despite constant data and class distribution. The range of the F_1 score is from 0 to 1, where 1 signifies maximum precision and recall, and 0 indicates either no precision or no recall.

The formula for the F_1 score is:

$$\text{F}_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}} \tag{2.6}$$

Matthews Correlation Coefficient Pearson's correlation coefficient, when applied to binary cases, transforms into Matthew's Correlation Coefficient (MCC). The MCC has gained popularity in machine learning, particularly for its effectiveness in imbalanced class situations. Essentially, it acts as a correlation coefficient between the true and predicted class labels, attaining high values only when the classifier performs well across all aspects of the CM. The range of the MCC is from -1 to 1, where 1 indicates perfect prediction accuracy, 0 indicates performance equivalent to random guessing, and -1 reflects complete discordance between prediction and observation.

The formula for the MCC is:

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}} \tag{2.7}$$

Additionally, an overall MCC for multi-class cases can be calculated as:

$$\text{Overall } MCC = \frac{c \cdot s - \sum_i^N p_i \cdot t_i}{\sqrt{s^2 - \sum_i^N p_i^2} \cdot \sqrt{s^2 - \sum_i^N t_i^2}} \tag{2.8}$$

In this equation, c represents the sum of products of corresponding elements of the CM, s is the sum of all elements in the CM, p_i and t_i are the sums of the rows and columns, respectively, of the CM. This formula allows a more nuanced understanding of classifier performance in scenarios with multiple classes.

Area Under the Receiver Operating Characteristic Curve This metric has been somewhat touched upon previously, as exemplified by Fig. 2.14, where the Receiver Operating Characteristic (ROC) curve was used as an evaluation metric. An ROC curve measures the performance of a classification model across various thresholds. It graphically plots two key metrics: True Positive Rate (TPR) and False Positive Rate (FPR).

The TPR, also known as recall, which has been previously defined, measures the proportion of actual positives correctly identified. On the other hand, the FPR is the ratio of false positives to the total actual negatives (i.e., 1− Specificity), and measures the likelihood of a false alarm. The FPR is defined as follows:

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{2.9}$$

Fig. 2.23 shows a standard ROC curve, it plots TPR versus FPR at various classification thresholds. Reducing the classification threshold results in more

instances being classified as positive, which consequently raises the counts of both FPs and TPs. It is inefficient to measure the model's performance many times with various classification thresholds. This is where the Area Under the Receiver Operating Characteristic Curve (AUROC) comes into play.

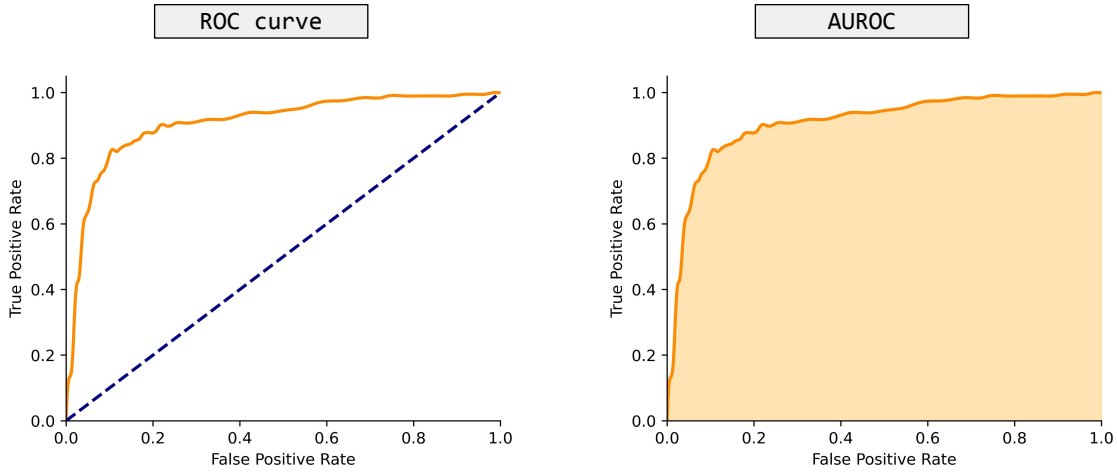


Figure 2.23: ROC curve and AUROC.

The AUROC, as the name suggests, calculates the total area underneath the ROC curve. It represents the probability that the model assigns a higher rank to a random positive sample than a random negative sample. A model that perfectly differentiates between the positive and negative classes will achieve an AUROC of 1. Conversely, a model with no better discriminatory ability than random guessing will have an AUROC of 0.5, and a AUROC of 0 indicates the model makes all predictions incorrectly. The AUROC is invariant to scale and classification threshold as it measures the ranking of predictions and quality of the predictions.

Area Under Precision-Recall Curve The Area Under Precision-Recall Curve (AUPRC) is a useful metric for evaluating performance on imbalanced data [135–139]. It computes the area under the Precision-Recall (PR) curve, akin to how AUROC operates with the ROC curve. The PR curve plots precision (the ratio of true positive predictions to all positive predictions) against recall (the ability of the classifier to find all the positive instances) at different thresholds. An example of PR curve is shown in Fig. 2.24. In scenarios with imbalanced data, AUPRC is often the preferred metric over AUROC as AUROC can present an overly optimistic view of model performance, especially when it poorly predicts the minority class [140, 141]. In contrast, AUPRC focuses on the positive class and provides a more realistic evaluation of a model's performance.

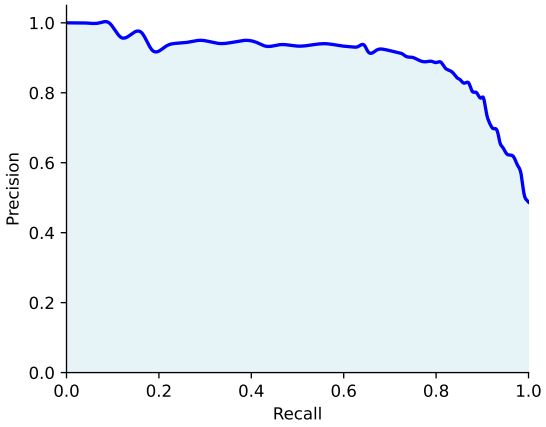


Figure 2.24: An example of the PR curve.

However, interpreting AUPRC is somewhat more complex compared to AUROC. For AUROC, the baseline is consistently 0.5. On the other hand, the baseline for AUPRC equals the ratio of positive samples in the dataset [142]. Thus, the AUPRC baseline varies across classes due to this dependency on the specific class distribution.

Although the varying baseline of AUPRC depends on the class distribution, this metric is favoured in imbalanced datasets precisely because of its sensitivity to the performance in the minority class. While it's true that the dependency on class distribution makes AUPRC a more complex metric to interpret, this characteristic enables it to more accurately reflect the challenges of predicting the minority class in imbalanced settings. In contrast to AUROC, which may mask performance issues in imbalanced datasets by averaging across all thresholds, AUPRC's focus on the positive class makes it a more informative metric when assessing the effectiveness of a model in these difficult scenarios. Therefore, while the interpretation of AUPRC requires consideration of the underlying class distribution, this also makes it particularly useful for evaluating model performance in the context of imbalanced data.

The AUPRC can be calculated by various methods, with the Average Precision (AP) [143] being one such method and the one selected for use in this thesis, as the use of linear interpolation to calculate AUPRC is inappropriate [141]. AP has been developed for binary classification but can be adapted to multi-class classification using a one-vs-all approach.

2.3 Chapter Summary

This chapter is a foundational bridge for interdisciplinary experiments presented in this thesis, specifically designed to address the knowledge gap between those unfamiliar with influenza and machine learning. The goal is to establish a solid understanding of both areas.

Given that the primary data in our experiments consist of protein sequence data, a comprehensive understanding of influenza, particularly its proteins and evolutionary mechanisms, is crucial. The chapter first dives into the foundation of influenza viruses. It also introduces the range of hosts susceptible to the virus. This exploration then extends to elucidate the mechanisms of antigenic drift and antigenic shift, which are responsible for the seasonal variability and potential pandemic outbreaks of influenza, respectively.

The chapter also delves into the realm of machine learning. It begins by defining what machine learning is and then explores various paradigms within the field. The discussion includes deep learning, the general workflow in machine learning projects, and specific algorithms that are particularly relevant to the nature of our data and research objectives. Methods such as nested k-fold Cross-validation (CV) and Bayesian Optimisation (BO) or the purposes of model evaluation and hyperparameter tuning, respectively. The chapter also discusses various evaluation metrics, which are pivotal in evaluating the efficacy of the employed machine learning models.

3

Sequence Data Description and Representations

In the scope of this thesis, protein sequence data is exclusively utilised in all our studies. To avoid repetitive explanations regarding the nature of these data in subsequent sections, this chapter lays out an introductory overview of protein sequences, underlining their differentiation from nucleotide sequences. Additionally, the methods of sequence representations will be briefly mentioned.

3.1 Nucleotide Sequence and Protein Sequence

Nucleotides are organic molecules that serve as the building blocks for nucleic acids like Deoxyribonucleic Acid (DNA) and Ribonucleic Acid (RNA). They comprise three primary components: a nitrogenous base, an aldopentose sugar, and phosphoric acid. The nitrogenous bases are classified into purines, which are *Adenine* (A) and *Guanine* (G), and pyrimidines, which include *Cytosine* (C) and *Thymine* (T) in DNA, and *Uracil* (U) in RNA.

In the case of influenza viruses, their genetic material is encoded in single-stranded RNA instead of double-stranded DNA, which is found in most living organisms. This viral RNA comprises chains of nucleotides, or polynucleotides, which whole genome sequencing has shown to consist of approximately 13,500 nucleotides that form the virus's genome [144]. These polynucleotide chains are crucial in encoding various proteins required for virus replication and function.

Proteins are large molecules essential for numerous biological functions. Each protein has a unique sequence of amino acids, which determines its specific role within the host. These functions are incredibly varied: proteins may act as enzymes to accelerate chemical reactions, as transporters to move substances across cellular

barriers, as antibodies to defend against pathogens, or as signalling molecules to coordinate biological processes. The sequence of amino acids in a protein and its subsequent three-dimensional structure defines its biological role, making proteins indispensable in the functioning and regulation of the body's tissues and organs.

The simplified relationship between DNA, RNA and protein is shown in Fig. 3.1. The coding systems used for nucleotides and amino acids are shown in Table 3.1. A typical nucleotide sequence is composed of four primary nucleotide codes: *Adenine* (A), *Cytosine* (C), *Guanine* (G), and *Thymine* (T) or *Uracil* (U) in RNA. These are the fundamental building blocks of DNA and RNA. In contrast, amino acids, the building blocks of proteins, are represented by 20 primary amino acid codes. Each amino acid has a corresponding single-letter code and a three-letter code. While nucleotide sequences generally use a more straightforward coding system, amino acids have a more diverse range due to the complexity of proteins.

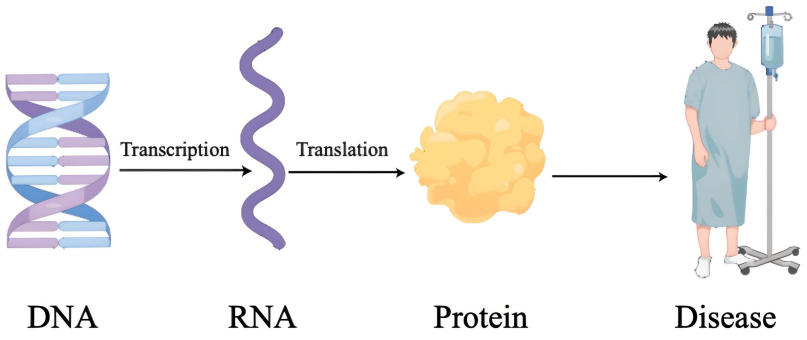


Figure 3.1: The simplified relationship between DNA, RNA, and protein, which depicts the general pathway of gene expression. The image was generated using `Figdraw` [96].

3.2 Sequence File Formats

3.2.1 Raw Format

The raw sequence data contain only a string of characters representing a sequence without any accompanying description or annotation. In this format, sequences are presented as continuous strings of nucleotide bases (e.g., A, T, C, G for DNA sequences) or amino acids (e.g., A, R, N, D for protein sequences). There are no line breaks, headers, or meta-information. The primary advantage of the raw format is its simplicity, making it straightforward for computational processing. However, it lacks contextual information that might be essential for understanding the source, function, or other relevant details of the sequence. As such, while raw formats are

Table 3.1: IUPAC nucleotide and amino acid codes [145, 146].

No.	Nucleotide Code	Base	Amino Acid Code	Three Letter Code	Amino Acid
1	A	Adenine	A	Ala	Alanine
2	C	Cytosine	C	Cys	Cysteine
3	G	Guanine	D	Asp	Aspartic Acid
4	T	Thymine	E	Glu	Glutamic Acid
5	U	Uracil	F	Phe	Phenylalanine
6	R	A or G	G	Gly	Glycine
7	Y	C or T	H	His	Histidine
8	S	G or C	I	Ile	Isoleucine
9	W	A or T	K	Lys	Lysine
10	K	G or T	L	Leu	Leucine
11	M	A, C	M	Met	Methionine
12	B	C, G or T	N	Asn	Asparagine
13	D	A, G or T	P	Pro	Proline
14	H	A, C or T	Q	Gln	Glutamine
15	V	A, C or G	R	Arg	Arginine
16	N	A, G, T or C	S	Ser	Serine
17	ȯr -	Gap	T	Thr	Threonine
18			V	Val	Valine
19			W	Trp	Tryptophan
20			Y	Tyr	Tyrosine
21			X	Xaa	Any amino acid
22			Z	Glx	Gln or Glu
23			B	Asx	Asp or Asn
24			-		Gap
25			. or *		Terminator

helpful for specific applications, they may not be ideal for comprehensive sequence analysis or sharing data across different platforms and researchers. An example of a sequence in raw format is shown in Fig. 3.2.

```
MKTTIILILLTHWVYSQNPTSGNNTATLCLGHHAVANGTLVKTITDDQIEVTNATELVQSISIGKICNNSYRVLDGRNCTLIDAMLGDPHCDDFQYENWDLFIER
SSAFSNCYPYDIPDHASLRSIVASSGTLEFTAEGFTWTGVTQNGGSGACKRGSADSFFSRLNWLTKSGNSYPILNVTMPNNKNFDKLYIWGIHHPSSNKEQTKLY
IQESGRVTVSTERSQQTVIPNIGSRPWVRGQSGRISIYWTIVKPGDILMINSNGNLVAPRGYFKLRTGESSVMRSDALIGTCVSECITPNGSIPNDKPFQNVNKV
TYGKCPKYIRQNTLKLATGMRNVPEKQIRGIFGAIAGFIENGWEGMVDGWYGFRYQNSEGTGQAADLKSTQAAIDQINGKLNRVIERTNEKFHQIEKEFSEVEGR
IQDLEKYVEDTKIDLWSYNAELLVALENQHTIDLTDAEMNKLFEKTRRQLRENAEDMGGGCFKIYHKCDNACIGSIRNGTYDHYIYRDEALNNRFQIKGVELKSG
YKDWILWISFAISCLLICVVLLGFIMWACQKGNIRCNICI
```

Figure 3.2: The example of hemagglutinin protein sequence (GenBank: AAB27733.1) for influenza A virus in raw data format.

3.2.2 FASTA Format

The FASTA format is one of the most commonly used formats for representing DNA, RNA, or protein sequences. It starts with a single-line description, known as the sequence header, which is typically denoted by a greater-than symbol (">") followed by the sequence identifier and an optional description. This is then followed by one or multiple lines of the sequence data. Each sequence in a FASTA file is represented by pairs of header and sequence lines. The sequences are written in

the same character codes representing nucleotides or amino acids. An example of a FASTA format sequence is shown in Fig. 3.3.

```
>AAB27733.1 hemagglutinin [Influenza A virus (A/equine/Taby/1991(H3N8))]
MKTTIILILLTHWVYSQNPTSGNNTATLCLGHHAVANGTLVKTITDDQIEVTNATELVQSISIGKICNNS
YRVLDGRNCTLIDAMLGDPHCDDFQYENWDLFIERSSAFSNCYPYDIPDHASLRSIVASSGTLEFTAEGF
TWTGVTQNGGSGACKRGSADSFFSRLNWLTKSGNSYPILNVTMPNNKNFDKLYIWGIHHPSSNKEQTKLY
IQESGRVTVSTERSQQTVIPNIGSRPWVRGQSGRISIYWTIVKPGDILMINSNGNLVAPRGYFKLRTGES
SVMRSDALIGTCVSECITPNGSIPNDKPFQNVNKVTYGKCPKYIRQNTLKLATGMRNVPEKQIRGIFGAI
AGFIENGWEGMVDGWYGFRYQNSEGTGQAADLKSTQAAIDQINGKLNRVIERTNEKFHQIEKEFSEVEGR
IQDLEKYVEDTKIDLWSYNAELLVALENQHTIDLTDAEMNKLFEKTRRQLRENAEDMGGGCFKIYHKCDN
ACIGSIRNGTYDHYIYRDEALNNRFQIKGVELKSGYKDWILWISFAISCLLICVVLLGFIMWACQKGNIR
CNICI
```

Figure 3.3: The example of hemagglutinin protein sequence from Influenza A virus strain A/equine/Taby/1991(H3N8) is shown in FASTA format with `AAB27733.1` as its GenBank identifier.

While the headers in FASTA format may vary across different databases, they typically include the name of the respective influenza virus strain. As illustrated in Fig. 3.3, the HA sequence from a non-human origin virus strain, labelled as A (`antigenic type`) / equine (`host of origin`) / Taby (`geographical origin`) / 1991 (`year of isolation`).

Ideally, the naming convention for influenza viruses should adhere to the internationally recognised format: `antigenic type / host of origin (specifically for non-human origin viruses) / geographical origin / strain number / year of isolation` [79, 147]. However, not every strain strictly follows this naming convention. Occasionally, some strain names might lack a strain number or use abbreviations for the place of isolation that are not immediately recognisable.

The FASTA format allows for easy readability and quick parsing of sequence data. FASTA files can contain a single sequence or multiple sequences, making them suitable for both individual sequence representation and datasets with various sequences. While the FASTA format offers a uniform way to present sequence data, it lacks capabilities for incorporating very detailed annotations or metadata.

In the context of this thesis, all data procured are presented in the FASTA format.

3.3 Sequence Representations

The raw sequence data, as depicted in Fig. 3.3, cannot be directly fed into models due to its non-numeric nature and the absence of a fixed structure required by many machine learning models. Therefore, before model training, it is essential to preprocess and transform these character-based sequences into numerical values that ensure that the underlying patterns within the data can be efficiently processed and learnt by these models.

In this section, various sequence representation methods commonly used in the field will be introduced, as well as those specifically employed in the studies presented in this thesis will be introduced.

3.3.1 Integer Encoding

Integer encoding, or ordinal encoding, assigns a unique integer value to each character, word, or categorical value. One benefit of using integer encoding is that it assigns a natural order to each element, making it apt for data with inherent ordinal relationships. Fig. 3.4 shows the integer encoding of the protein sequence from Fig. 3.3. For simplicity, only the first five amino acids (i.e., "MKTTL") are encoded in the example.

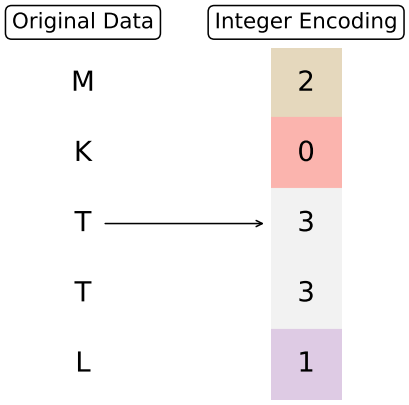


Figure 3.4: An example of integer encoding.

3.3.2 One-hot Encoding

For data without ordinal relationships, using integer encoding may result in poor performance and introduce noise. This occurs because the model might erroneously interpret the numeric values as having inherent ordinal relationships, leading to incorrect or suboptimal predictions. In such cases, one-hot encoding can serve as an alternative choice.

Fig. 3.5 shows the one-hot encoding of the first five amino acids from the protein sequence depicted in Fig. 3.3. Each amino acid is mapped to a binary vector, where the length of the vector corresponds to the number of unique amino acids in the data. Hence, the binary vector for each sequence has a size of $(M \times N)$, where M is the sequence length, and N is the number of unique amino acids present in the sequence. The position corresponding to the particular amino acid is marked

with a "1", while all other positions are set to "0". This method ensures a unique binary representation for every amino acid.

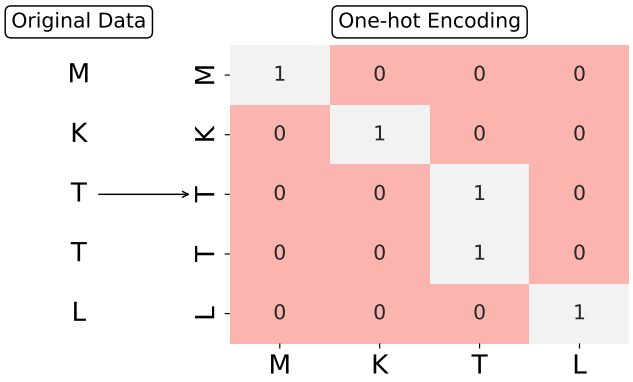


Figure 3.5: An example of one-hot encoding.

However, using one-hot encoding increases the dimensionality of the data, as each unique amino acid is represented by a binary vector whose length matches the total number of unique amino acids in the data. When working with large data, this can result in large and sparse matrices, thereby demanding more computational resources.

3.3.3 Biological Property-based Sequence Representation

We can use the physicochemical and structural characteristics of proteins, including properties such as hydrophobicity, molecular weight, electronegativity, charge, heat of formation, hardness, and electrophilicity, to transform protein sequences into numerical vectors that capture the inherent biological behaviours and functionalities of proteins. The commonly used relevant protein descriptors employed for this purpose are detailed in Table 3.2.

This encoding methodology delves deeply into the functional facets of proteins, harnessing their intrinsic properties to yield a biologically pertinent representation. This not only bolsters the precision and efficiency of diverse computational analyses but also manifests its indispensability in specialised tasks such as protein-protein interaction studies, protein structure prediction and function annotation, where a comprehensive understanding of the sequence's physicochemical context becomes paramount. For more details of practical applications and comparison studies of these encoding schemes, please refer to [155–157].

Table 3.2: Biological property-based protein descriptor.

Descriptor	Type	Dimension
AAC	Physicochemical	20
DPC	Physicochemical	400
TPC	Physicochemical	8000
VHSE [148]	Physicochemical	-
Z-scales [149]	Physicochemical	-
T-scales [150]	Topological	-
ST-scales [151]	Topological	-
BLOSUM62 [152, 153]	Physicochemical and substitution matrix	-
MS-WHIM [154]	3D electrostatic potential	-

3.3.4 Word Embedding

A protein sequence is composed of letters, each representing a specific amino acid, which is quite analogous to the structure of a text, with alphabets forming words. Hence, methodologies from Natural Language Processing (NLP) can be applied to protein sequences. In NLP, a commonly used technique is word embedding, which captures the semantic meaning of words by analysing their contextual usage and relationships within the text. Word embeddings are dense vector representations and can be adapted and applied to analyse and process protein sequences.

Several algorithms have been devised for generating word embeddings, and one intuitive method to adapt this concept for protein sequences involves integrating an embedding layer into a neural network. This layer functions analogously to word embeddings in NLP, considerably alleviating the necessity for laborious feature engineering. It autonomously learns to represent amino acids in a high-dimensional space, thereby capturing the underlying patterns and relationships within the data. Fig. 3.6 is an example of word embedding using the Word2Vec [28, 29], wherein each word is mapped to a dense vector in a high-dimensional space to encapsulate its semantic meaning, and words with similar semantic relationships are positioned closely together.

In our work, we have employed word embedding techniques in both Section 4.3.2 and Section 5.2.3.

3.3.5 Feature Derived from Pre-trained Models

Considering that training models on substantial datasets demand a considerable amount of time and computational resources, initiating the training process from scratch for every individual task can become impractically time-consuming, particu-

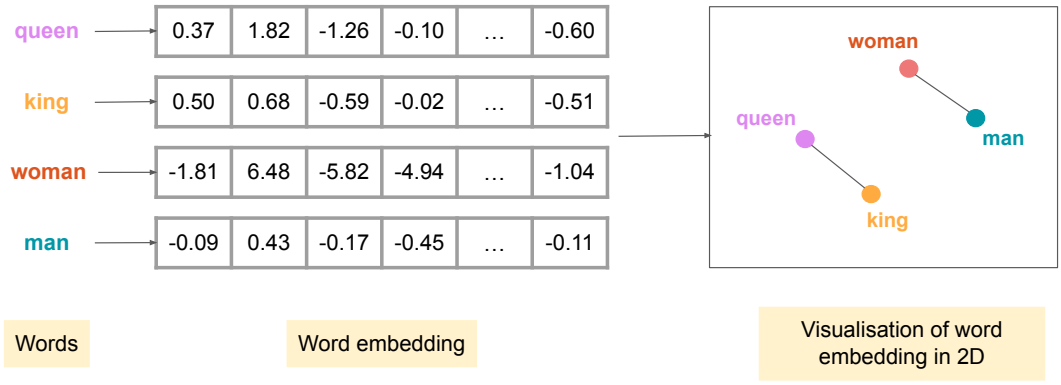


Figure 3.6: An example of word embedding using Global Vectors for Word Representation (GloVe) [158]: each word is represented by a vector in a 50-dimensional space, with a subset of these dimensions and their numerical values shown here to reflect the word's semantic and syntactic properties. These vectors, generated through the GloVe model, capture the relationships between words in a multi-dimensional space. The dimensionality reduction process is applied to project these high-dimensional vectors onto a 2D plane for visualisation, showing how related words cluster together (e.g., "man" is close to "woman", "king" to "queen").

larly when multiple tasks require the same training data. This redundancy could lead to unnecessary repetition and wasted resources in such scenarios.

To tackle the challenges and inefficiencies associated with model training, using pre-trained models has become a widely accepted practice. A pre-trained model is a saved network that was previously trained on a large dataset, resulting in saved parameters that capture crucial information about the underlying structure of the data. These models can be readily applied to a diverse range of tasks involving similar types of data through fine-tuning, even when the dataset for the new task is limited. As a result, using pre-trained models eliminates the need to train from scratch, leading to considerable time savings.

Beyond using the embedding layer, we also use features generated by pre-trained models in our work, as detailed in Section 5.2.3 and Section 6.2.2.

3.4 Chapter Summary

The focus of this chapter is to introduce sequence data and its various representations, a critical aspect of understanding the biological and computational elements of the research.

The chapter begins by elucidating the concepts of nucleotide and protein sequences, laying the groundwork for understanding the fundamental building blocks of the genetic and proteomic data central to the study. It then transitions

to discussing sequence file formats to highlight the differences between the raw format and the FASTA file format. Before feeding data into a machine learning model, it is necessary to preprocess the data, and different sequence file formats may require different preprocessing methods.

Protein sequence data cannot be directly fed into models. This chapter then delves into the different methods of sequence representation, an essential step in transforming biological data into a format amenable to computational analysis. The first method explored is integer encoding, where sequences are converted into numerical forms for computational convenience. Following this, the chapter explores one-hot encoding, a widely used representation in machine learning that transforms categorical data into a binary matrix.

Subsequently, the chapter introduces the biological property-based sequence representation. This method takes into account the specific properties of biological sequences, offering a more detailed and context-rich representation. Next, the chapter discusses word embedding techniques, which are used for capturing the contextual relationships within sequences.

Finally, the chapter presents the application of features derived from pre-trained models for encoding sequences. This advanced approach leverages models meticulously trained on extensive protein databases to extract complex features from sequence data, enabling more sophisticated analysis of biological information.

4

Predicting Influenza Host Origins through Bioinformatics and Machine Learning Approaches

4.1 Introduction

Influenza is a highly contagious respiratory illness that has consistently been at the forefront of scientific investigation due to its pandemic potential, posing significant health and economic challenges. One of the critical questions surrounding influenza is its host origins — understanding which species are responsible for specific strains can guide efforts in prevention, surveillance, and response. Traditional laboratory methods, although reliable, can be time-consuming and resource-intensive. This has led to a growing interest in employing computational techniques, especially as the volume of available genomic data continues to expand.

Only protein sequence data have been used in this thesis. As mentioned in Section 3.3, the representation of sequences is important because raw protein sequence data cannot be directly fed into machine learning models, and different data pre-processing methods may lead to varying performance outcomes on the same model. In this chapter, we delve into two primary approaches for handling sequence data based on the characterise of protein sequences: Position-Specific Scoring Matrix (PSSM)-based and word embedding techniques.

PSSM-based methods are primarily utilised to extract biologically relevant information, especially evolutionary features, from protein sequence data and serve as an input for traditional machine learningg models. In contrast, word embedding techniques are predominantly used in deep learning models to extract features that capture both the local context and the complex patterns within the protein

sequences, which are integral to the architectures of deep learning models. The size of a PSSM varies with the sequence length, making it essential to standardise its size. This chapter outlines three methods to standardise PSSM, with detailed explanations in Section 4.3.2. These three methods are:

- **EG-PSSM (Extended Grouped-PSSM)**: This method leverages the concept of residue grouping to manage the variability in sequence and PSSM lengths. Applying this grouping rule to each row of the grouped-PSSM (GPSSM), a 10×10 matrix is generated. This matrix is then restructured into a 1×100 feature vector, making it compatible for integration with classical machine learning models. The EG-PSSM method is instrumental in condensing the sequence information into a manageable format for traditional machine learning models, capturing the essence of the sequence in a compact representation.
- **GDPC-PSSM (Grouped Dipeptide Composition-PSSM)**: This approach extends the idea of dipeptide compositions to PSSMs. Each $L \times 10$ GPSSM is restructured into a 10×10 matrix, and then transformed into a 100-dimensional feature vector. The GDPC-PSSM extracts both amino acid composition and partial local-order information from protein sequences, offering a nuanced understanding of the sequence structure.
- **ER-PSSM (Extended Reduced-PSSM)**: Similar to GDPC-PSSM, this method extracts local sequence order information. However, it extends the computation of pseudo-compositions of dipeptides for amino acids in sequences with variable gaps. The resulting 91×10 matrix for each sequence is then converted into a 1×910 feature vector. ER-PSSM offers a comprehensive representation by considering the relationships between amino acids at various positions within the sequence.

The quality of data is a pivotal factor influencing model performance. It's common practice to cleanse the data during preprocessing, ensuring that only high-quality data is utilised for training models. However, this approach may not adequately prepare models for real-world scenarios, which often include a substantial amount of noisy data. Thus, the experiments detailed in this chapter adopt a dual-faceted approach. Alongside the high-quality data used for model training, data imbued with noise is also incorporated. This strategy is designed to reflect the model's potential performance more accurately in practical, real-life situations.

In addition to the aforementioned aspects, the results have been analysed comprehensively, casting light on the performance of our approach across diverse taxonomic levels and individual hosts. The analysis uncovered that a shallow Neural Network (NN) is adequate for processing influenza sequences. Moreover, 3-grams word embeddings surpass other sequence representations in performance. While PSSMs generation is time-consuming, opting for a smaller reference database does not significantly impact overall performance. It was also determined that a unified scheme for PSSMs is vital, considerably affecting model performance, with PSSM and word embedding-based sequence representations producing comparable outcomes.

Furthermore, Transformers with 5-grams inputs show superior performance and are least affected by incomplete sequences. The analysis acknowledges challenges in predicting sequences collected during pandemics, which underscores the model's potential in identifying strains with cross-transmission capabilities. For more detailed information about these results, please refer to Section 4.4.

4.2 Position-specific Scoring Matrix

One of the most commonly used methods to extract evolutionary information from protein sequences is the Position-Specific Scoring Matrix (PSSM) [159], which can be generated using Position-Specific Iterative Basic Local Alignment Search Tool (PSI-BLAST) [160]. PSSM encapsulates the pattern of sequence motifs across a collection of aligned sequences. PSSMs play a vital role in computational biology, particularly in identifying conserved motifs within a protein family and aiding in predicting proteins' secondary and tertiary structures.

A PSSM is generated from a Multiple Sequence Alignment (MSA) of various related sequences. It effectively measures the frequency of each amino acid's occurrence at specific positions in the sequence alignment. In the matrix, each column corresponds to a particular position in the sequence alignment. The values in the matrix, often derived from the frequency of each amino acid at each position and adjusted for background amino acid frequencies, represent the likelihood (score) of each amino acid occurring at each position.

The application of PSSMs extends to identifying and characterising protein domains and motifs, which are specific regions within proteins with particular structural or functional attributes. These motifs typically exhibit conserved sequences, and PSSMs are instrumental in predicting such conserved areas in different protein sequences.

Furthermore, PSSMs are employed in a variety of sequence analysis tools and

algorithms. For instance, in database search tools like the Basic Local Alignment Search Tool (BLAST), PSSMs enhance the sensitivity of sequence similarity searches by allowing for the identification of sequences sharing similar patterns or motifs.

In bioinformatics, PSSMs are highly valued for their ability to capture evolutionary information from amino acid sequences across protein families. They focus on evolutionary details, which are often more informative compared to physicochemical features derived solely from sequences [161, 162]. They are adept at extracting evolutionary information from protein sequences, which proves more informative for various predictive tasks. This includes identifying protein interactions and functionalities, thus making PSSM a superior choice for improving the accuracy of the predictions related to protein structure and function [163].

As protein sequences typically contain 20 different types of amino acids (i.e., A, R, ... V), the original PSSM, denoted as $\text{PSSM}_{\text{original}}$, is a $L \times 20$ matrix that consists of scores representing the substitution probabilities of amino acids across a query protein sequence with L length. The sequence is represented as $a_1 a_2 \dots a_L$, where a_1 corresponds to the amino acid type at the i-th position in a sequence of length L. An illustrative example of the $\text{PSSM}_{\text{original}}$ is provided below:

$$\text{PSSM}_{original} = \begin{pmatrix} & A & R & \dots & V \\ a_1 & p_{1,1} & p_{1,2} & \dots & p_{1,20} \\ a_2 & p_{2,1} & p_{2,2} & \dots & p_{2,20} \\ \dots & \dots & \dots & \dots & \dots \\ a_L & p_{L,1} & p_{L,2} & \dots & p_{L,20} \end{pmatrix}, \tag{4.1}$$

where $p_{i,j}$ is the score of the amino acid a_i that mutates to a_j. It can also be interpreted as a probability of mutation in the range of $[0, 1]$ using the sigmoid function:

$$p_{i,j} = \frac{1}{(1 + e^{-p_{i,j}})}, i = 1, 2, \dots, L; j = 1, 2, \dots 20, \tag{4.2}$$

4.3 Experiments

4.3.1 Data Collection

In this study, two datasets were used. The primary difference between the two sets is that Dataset 2 includes both complete and incomplete Hemagglutinin (HA) protein sequences, whereas Dataset 1 exclusively contains high-quality, complete sequences. Therefore, Dataset 2 is completely unforeseeable for models and noisier than Dataset 1. We used Dataset 2 as an additional testing set to test the performance of the

pre-trained model. The pre-trained models were trained and validated by Dataset 1.

Dataset 1

Dataset 1 includes the complete Influenza A Virus (IAV) HA protein sequences isolated from avian, swine, and human samples in the Global Initiative on Sharing All Influenza Data (GISAID) [164] database (status 2020-09-25). Only HA protein sequences are used in this study, given that HA is the most dominant protein for immunity response and helps the virus bind to target hosts [165]. To maintain the quality of the data, we further removed sequences that are either redundant, multi-label, or contain ambiguous amino acids X (any amino acid), B (Asp or Asn), and Z (Gln or Glu) [166]. Therefore, 59,785 sequences from the original set have been selected. Only Dataset 1 was subjected to nested k-fold Cross-validation (CV), as described in Section 2.2.6.

Dataset 2

Dataset 2 is an additional testing set. It includes the IAV HA protein sequences collected from 2020-09-26 to 2022-05-05 in the GISAID [164] database. Sequences present in both datasets 1 and 2 have been removed from Dataset 2 to ensure mutual exclusivity between the datasets. In addition, we have applied the same process as used in Dataset 1 to eliminate any redundant or multi-label sequences. However, in contrast to the approach taken with Dataset 1, Dataset 2 retains sequences that contain ambiguous amino acids. As a result, the sequences in Dataset 2 are also unique and have an unambiguous isolated host. The resulting final set consists of 3,686 sequences.

The change in the amount of data before and after filtering is presented in Table 4.1. Fig. 4.1 and Fig. 4.2 show the data distribution of different taxonomic levels. In terms of the performance of the model at a higher taxonomic level, we only consider three classes: human, avian, and swine, whereas at a lower classification level, avian data is divided further and results in 26 classes. When the number of classes increases, models face more challenges.

Table 4.1: Amount of data before and after selection.

Data Sets	Original #	Selected #	Only Complete?	NR[a]?	No Multi-label?	No X, B, Z?	Purposes[b]
1	180,833	59,785	✓	✓	✓	✓	Train, Val and Test
2	13,798	3,686		✓	✓		Test

[a] NR: non-redundant

[b] The purpose of the dataset: training, validation or testing.

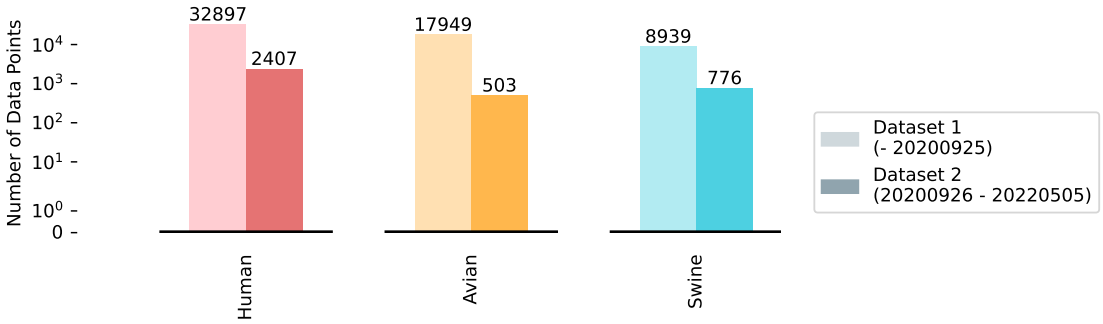


Figure 4.1: Data distribution (higher taxonomic level).

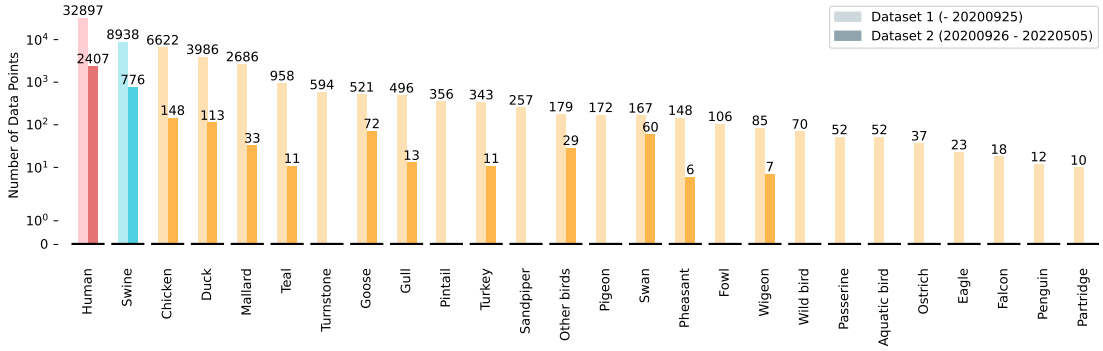


Figure 4.2: Data distribution (lower taxonomic level).

The original dataset contained redundant data. To address this, all identical sequences were removed, yielding a non-redundant dataset for this research. No additional sequence reduction steps aimed at removing redundancy were undertaken, considering the IAV's cross-species infection potential, which could lead to high similarity among sequences from different hosts. Consequently, omitting similar sequences would diminish the dataset's diversity and potentially skew the results, as illustrated in Fig. 4.3.

4.3.2 Sequence Representations

Position-Specific Scoring Matrix-Based Representations

We ran PSI-BLAST [169] iteratively with default parameters (E-value = 0.001, number of iterations= 3) on a partial NCBI's Non-Redundant Protein Sequence Database (NR) database. The NR database is a comprehensive collection of protein sequences from various sources, including GenBank, RefSeq, and PDB, which eliminates redundancies. Given the vast scope of the complete NR database, we opted for a subset tailored to ensure a focused analysis with reduced computational demands.

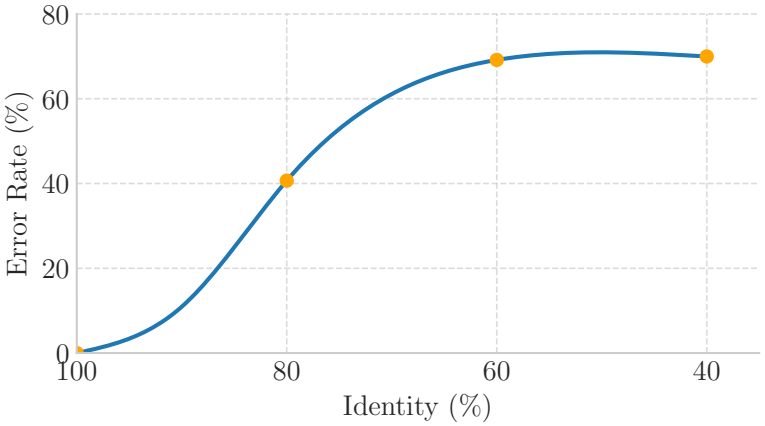


Figure 4.3: Error rate versus sequence identity for a set of aligned sequences. The plot shows the error rate as a function of sequence identity, with sequence identity ranging from 40% to 100%. The solid line represents a cubic spline fit to the data points, while the orange circles indicate the actual data points. The error rate for a set of clustered sequences is calculated as the total number of sequences with a different label than their corresponding representative sequence in their cluster, divided by the total number of sequences across all clusters. CD-HIT [167, 168] was used to reduce redundancy based on sequence similarity, wherein sequences were clustered together if they shared a certain percentage of similarity.

PSSMs cannot be fed directly into classic machine learning models due to their variable size. To overcome this hindrance, we propose three sequence encoding schemes based on PSSM. In order to reduce the complexity of proteins and unnecessary computations, we first introduce a residue grouping rule.

Residue Grouping Rule As amino acids have similar properties in proteins, they can be classified into 10 groups [170], which we have labelled for ease of reference: G_1 (F, **Y**, W), G_2 (M, **L**), G_3 (I, **V**), G_4 (A, T, **S**), G_5 (**N**, H), G_6 (Q, **E**, D), G_7 (R, **K**), G_8 (C), G_9 (G) and G_{10} (P). A grouped-PSSM (GPSSM) with $L \times 10$ dimensions can be created by applying residue grouping rules to the columns of the original PSSM (4.1).

$$\text{PSSM}_G = \begin{pmatrix} & G_1 & G_2 & \dots & G_{10} \\ a_1 & g_{1,1} & g_{1,2} & \dots & g_{1,10} \\ a_2 & g_{2,1} & g_{2,2} & \dots & g_{2,10} \\ \dots & \dots & \dots & \dots & \dots \\ a_L & g_{L,1} & g_{L,2} & \dots & g_{L,10} \end{pmatrix}, \tag{4.3}$$

where

$$g_{i,j} = \frac{\sum p_{i,G_j}}{|G_j|}, \tag{4.4}$$

The GPSSM is produced based on the original PSSM (4.1). Therefore, $\sum p_{i,G_j}$ represents the score of an amino acid a_i that is mutated to an amino acid belonging to j-th group G_j. L is the length of sequences; $i = 1, 2, \ldots, L$; $|G_j|$ is the number of amino acid types in G_j. The GPSSM (4.3) was used to derive the following proposed feature sets: EG-PSSM, GDPC-PSSM, and ER-PSSM.

EG-PSSM (Extended Grouped-PSSM) The length of the input sequence as well as the original PSSM (4.1) and GPSSM (4.3) may vary. This means they cannot be directly used in many machine learning models. An intuitive method for overcoming this problem is applying the residue grouping rule across rows of the GPSSM (4.3). This will result in a matrix of 10×10. We reformat the matrix by iterating through it row by row, moving from left to right within each row, thereby generating a 1×100 feature vector from a GPSSM (4.3) before feeding it to classical machine learning models:

$$\text{PSSM}_{EG} = \begin{pmatrix} E_{G_1,G_1} & E_{G_1,G_2} & \cdots & E_{G_{10},G_{10}} \end{pmatrix}^T, \tag{4.5}$$

where

$$E_{G_i,G_j} = \frac{\sum g_{G_i,j}}{|G_i|}, i, j = 1 \ldots 10, \tag{4.6}$$

GDPC-PSSM (Grouped Dipeptide Composition-PSSM) By using Dipeptide Composition (DPC) [171], we are able to determine amino acid composition information and partial local-order information in protein sequences. DPC acts directly on raw sequence data and generates a 400−dimensional feature vector for each sequence, representing the 400 dipeptide combinations from 20 standard amino acids. DPC can also be extended to PSSM [172]. Therefore, each $L \times 10$ GPSSM (4.3) can be rewritten as a 10×10 matrix by grouped dipeptide composition encoding. This 10×10 matrix can then be reshaped as a 100-dimensional feature vector:

$$\text{PSSM}_{GDPC} = \begin{pmatrix} D_{1,1} & D_{1,2} & \cdots & D_{10,10} \end{pmatrix}^T, \tag{4.7}$$

where

$$D_{i,j} = \frac{1}{L-1} \sum_{k=1}^{L-1} g_{k,i} \times g_{k+1,j} \; i, j = 1, 2, \ldots, 10, \tag{4.8}$$

Each $g_{k,i}$ is the value of row k and column i in the GPSSM (4.3).

ER-PSSM (Extended Reduced-PSSM) The third proposed sequence representation is adapted from RPSSM [173], which computes the pseudo-composition

of the dipeptide in sequences. As with GDPC-PSSM, RPSSM also extracts partial local sequence order information from sequences. RPSSMs only compute the pseudo-composition of any two adjacent amino acids. We extended the computation of RPSSM for any two amino acids $a_k a_{k+t}$ with gap t in sequences and extracted a 91×10 matrix per sequence. The matrix can be reformatted as a 1×910 feature vector:

$$\text{PSSM}_{ER} = \begin{pmatrix} M_{1,1,1} & M_{1,2,1} & \cdots & M_{10,10,9} & T_1 & \cdots & T_{10} \end{pmatrix}^T, \quad (4.9)$$

where

$$M_{i,j,t} = \frac{1}{L-t} \sum_{k=1}^{L-t} \frac{(g_{k,i} - g_{k+t,j})^2}{2}, \quad i, j = 1, 2, \ldots, 10; t = 1, 2, \ldots, 9 \quad (4.10)$$

and

$$T_i = \frac{1}{L} \sum_{k=1}^{L} \left(g_{k,i} - \bar{G}_i\right)^2, \quad i, j = 1, 2, \ldots, 10. \quad (4.11)$$

$\bar{G}_i$ is the average of values of GPSSM (4.3) in column i, T_i computes the average pseudo-composition of all the amino acids in the protein sequence corresponding to column i in GPSSM (4.3).

Learning Representations

Overlapping N-grams Protein sequences are morphologically similar to text sentences, except that text is composed of words, whereas amino acid letters comprise a protein sequence. In order to transform a protein sequence into a protein "sentence", we split the sequence into overlapping N-grams (N varies from 3 to 5). An N-gram is a protein "word" composed of successive N amino acids. Fig. 4.4 is an example of overlapping 3-grams for a protein sequence, and Fig. 4.5 shows the word clouds of trigrams for human, avian and swine.

seq: M L S I T I L F L ...
trigrams: MLS LSI SIT ITI TIL ILF LFL ...

Figure 4.4: Example of overlapping trigrams: the protein sequence MLSITILFL can be converted into a protein "sentence" containing 7 protein "words" MLS, LSI, SIT, ITI, TIL, ILF and LFL.

(a) Avian **(b)** Human **(c)** Swine

Figure 4.5: Word clouds of trigrams for each class were generated using MATLAB®, where the size of each trigram corresponds to its frequency of occurrence within the protein sequence data, with more common trigrams appearing larger in the cloud.

Word Embedding Word embedding overcomes the drawbacks of one-hot encoding. It is capable of producing dense vectors as well as capturing relationships between words. The idea behind word embedding is to map words to an embedding space, where words with similar meanings are closer together and, hence, have similar embeddings. Word2Vec [28, 29] is a popular implementation of word embedding, but it does not include domain-specific words. Thus, we generated a custom word embedding from the training set and mapped the N-grams of each sequence to the embedding vectors. An N-gram is represented as a vector of size N, and a protein sequence is represented as an $L \times N$, where L is the length of the sequence (i.e., the number of N-grams in the sequence) and N is the embedding dimension.

To unify the dimensions of matrices, we apply left-padding to the sequences with the longest sequence length (i.e., adding zeros or a predefined placeholder value to the beginning of sequences to match the length of the longest sequence in the dataset). Therefore, while most sequence information will be retained, more noise will be introduced to the shortest sequence.

4.3.3 Implementation and Evaluation

In this work, we employed a diverse set of traditional machine learning and deep learning algorithms, including Convolutional Neural Network (CNN), Multi-Layer Perceptron (MLP), Random Forest (RF), Random Undersampling Boosting (RUSBoost), Support Vector Machine (SVM), Transformer, and Extreme Gradient Boosting (XGBoost). A detailed description of each algorithm and its corresponding theory can be found in Section 2.2.5. In this section, we will focus primarily on discussing the parameter settings and architecture of the models employed in our experiments.

We designed a simple CNN for classifying protein sequences. The CNN in this

study consists of one input layer (*Input*), three convolution layers (*Conv*), three max-pooling layers (*Max-Pool*), one flattening layer (*Flatten*), three dense layers with Rectified Linear Unit (ReLU) activation (*Dense*) and one dense layer with softmax activation (*Output*). PSSM or tokenised virus sequences can be used as input data. The CNN also includes an embedding layer (*Embedding*) when the input data are tokenised sequences, as shown in Fig. 4.6. The hyperparameter settings for CNN can be seen in Table 4.2.

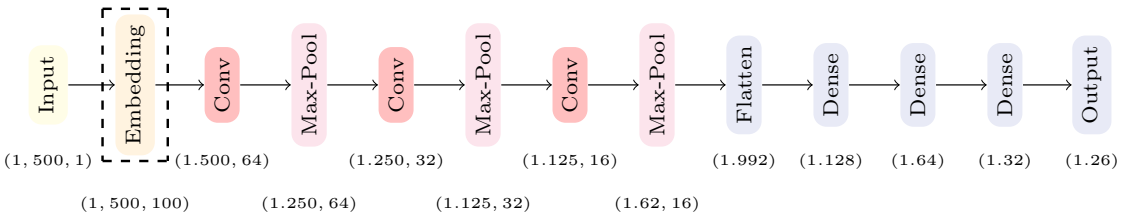


Figure 4.6: Example of the CNN architecture with an embedding layer: here, for simplicity, we assume that the longest length of the sequence is 500, the dimension of embedding is 100, and the number of filters for the first, second, and third convolutional layers is 64, 32, and 16, respectively. The layers are represented by coloured boxes: input layer (light yellow), embedding layer (light orange), convolutional layer (light red), max-pooling layer (light pink), and flatten/dense layer (light blue).

The sequence-sequence Transformer model usually includes an encoder block and a decoder block, which take a sequence as input and output a sequence, such as the sequence-to-sequence model often used in language translation. In contrast, a sequence-to-vector model accepts a sequence as input and outputs a class label for that sequence, and no decoder block is needed. We used a sequence-vector Transformer model, which accepts a sequence as input and outputs a class label for that sequence. Thus, the Transformer used in this study only has an encoder block. The Transformer architecture used in this study is shown in Fig. 4.7. The number of heads (*num_heads*) ranges from 1 to 5, and the embedding dimension (*embed_dim*) varies between 32, 64, and 128.

All models were evaluated using optimised parameters, as shown in Table 4.2. We used Bayesian Optimisation (BO) to adjust hyperparameters in 5 iterations automatically. RF, SVM and MLP were implemented by the `Scikit-learn` [174], RUSBoost was implemented by `imblearn` [175], XGBoost was implemented in `XGBoost Python package` [117]. CNN and Transformer mode were implemented by `Keras` [176].

In this study, we used nested k-fold Cross-validation (CV) with $k_{outer} = 5$ and $k_{inner} = 4$. Each outer fold, which comprises 20% of the total data, serves as the test set. The remaining 80% forms the training$_{\text{outer}}$ set, which is further divided into

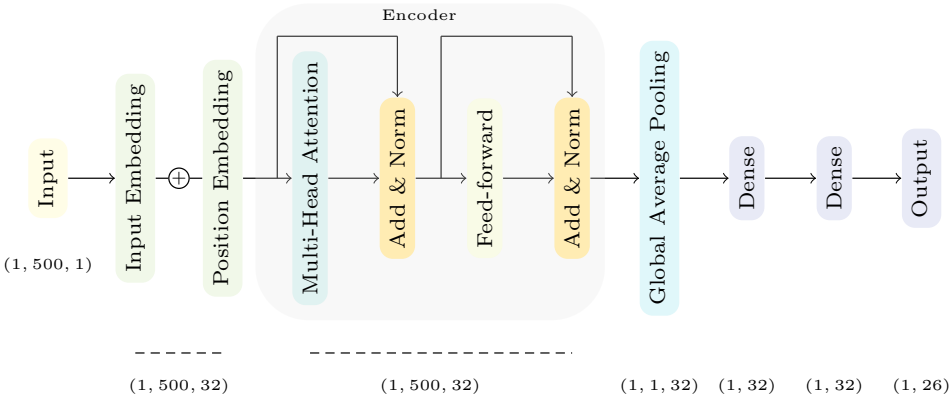


Figure 4.7: Example of the Transformer architecture: here, we assume that the number of heads is 3 and the dimension of embedding is 32.

Table 4.2: Hyperparameter settings.

Algorithms	**Hyperparameters**
SVM	C = 0.01, 0.1, 1, 10, 100 gamma = 0.001, 0.01, 0.1, 1
RF	n_estimators = 100, 200, 500, 1000, 1500, 2000 max_depth = 5, 10, 15, 20
RUSBoost	n_estimators = 50, 100, 200, 500, 1000, 1500, 2000 learning_rate = 0.001, 0.01, 0.1
XGBoost	max_depth = 5, 10, 15, 20 eta = 0.001, 0.01, 0.05 colsample_bytree = 0.1, 0.3, 0.5, 0.8, 1
MLP	alpha = 0.001, 0.01, 0.05 max_iter = 500 learning_rate_init = 0.001, 0.01, 0.05
CNN	num_filters = 64, 128, 256 learning_rate = 0.01, 0.05, 0.001, 0.0001 batch_size = 128 epochs = 300 kernel_size = 3
Transformer	embed_dim = 32, 64, 128 num_heads = 1, 2, 3, 4, 5 batch_size = 128 epochs = 300

four inner folds. In each inner loop, one fold (representing 20% of the total data or 25% of the training$_{\text{outer}}$ set) acts as the validation set, with the rest used for training. As a result, approximately 60%, 20%, and 20% of the data from Dataset 1 were used for training, validation, and testing purpose, respectively. The nested k-fold CV was applied exclusively to Dataset 1, as Dataset 2 only served as an additional test set.

Evaluation measurements used in the study include F_1 score (F_1), Area Under

Precision-Recall Curve (AUPRC) and Matthew's Correlation Coefficient (MCC). The equations of F_1 score and MCC for each class are defined as in Section 2.2.8. We also used the mean score to present the overall performance of each model. The mean score is defined as the mean of OAUPRC, OF_1 and OMCC:

$$\text{Mean Score} = average(OAUPRC, OF_1, OMCC) \tag{4.12}$$

where OAUPRC, OF_1 and OMCC represent the overall AUPRC, F_1, and MCC, respectively.

Regarding the models' performance in each class, we only show the results of AUPRC as AUPRC is recommended for evaluating classifiers when data is highly imbalanced [135].

4.4 Results and Discussions

In this study, we conducted various experiments to assess the performance of our models. This included analyses on different aspects such as comparing sequence representations, evaluating various machine learning models, analysing overall performance, examining performance across different hosts, investigating the impact of incomplete sequences, evaluating ensemble approaches, and considering the influence of the reference database used for generating Position-Specific Scoring Matrices (PSSMs).

Our findings indicate that a simple shallow Neural Network (NN) is sufficient for processing influenza sequences. We observed that 3-grams-based word embeddings tend to outperform other sequence representations in terms of performance. Although the generation of PSSMs is time-intensive, we found that utilising a smaller reference database does not significantly compromise overall performance. Furthermore, it is critical to implement a unified scheme for PSSMs generation, as this significantly affects model performance. We also discovered that PSSMs and word embedding-based sequence representations could achieve comparable performance. In terms of overall performance and resilience to incomplete sequences, Transformers configured with 5-grams inputs were superior. However, sequences collected during a pandemic presented challenges in accurate prediction. This indicates a difficulty in consistently identifying strains with the potential for cross-transmission. By considering the outputs from all models, we observed a potential for ensemble approaches to identify the most hard-to-predict sequences that all models struggle to predict accurately.

The following sections provide details of these findings.

4.4.1 Performance at Different Taxonomic Levels

We evaluated all models across varying taxonomic levels. The higher taxonomic level represented only three classes: avian, human, and swine. However, the avian class was further subdivided, and the dataset comprised 26 classes, making it challenging for models to classify hosts at the more refined taxonomic levels. In this section, we discussed the performance of sequence representations and machine learning algorithms across Dataset 1, as well as the overall results of each model.

Comparison of Sequence Representations

To encode protein sequences, we used two kinds of representation: sequence alignment-free (word embedding) and sequence alignment-based (PSSM-based representations). Table 4.3 compares sequence representation, where the metric scores are averaged across the respective machine learning algorithms applied for each representation. We seek to determine if certain representations universally boost algorithm performance or their effectiveness varies with specific algorithms, aiming to understand the adaptability and efficiency of sequence representations across different computational models.

At lower and higher taxonomical levels, 3-grams word embeddings reached mean scores of 87.73% and 97.94%, respectively. Additionally, as the N-gram size in word embedding increased from 3 to 5, a noticeable trend of declining performance emerged across all evaluated metrics at both higher and lower classification levels. This suggests that longer N-grams may be less effective in capturing context for Dataset 1.

In contrast, PSSM-based representations showed lower performance and higher variability compared to word embeddings, possibly because of their instability and poor performance on RUSBoost. When comparing PSSM-based representations, EG-PSSM had a relatively low deviation and a higher mean score.

The performance of all evaluated methods was consistently superior at the higher taxonomic level. In contrast, data became more skewed at the lower taxonomic level, with mean score drops for all sequence representations ranging from 10% to 15%. Among the various methods evaluated, ER-PSSM was the most affected, with a mean score drop of 14.85%, while 3-grams word embeddings were the least affected, with a mean score drop of 10.21%.

Table 4.3: Comparison of sequence representations on Dataset 1.

Classification Level	Higher Classification Level				Lower Classification Level			
Representations	Mean Score (%)	AUPRC (%)	F_1(%)	MCC (%)	Mean Score (%)	AUPRC (%)	F_1 (%)	MCC (%)
WE*(2-grams)	97.83 (0.19)	99.50 (0.07)	97.78 (0.19)	96.21 (0.32)	87.42 (0.74)	94.64 (0.42)	87.18 (0.71)	80.45 (1.09)
WE (3-grams)	**97.94 (0.16)**	**99.51 (0.05)**	**97.90 (0.16)**	**96.41 (0.29)**	**87.73 (0.56)**	**94.76 (0.35)**	**87.52 (0.55)**	**80.92 (0.81)**
WE (4-grams)	97.54 (0.36)	99.41 (0.09)	97.48 (0.36)	95.72 (0.61)	87.21 (0.47)	94.41 (0.35)	87.02 (0.43)	80.22 (0.65)
WE (5-grams)	97.32 (0.85)	99.36 (0.22)	97.26 (0.87)	95.34 (1.46)	86.56 (1.33)	93.97 (0.94)	86.41 (1.24)	79.30 (1.80)
EG-PSSM	92.52 (8.02)	96.02 (5.93)	92.93 (7.33)	88.62 (10.85)	79.30 (10.81)	87.78 (10.06)	79.97 (9.54)	70.16 (12.88)
ER-PSSM	89.77 (13.81)	94.15 (9.40)	90.20 (13.72)	84.95 (18.38)	75.42 (15.29)	85.42 (13.63)	76.15 (14.18)	64.68 (18.99)
GDPC-PSSM	85.13 (21.06)	90.96 (14.46)	85.50 (20.94)	78.92 (28.00)	70.28 (19.79)	82.15 (18.08)	72.90 (16.54)	55.77 (30.31)

* WE: Word Embedding

Comparison of Machine Learning Algorithms

Table 4.4 represents the comparison of machine learning algorithms with the averaged metric scores across sequence representations. RUSBoost performed the worst at both classification levels, but it maintained a narrower fluctuation in performance at the higher classification level compared to the lower one. Contrary to RUSBoost, SVM had a larger deviation under conditions of increased data skewness and class imbalance at the lower taxonomic level. Therefore, the performance of RUSBoost and SVM exhibited a dependency on the sequence representation, but RUSBoost was least affected by data skewness among all methods compared with SVM.

The Transformer and XGBoost performed best at the higher and lower taxonomic levels, respectively. Nevertheless, a decline in performance was observable for all classifiers when transitioning from the higher to the lower taxonomic level, with the mean score experiencing reductions between 9% and 30%. Among the evaluated algorithms, SVM was the most adversely affected, undergoing a substantial 26.1% drop in mean score. On the other hand, XGBoost appeared to be fairly stable, showing a moderate decrease in the mean score of 9.74%, which suggested it could manage difficult classification situations reasonably well.

Table 4.4: Comparison of machine learning algorithms on Dataset 1.

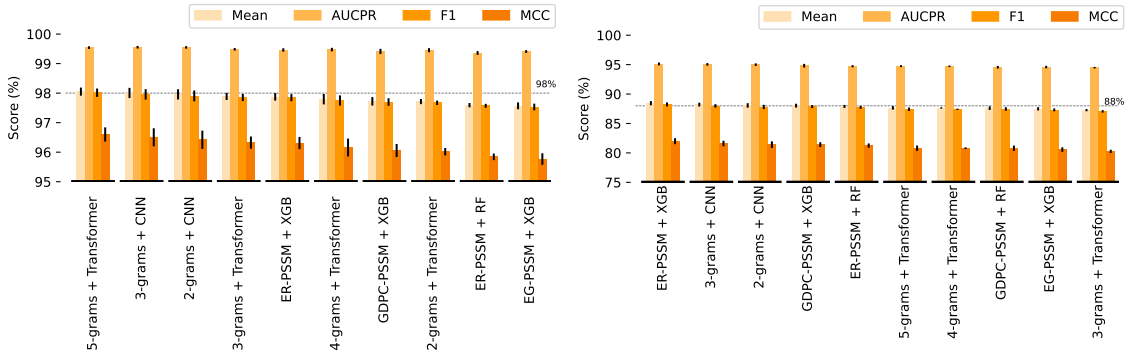
Classification Level	Higher Classification Level				Lower Classification Level			
Classifiers	Mean Score (%)	AUPRC (%)	F_1 (%)	MCC (%)	Mean Score (%)	AUPRC (%)	F_1(%)	MCC (%)
CNN	97.07 (0.74)	99.20 (0.31)	97.05 (0.71)	94.96 (1.22)	85.71 (2.01)	93.43 (1.31)	85.63 (1.87)	78.06 (2.86)
MLP	93.00 (0.97)	96.63 (0.85)	93.49 (0.77)	88.88 (1.33)	78.10 (1.64)	87.42 (1.61)	78.95 (1.28)	67.94 (2.08)
RF	97.41 (0.16)	99.31 (0.07)	97.38 (0.16)	95.52 (0.28)	87.57 (0.40)	94.53 (0.26)	87.43 (0.38)	80.75 (0.59)
RUSBoost	59.49 (17.92)	72.80 (11.56)	60.74 (18.93)	44.92 (23.79)	47.96 (7.27)	55.97 (11.12)	51.88 (6.60)	36.02 (5.85)
SVM	90.67 (3.78)	95.17 (2.33)	91.39 (3.51)	85.46 (5.55)	64.57 (15.01)	85.72 (3.63)	68.12 (11.21)	39.86 (30.68)
Transformer	**97.86 (0.18)**	**99.48 (0.06)**	**97.82 (0.18)**	**96.27 (0.32)**	87.33 (0.41)	94.55 (0.24)	87.11 (0.41)	80.34 (0.62)
XGBoost	97.72 (0.18)	99.42 (0.07)	97.69 (0.18)	96.04 (0.31)	**87.98 (0.52)**	**94.83 (0.34)**	**87.80 (0.49)**	**81.31 (0.75)**

Overall Results

Fig. 4.8 illustrates the top 10 models with the highest performance at different taxonomic levels. The results were ranked in descending order according to the mean score of each model, with the naming convention of the model being *sequence*

representation - machine learning algorithm. Most models reached at least a 96% mean score, particularly when the dataset comprised fewer classes, suggesting increased efficiency in less complex classification tasks.

Specifically, while the majority of the top ten models demonstrated consistent performance across different taxonomic levels, notable exceptions were observed in the case of the GDPC-PSSM-RF and 2-grams-Transformer models. The 2-grams-Transformer model excelled particularly at higher taxonomic levels. In contrast, the GDPC-PSSM-RF model tended to perform better at lower taxonomic levels. Among these models, ER-PSSM-XGBoost, 3-grams-CNN, 2-grams-CNN, and 5-grams-Transformer more effectively at both taxonomic levels, as indicated by their higher average rankings in comparison to other models.



(a) Higher Classification Level **(b)** Lower Classification Level

Figure 4.8: Performance of different models at different classification levels on Dataset 1.

4.4.2 Performance in Individual Hosts

Given that most machine learning algorithms are inclined to favour the majority class, it stands to reason that their performance would be better on the majority class compared to the minority class. In Dataset 1, 55% of sequences belonged to humans, while only 0.01% belonged to partridges. This degree of data imbalance brought great challenges to classifiers. ER-PSSM-XGBoost, 3-grams-CNN, and 5-grams-Transformer were the top three models that worked better, no matter how skewed the data was. Their AUPRC score in individual hosts of Dataset 1 was shown in Fig. 4.9.

All three models scored AUPRC below 80% in all hosts, except human, swine, and chicken, which accounted for approximately 81% of the sequences in Dataset 1. However, the baseline of the AUPRC for each class was the proportion of each class in the dataset. Therefore, human, swine, and chicken classes also had a relatively

higher baseline than other classes. The AUPRC and corresponding baseline for each class were shown in lime and green. These three models achieved higher scores than the baseline, but the variability increased with fewer classes.

Fig. 4.10 illustrates the performance of the three models in individual hosts at a lower taxonomic level for Dataset 2. The AUPRC of 5-grams-Transformer and 3-grams-CNN in each class still outperformed the baseline, even with the introduction of incomplete sequences, while ER-PSSM-XGBoost very slightly beat the baseline.

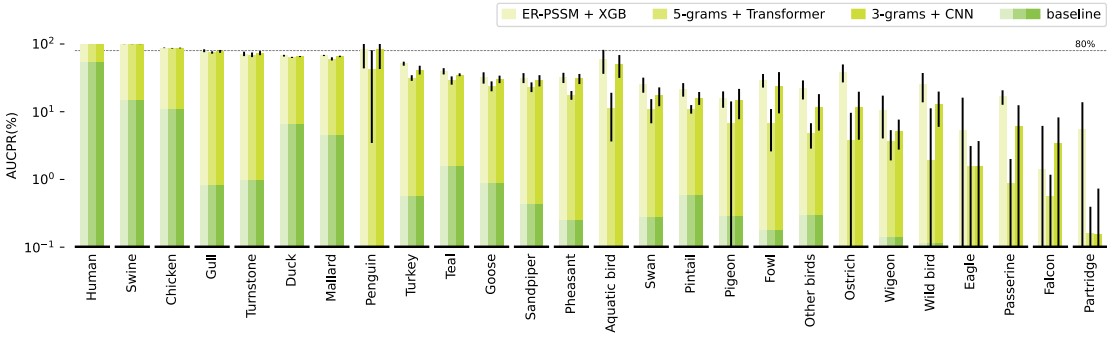
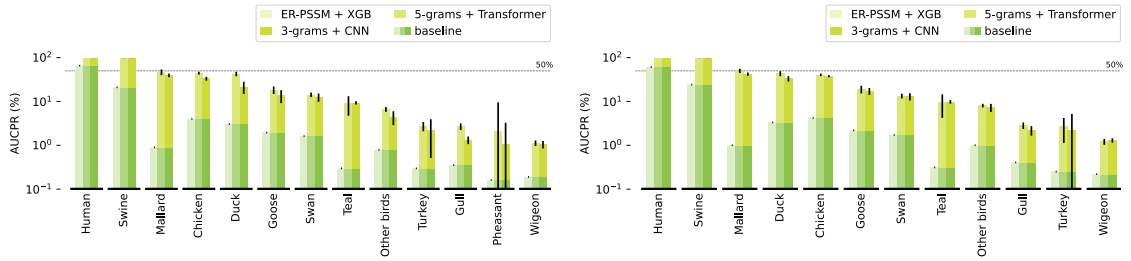


Figure 4.9: Performance of different models in individual hosts at a lower taxonomic level on Dataset 1.



(a) Include Incomplete Sequences **(b)** Exclude Incomplete Sequences

Figure 4.10: Performance of different models in individual hosts at a lower taxonomic level on Dataset 2.

4.4.3 Effect of Incomplete Sequences

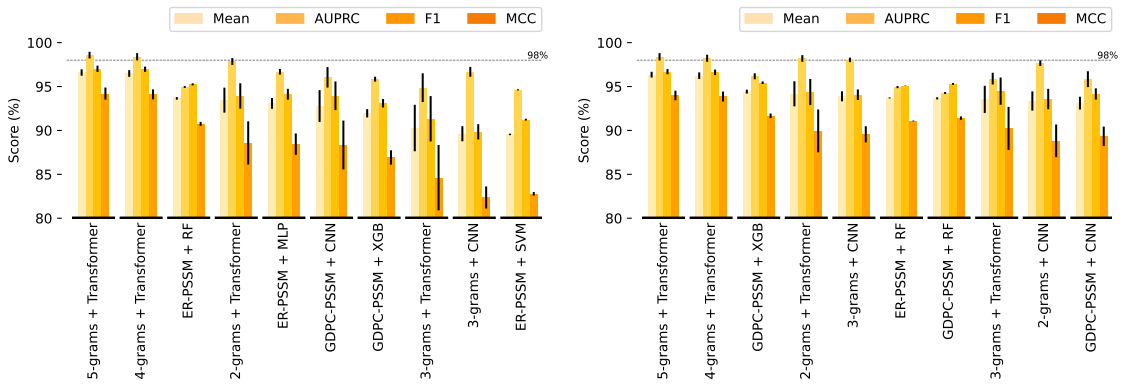
It's not uncommon to come across partial or incomplete sequence data in real-world scenarios. Instead of simply discarding such data, we're interested in exploring their potential for use. We sought to discover if our models can maintain strong performance on these less-than-ideal datasets, showcasing their adaptability and effectiveness in real-world contexts.

Reflecting on the composition of two data sets used in the study, Dataset 1 contained numerous selected HA sequences that we used to produce a pre-trained

model. Meanwhile, Dataset 2 contained 103 incomplete sequences, which we also used for evaluating the impact of incomplete sequences on models for Dataset 2, as results were illustrated in Fig. 4.11 and Fig. 4.12.

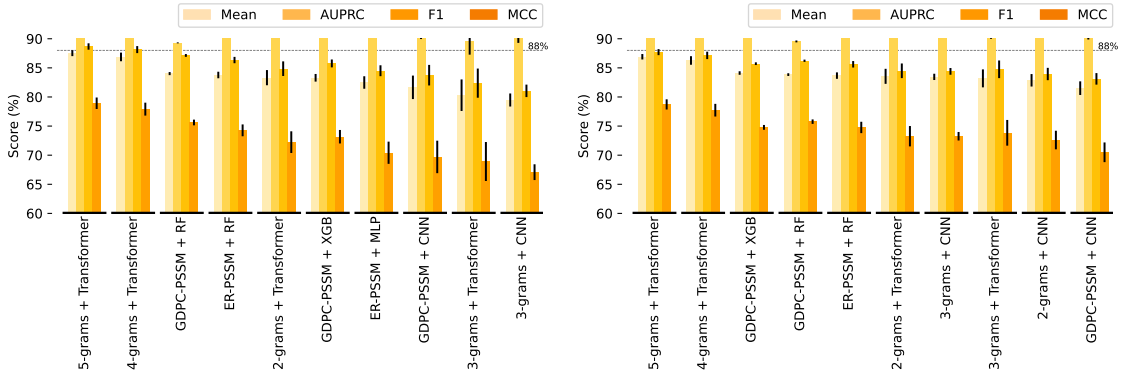
Among the various algorithms tested, the Transformer algorithm exhibited relatively stable performance across both taxonomic levels, showing a degree of resilience to incomplete sequences.

Although most models, including 3-grams-Transformer and 3-grams-CNN, exhibited a decline in performance and stability on Dataset 2 due to the addition of noisy data from incomplete sequences, the models still retained a degree of robustness. The influence of such data on their effectiveness was not excessively detrimental. This suggests a certain resilience of these models to less-than-ideal data conditions, a relevant factor for real-world applications.



(a) Include Incomplete Sequences **(b)** Exclude Incomplete Sequences

Figure 4.11: Performance of different models at a higher taxonomic level on Dataset 2.



(a) Include Incomplete Sequences **(b)** Exclude Incomplete Sequences

Figure 4.12: Performance of different models at a lower taxonomic level on Dataset 2.

4.4.4 Ensemble Results

Each algorithm naturally tends towards specific biases based on its design and underlying principles, and we sought to uncover the types of data that consistently elude correct prediction across all employed models. With this in mind, we assembled all the models for an ensemble approach, aiming to leverage their combined strengths and mitigate individual weaknesses. This ensemble generated predictions for Dataset 1, achieving a 97.56% match between the predicted and true labels, while 2.44% of the sequences were incorrectly predicted. More specifically, we found that 0.88% of the sequences across all models were mismatched with their true labels. To delve into which sequences were consistently "mispredicted" across all models, we conducted a basic statistical analysis.

Approximately 30% of these mispredicted sequences were collected from 2009 to 2011, a timeline that corresponds with the 2009 Influenza pandemic. With respect to their source, 57.52% of the sequences were derived from swine, while 40.76% were isolated from humans. As for their subtypes, 50.66% belonged to H1N1, 23.42% to H5N1, 8.57% to H9N2, and 4.00% to H5N6. We also generated the word clouds for these sequences, as presented in Fig. 4.13. The most frequent tokens revealed were LVL and GAI, which had also been the most frequent tokens in swine sequences, as shown in Fig. 4.5.

Figure 4.13: Word clouds of sequences that cannot be "correctly" predicted by all models, generated by MATLAB®.

The sequences that did not align with model predictions might have been collected during outbreak periods, such as pH1N1-like situations, or might have possessed strong cross-species characteristics. For example, *A/Beijing/1/2017* and *A/India/TCM2581/2019* were avian viruses isolated from humans [177, 178]; therefore, the label of these sequences was "human" while our prediction was "avian".

In a similar vein, *A/swine/Jangsu/48/2010* was a pH1N1-like swine virus used to prove retro-infection from swine back to humans in China [179]. Although it was labelled as "swine", our models predicted "human". This could have indicated that the models were picking up on the cross-species characteristics of these viruses, although they did not match the assigned labels.

4.4.5 Effect of Reference Database

In this section, we delved into how the choice of reference database influences PSSM-based models. The reference database used for BLAST could impact the quality of PSSM profiles as discussed in previous studies [163, 180], one primary limitation of PSSM that these matrices are derived from large datasets, which can be time-consuming. The time for generating the matrices will depend on the size of the reference database. Specifically, the duration required to generate these matrices is related to the size of the reference database.

Accordingly, we evaluated the performance of sequence-alignment-based models using PSSM generated from two distinct reference databases: the first part of NCBI's Non-Redundant Protein Sequence Database (NR) (i.e. nr.00) and Universal Protein Resource Database with at least 50% sequence identity (UniRef50). The general comparison between partial NR, complete NR, and UniRef50 was presented in Table. 4.5.

Table 4.5: Comparison for NR and UniRef50

Database	Size	# Sequences	Version
NR.00	26 GB	$>$ 7 million	2023-09-09
NR	200 GB	$>$ 600 million	2023-09-09
UniRef50	12 GB	$>$ 60 million	2023-06-28

The PSSM profiles generated by BLAST processing with UniRef50 were executed on the POSSUM [162] server, which took approximately 3 minutes per sequence. In relative terms, the number of sequences in nr.00 was only 9% of that in UniRef50. As a result, it became feasible to execute the processing on a local machine, which significantly decreased the processing time. For convenience, we referred to the PSSM generated by UniRef50 as UniRef50-PSSM, and the one generated by partial NR as PartialNR-PSSM.

The evaluation was also carried out at various taxonomic levels, with the results illustrated in Fig. 4.14 and Fig. 4.15. It was observed that most of the models exhibited consistent performance across different taxonomic levels, and no significant differences were discernible between the outcomes derived from either

of the reference databases for most of the models. This suggested that the models were fairly stable when dealing with variations in taxonomic levels and highlighted a level of equivalence between the two databases in terms of their contribution to the model's predictive accuracy.

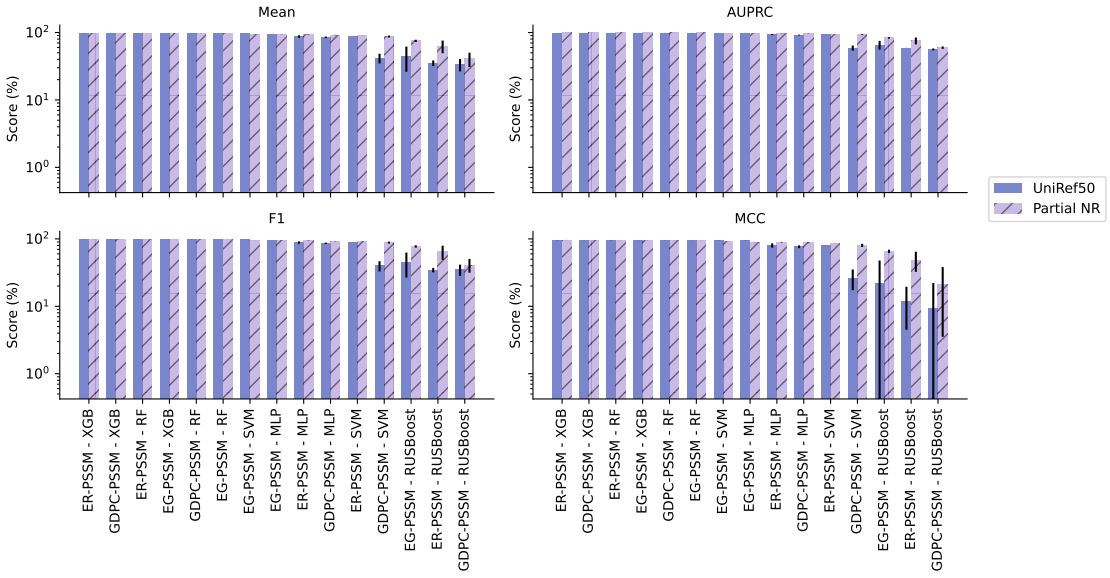


Figure 4.14: Comparison of model performance using UniRef50-PSSM and PartialNR-PSSM at the higher taxonomic level.

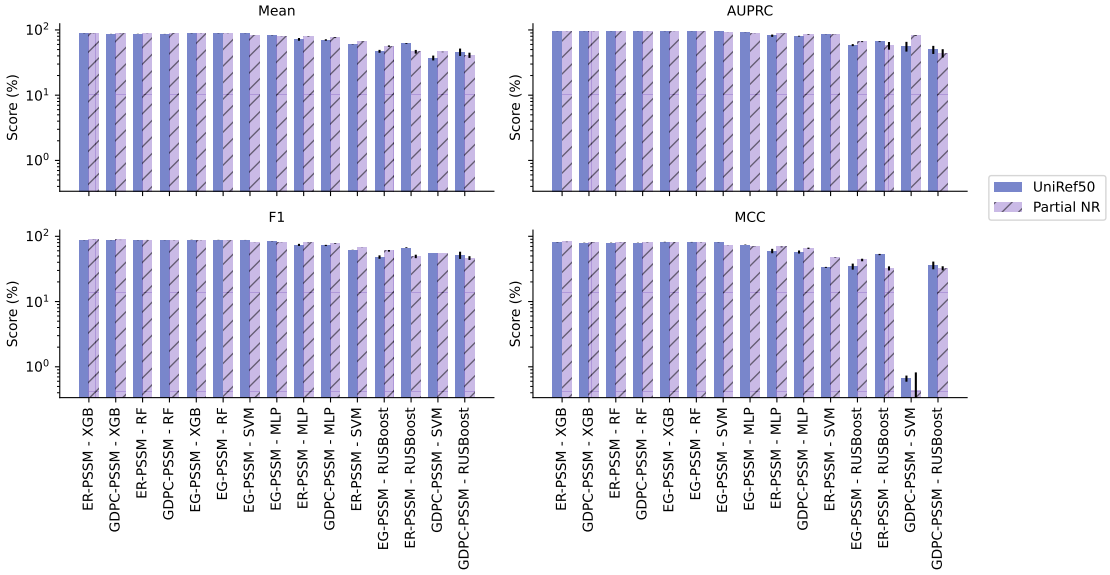


Figure 4.15: Comparison of model performance using UniRef50-PSSM and PartialNR-PSSM at the lower taxonomic level.

Nevertheless, RUSBoost-based models showed more pronounced variability across different reference databases at a higher taxonomic level while displaying

more consistency at lower taxonomic levels. ER-PSSM-XGBoost exhibited a slightly superior performance based on the mean score compared to other models. Among these, PartialNR-PSSM demonstrated superior performance compared to UniRef50-PSSM at a higher taxonomic level. Additionally, it was observed that the performance of GDPC-PSSM-SVM decreased as the number of classes increased. However, using the UniRef50-PSSM had been found to enhance its performance, showcasing its potential in improving model effectiveness under certain conditions.

Regarding the performance of models on incomplete sequences, we used the same test set (i.e. Dataset 2) as mentioned in Section 4.4.3, with the results illustrated from Fig. 4.16 to Fig. 4.19. UniRef50-PSSM had been observed to enhance

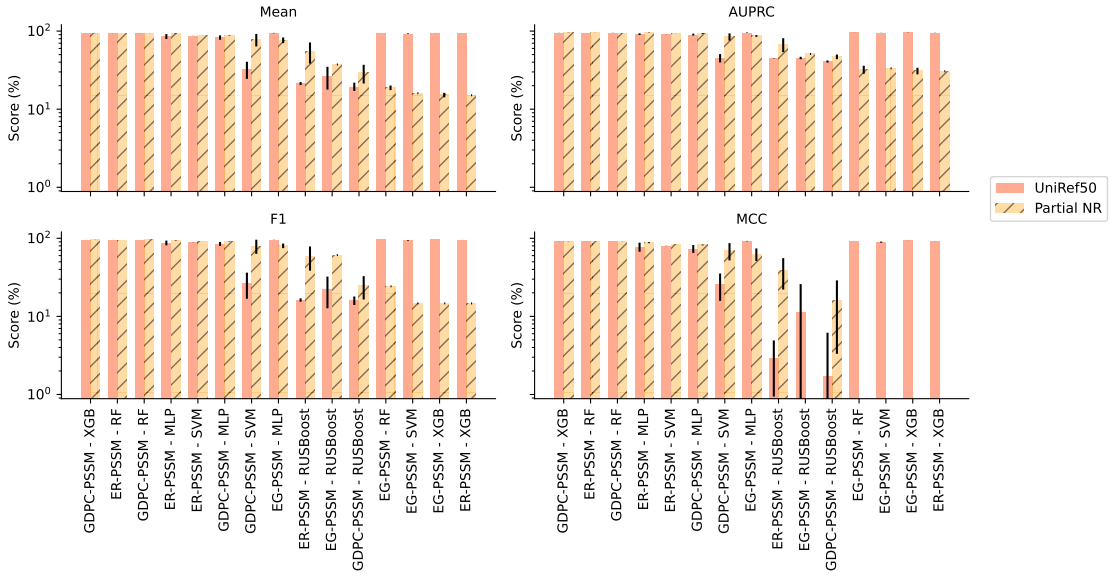


Figure 4.16: Comparison of model performance on Dataset 2 (excluding incomplete sequences) at the higher taxonomic level using UniRef50-PSSM and PartialNR-PSSM.

the performance of models to a noticeable extent, a trend that was particularly prominent in models such as EG-PSSM-SVM, EG-PSSM-XGBoost, EG-PSSM-RF, and ER-PSSM-XGBoost. This suggested that the UniRef50-PSSM database contributed positively to the predictive accuracy and efficiency of these specific models. Conversely, models including GDPC-PSSM-SVM, ER-PSSM-RUSBoost, and EG-PSSM-RUSBoost tended to perform better when using the PartialNR-PSSM, showcasing an inclination towards this particular reference database for optimal results.

When it came to handling incomplete sequences, specific models displayed some resilience. For instance, EG-PSSM-RF showed only a minor decrease in performance in the presence of incomplete sequences, a trend that was less pronounced when

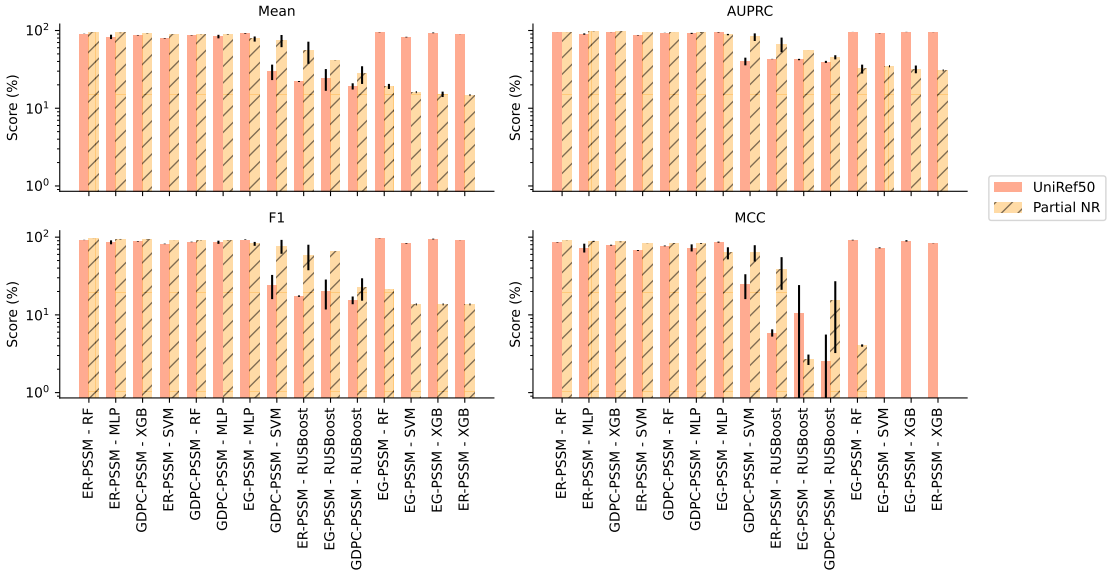


Figure 4.17: Comparison of model performance on Dataset 2 (including incomplete sequences) at the higher taxonomic level using UniRef50-PSSM and PartialNR-PSSM.

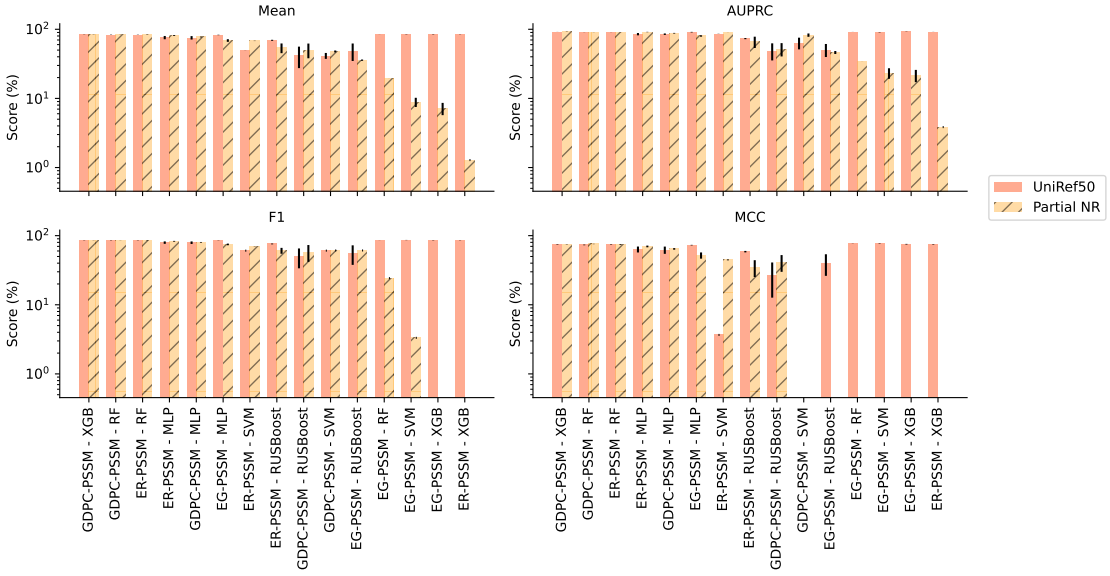


Figure 4.18: Comparison of model performance on Dataset 2 (excluding incomplete sequences) at the lower taxonomic level using UniRef50-PSSM and PartialNR-PSSM.

compared to other models. This variation underlined the resilience of some algorithms to handle noisy or incomplete data effectively.

Additionally, it was observed that the performance of models using UniRef50-PSSM improved as the number of classes increased. This trend highlighted the adaptability and efficiency of UniRef50-PSSM in managing more complex classification scenarios, where the diversity and quantity of classes presented a

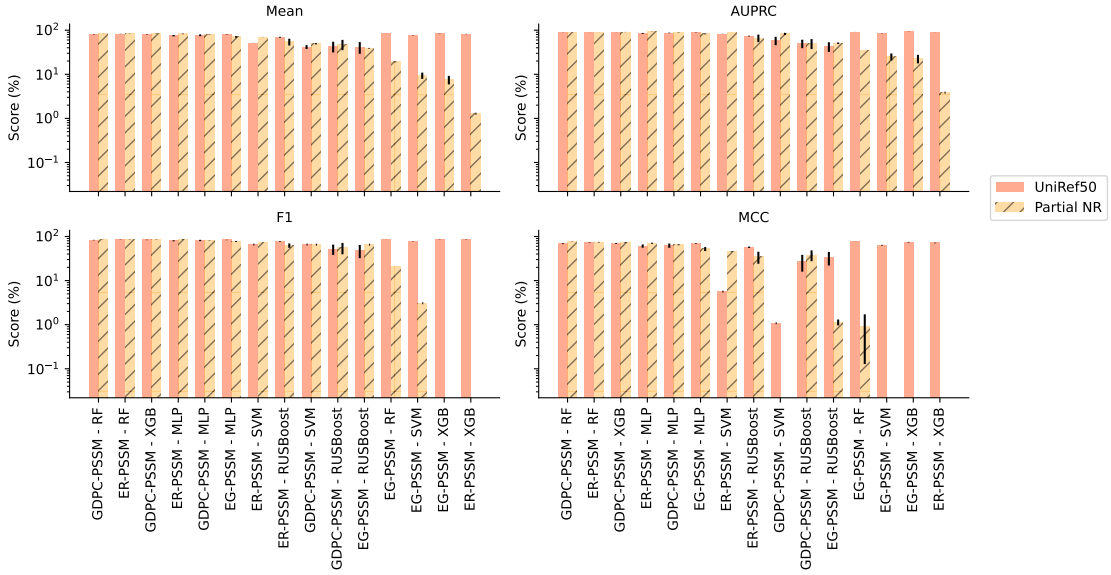


Figure 4.19: Comparison of model performance on Dataset 2 (including incomplete sequences) at the lower taxonomic level using UniRef50-PSSM and PartialNR-PSSM.

greater challenge.

In general, models tended to perform somewhat less effectively on data sets that incorporated incomplete sequences than those that did not. This suggests that incomplete sequences might have introduced challenges or noise during prediction. Furthermore, the performance variation observed across models using PSSMs generated by different reference databases indicated that the choice of reference database played a significant role in the model's predictive capabilities, and it was crucial to align the database selection with the specific model to ensure optimal performance.

4.5 Chapter Summary

The chapter focuses on predicting host origins of Influenza A Viruses (IAVs) through bioinformatics and Machine Learning (ML) approaches, addressing the significant health and economic challenges posed by influenza. It emphasises the importance of understanding the host origins of specific IAV strains to guide prevention, surveillance, and response efforts. Traditional laboratory methods for this purpose, while reliable, are noted to be time-consuming and resource-intensive. This has led to an increased interest in computational techniques, especially given the expanding volume of available genomic data.

Two primary approaches for handling sequence data are explored: Position-

Specific Scoring Matrix (PSSM)-based methods and Word Embedding (WE) techniques. PSSM-based methods are used to extract biologically relevant information from protein sequence data, serving as inputs for traditional machine learning models. In contrast, word embedding techniques are used in deep learning models to extract features that capture both the local context and intricate patterns within protein sequences.

The chapter discusses three methods for standardising the size of PSSMs due to the dependency of PSSM size on sequence length:

1. **EG-PSSM (Extended Grouped-PSSM)**: Utilises the concept of residue grouping to manage variability in sequence and PSSM lengths, condensing sequence information into a manageable format for traditional machine learning models.

2. **GDPC-PSSM (Grouped Dipeptide Composition-PSSM)**: Extends the idea of dipeptide compositions to PSSMs, extracting both amino acid composition and partial local-order information from protein sequences.

3. **ER-PSSM (Extended Reduced-PSSM)**: Focuses on extracting local sequence order information, extending the computation of pseudo-compositions of dipeptides for amino acids in sequences with variable gaps.

The chapter emphasises the importance of data quality, pointing out that models trained solely on high-quality data may not be adequately prepared for real-world scenarios often filled with noisy data. Therefore, the experiments incorporate high-quality and noisy data, aiming to reflect the model's potential performance in real-life situations accurately.

The results of these methods are analysed comprehensively, shedding light on the performance across different taxonomic levels and individual hosts, and factors like incomplete sequences and the choice of reference database are considered. The integration of ensemble results is also discussed, providing a holistic perspective on the efficacy of combining multiple models. For a brief summary of the results from these extensive analyses, please refer to Table 4.6 and Table 4.7.

Table 4.6: Summary of key findings and analysis in Chapter 4.

Chapter/Section	Key Findings
Sequence Representations	- 3-grams-based WEs generally perform the best. - PSSM and WE-based representations can have comparable performance, subject to the choice of N-grams for WE and the unified schema for PSSM.
ML Models	- Simple shallow NN effective for IAV sequences. - Transformers with 5-grams inputs show superior performance in general.
Performance Across Different Hosts	- Data skewness impacts performance. - ER-PSSM-XGBoost, 3-grams-CNN and 5-grams-Transformer work better. - 3-grams-CNN and 5-grams-Transformer show resilience to incomplete sequences.
Impact of Incomplete Sequences	- Transformers show resilience. - Performance generally decreases slightly with incomplete sequences.
Ensemble Approaches	- Ensemble methods help identify hard-to-predict sequences, improving overall accuracy. - Achieved a 97.56% match between predicted and true labels. - Identified sequences commonly mispredicted across models primarily originate from the pandemic.
Reference Database Influence	- Choice of database can affect PSSM-based models. - Smaller reference databases (e.g., PartialNR-PSSM) do not significantly compromise performance. - Certain models perform better with specific reference databases.

Table 4.7: Summary of performance at different taxonomic levels.

Feature	Higher Taxonomic Level	Lower Taxonomic Level
Best Performing Sequence Representation	3-grams WEs (Mean Score: 97.94%)	3-grams WEs (Mean Score: 87.73%)
Worst Performing Sequence Representation	GDPC-PSSM (Mean Score: 85.13%)	GDPC-PSSM (Mean Score: 70.28%)
Performance Trend with N-grams Size	Decrease in performance with increase from 3 to 5-grams.	
Best Performing ML Algorithm	Transformer (Mean Score: 97.86%)	XGBoost (Mean Score: 87.98%)
Worst Performing ML Algorithm	RUSBoost (Mean Score: 59.49%)	RUSBoost (Mean Score: 47.96%)
Most Stable Algorithm across Levels	XGBoost (Moderate decrease in performance at a higher level)	
Best Performing ML	ER-PSSM-XGBoost, 3-grams-CNN, 2-grams-CNN, 5-grams-Transformer	

5

An End-to-End Multi-Channel Neural Network Approach for Predicting Influenza A Virus Hosts and Antigenic Types

5.1 Introduction

Influenza A Viruses (IAVs) can be transmitted across species. Therefore, employing machine learning to predict the host of the influenza virus can provide insights into which strain has the capacity for cross-species infection. Furthermore, precise prediction of subtypes is not only crucial for early detection and management of influenza outbreaks but also instrumental in identifying emerging novel subtypes. This enhanced understanding of subtype dynamics is crucial for public health planning and can significantly contribute to more effective outbreak control and mitigation efforts.

The process of training individual models for each distinct task associated with influenza virus prediction is not only time-consuming but also demands significant resources. Addressing this challenge, this chapter presents an end-to-end Multi-Channel Neural Network (MC-NN) tailored for predicting the hosts and antigenic types of IAVs. Rather than relying on traditional methods, this approach harnesses the power of deep learning, tapping into the depth and breadth of information hidden in sequence data.

In line with the experiments detailed in the previous chapter, this chapter continues to utilise protein sequence data. However, we've broadened our sources to include multiple databases, namely Influenza Research Database (IRD) and Global Initiative on Sharing All Influenza Data (GISAID), and have also included Neuraminidase (NA) sequence data in our analysis. After thoroughly filtering, we've

curated a final dataset consisting of 46,172 unique pairs of Hemagglutinin (HA) and NA sequences. This dataset has been divided into two periods: pre-2020 and 2020-2022, and it includes a subset of incomplete sequences as well.

The model is trained on the pre-2020 set, while the 2020-2022 set is used to simulate the real-world scenario, evaluating the model's prediction stability when new data is added. In contrast to the previous chapter, where incomplete and complete sequences were mixed, we separated incomplete sequences into a separate set. This allows for a better evaluation of the impact of incomplete sequences on the model's performance.

In this chapter, we refine the classification of each sequence's host and delve into the predictive capabilities of using features from pre-trained models for influenza virus tasks, focusing on host and subtype predictions. This chapter shifts focus away from Position-Specific Scoring Matrix (PSSM)-based representations towards the exploration of features derived from pre-trained models alongside custom word embeddings. Custom word embeddings for protein sequences are typically generated by training models on specific datasets, which requires the construction of tailored vocabularies and the initialisation of embeddings with random values. In contrast, pre-trained models leverage embeddings derived from extensive training on large, diverse text corpora, so they can be fine-tuned to better suit specific tasks without the need to build dictionaries from scratch. Therefore, we aimed to assess whether these advanced methods could enhance model performance or offer new insights compared to traditional approaches.

Our experimental setup employed a MC-NN architecture that processes both HA and NA sequences to predict host, HA, and NA labels. However, not every influenza strain has pairs of H/N sequences in real-life scenarios. Therefore, in addition to the regular evaluation, the model performance is evaluated under scenarios with only HA or NA sequence inputs.

The findings revealed that Convolutional Neural Network (CNN) and Transformer models demonstrated superior overall performance, showing that these architectures can effectively handle the complexity of influenza virus prediction tasks. Additionally, custom word embeddings were as effective as features generated by pre-trained models, suggesting that carefully tailored embeddings can match the sophistication of widely used pre-trained models. Furthermore, when faced with incomplete input data, the Transformer model exhibited exceptional performance with HA-only sequences, while the Bidirectional Gated Recurrent Unit (BiGRU) model was particularly adept at handling NA-only inputs.

This exploration underscores the potential of leveraging advanced Neural Network

(NN) architectures and using pre-trained model features to improve the accuracy and efficiency of IAVs' prediction. It opens up avenues for future research to further refine these methods and explore their applicability to other virology-related prediction tasks.

5.2 Experiments

5.2.1 Data Collection

The Hemagglutinin (HA) and Neuraminidase (NA) sequences were acquired from two sources: the Influenza Research Database (IRD) [181][1] and the Global Initiative on Sharing All Influenza Data (GISAID) [84]. The initial data collection process yielded 381,369 HA sequences and 338,631 NA sequences (downloaded on 13 December 2022). To maintain the uniqueness of each strain, redundant and multi-label sequences were filtered, resulting in a unique HA and HA sequence pair for each strain in the final dataset. To prevent duplicates, the integration process involved removing GISAID sequences if they were already present in IRD. Additionally, strains belonging to the H0N0 subtype, which have an uncleaved HA0 protein that is non-infectious, were also removed from the dataset. The HA0 protein is a precursor form of the HA protein, and it must undergo cleavage into its active forms, HA1 and HA2, for the virus to become infectious. Removing these strains with uncleaved HA0 proteins ensures that only infectious strains are included in the dataset, improving the data quality for our analysis.

The data curation process also included the removal of sequences with erroneous or ambiguous metadata labels, such as the case with `A/American Pelican/Kansas/W22-200/2022` (GISAID Isolated ID: `EP_ISL_14937098`), which was inaccurately labelled as "host". The class labels for each sequence primarily rely on the metadata extracted from their corresponding FASTA files. In FASTA files, the metadata is usually indicated by the information following the ">" symbol, as explained in Section 3.2.2. Subsequently, the final result comprised 46,172 unique pairs of complete and partial HA and NA sequences.

The sequence was considered complete if its length was equivalent to the actual genomic sequence [84] or it covered the complete coding region as defined by the National Center for Biotechnology Information (NCBI) [182]. Completeness annotation cannot be explicitly obtained from strain metadata. Therefore,

[1]The IRD is now known as the Bacterial and Viral Bioinformatics Resource Center (BV-BRC), available at `https://www.bv-brc.org`.

incomplete sequences were obtained by filtering complete sequences from the entire influenza database, which comprises complete and incomplete sequences ($all\ sequences = complete\ sequences \cup incomplete\ sequences$).

The pre-trained model was trained using a training dataset comprising sequences of strains isolated before 2020. On the contrary, the strain sequences isolated from 2020 to 2022 were used solely to evaluate the performance of the models during testing, which also incorporated incomplete sequences. The characteristics of the datasets used in this study are presented in Table 5.1.

Table 5.1: Summary statistics of datasets.

Dataset (*alias*)	# Total Pairs	# Seqs from IRD	# Seqs from GISAID
$<$ 2020 (*pre-20*)	33,159	41,940	24,378
2020 - 2022 (*post-20*)	4,488	3,232	5,744
Incomplete (*incomplete*)	8,525	11,111	5,939

5.2.2 Label Reassignment

While the GISAID and IRD databases recorded more than 300 hosts, only 30% were consistent between both databases, potentially due to the mixed use of common and scientific names. To address this inconsistency, we manually assign hard labels, regrouping viral hosts into 25 categories based primarily on the classification of the biological family of the animals. The distribution of reassigned hosts is presented in Fig. 5.1. We reclassified certain subtypes with insufficient data points in the dataset under a broader "other" category, in order to facilitate more robust Cross-validation (CV) (i.e., H15, H17, H18, N10, and N11), as shown in Fig. 5.2.

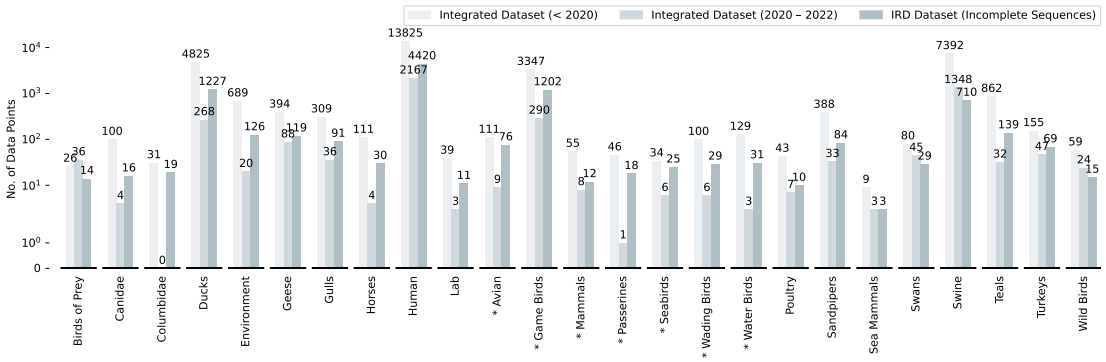


Figure 5.1: Distribution of data based on hosts.

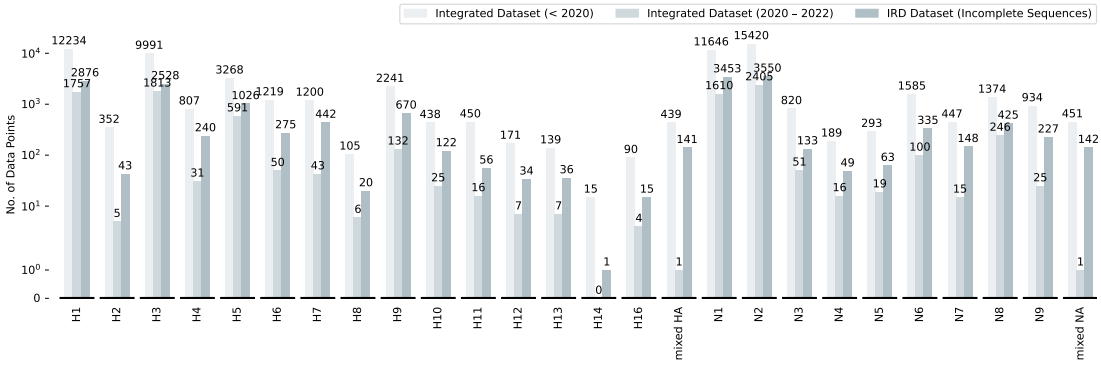


Figure 5.2: Distribution of data based on subtypes.

5.2.3 Sequence Representations

In this study, we used the word embedding to encode the protein sequences, as detailed in Section 4.3.2. Additionally, we applied four pre-trained language models for generating protein sequence embeddings: ESM-2 [183], ProtBert [184], ProtT5 [184], and ProtVec [185]. We will not elaborate on word embedding further here as it has been covered in the aforementioned section, and the focus of this part is to highlight the introduction of the pre-trained models implemented in our research.

ESM-2

ESM-2 [183] is a Transformer-based protein language model trained using masked language modelling, forming a part of the Evolutionary Scale Modeling (ESM) series. ESM-2 aims to learn the evolutionary patterns in proteins only using protein sequences, and it offers a diverse range of model sizes, ranging from smaller configurations with 8 million parameters to extensive, highly complex versions comprising up to 15 billion parameters.

ESM-2 uses the Transformer architecture, incorporating its attention mechanism to facilitate a comprehensive understanding of amino acid relationships within protein sequences. Trained using masked language modelling objectives, ESM-2 adeptly learns dependencies between amino acids, culminating in the formation of attention patterns that correlate with residue-residue contact maps. As a result, ESM-2 is capable of developing attention patterns that correlate with residue-residue contact maps, ultimately reflecting the protein's tertiary structure and providing valuable insights into its three-dimensional conformation. ESM-2 was exposed to approximately 65 million unique protein sequences throughout its training phase, ensuring a thorough and well-rounded training process for the model.

ESM-2 surpasses many single-sequence protein language models in structure prediction tasks and effectively predicts protein structure, function, and other characteristics directly from individual sequences. It is adaptable for fine-tuning a variety of tasks involving protein sequences.

In this study, we opted for the `esm2_t30_150M_UR50D` checkpoint of ESM-2 from Hugging Face, which contains 30 layers and a total of 150M parameters. For ease of reference, we have abbreviated the name of this particular pre-trained model to ESM2-30. More details of the model's training procedure can be referred to [183, 186, 187].

ProtBert

ProtBert [184] is based on Bidirectional Encoder Representations from Transformers (BERT), akin to ESM-2, built upon the Transformer architecture. These BERT-based models are particularly effective due to their deeply bidirectional nature, considering bidirectional context across all layers, resulting in robust representations. ProtBert is tailored to process uppercase amino acid sequences. It distinguishes itself by interpreting each sequence as a discrete document, dispensing with the next-sentence prediction component inherent in traditional BERT models.

During its training phase, ProtBert also employed a masked language modelling objective similar to the original BERT; it randomly masked 15% of amino acids in the input sequences. Of these, 80% were replaced with the [MASK] token, 10% with a random amino acid, and the remaining 10% were left unchanged. This approach enables the model to effectively infer patterns from the unlabelled data, which is a massive corpus from the UniRef100 dataset, including 217 million protein sequences. The sequences were tokenised by spaces and limited to a vocabulary of 21 unique terms, thereby generalising less common amino acids such as "U, Z, O, B" to "X". For efficient processing, sequences were batched by length, classified into either shorter than 512 or under 2,048 amino acids, and treated as independent units, streamlining the pre-training process.

ProtBert can generate embeddings from unlabelled sequences that capture essential biophysical characteristics relevant to protein structure. The resulting insights reflect a profound grasp of protein "language". Although ProtBert excels at feature extraction from protein sequences, it reaches its full potential when fine-tuned for specific applications. More details of the model's training procedure can be referred to [184, 188].

ProtT5

ProtT5 is a protein language model based on Text-to-Text Transfer Transformer (T5), designed specifically for protein-related applications. T5 is a transformer model that has restructured all NLP tasks into a text-to-text format. Therefore, when the T5-based model is applied to proteins, it can be used for tasks such as predicting protein-protein interactions or the effects of mutations by treating them as text-to-text problems. Unlike the original T5-3B, which employed a span denoising objective, ProtT5 was pre-trained with a BERT-like masked language modelling denoising objective. This training maintained the traditional T5 strategy of randomly masking 15% of amino acids in the input sequences.

This model was rigorously pre-trained in a self-supervised manner on UniRef50, a dataset encompassing 45 million protein sequences without any human-annotated labels. This automated process allowed for the extensive use of publicly available data, generating inputs and labels directly from the raw protein sequences. During its training, ProtT5 adopted methodologies similar to ProtBert, processing protein sequences that are uppercased, tokenised using spaces, and using a specialised vocabulary of 21 symbols, with infrequent amino acids "U, Z, O, B" remapped to "X". Sequences were truncated or padded to 512 tokens to fit the model architecture. Furthermore, the masked language modelling objective was implemented by masking 15% of the amino acids: 90% of the time, these were replaced with a [MASK] token, and 10% of the time, with a random different amino acid.

For our research, we employed a more compact and computationally efficient version of ProtT5 (prot_t5_xl_half_uniref50-enc), which operates solely with the encoder part of the original ProtT5 (ProtT5-XL-UniRef50) model and is configured to run in half-precision (float16) mode. This configuration is optimised for generating protein and amino acid representations with reduced demand on GPU resources but matching the full model's capabilities on downstream tasks without compromising performance. More details of the model's training procedure can be referred to [184].

ProtVec

ProtVec [185] distinguishes itself by offering k-mer based vector embeddings for proteins. In contrast to ESM-2, ProtBert, and ProtT5, which leverage the Transformer architecture and masked language modelling for their models, ProtVec is inspired by Word2Vec, which represents words in a continuous vector space. ProtVec analogously treats amino acid subsequences, or k-mers, as the basic elements similar to words, thereby embedding protein sequences into a continuous vector space.

ProtVec was developed using an extensive training set from the Swiss-Prot database, encompassing 549,790 sequences. The training process harnesses the power of the Skip-gram neural network model, which is designed to optimise the probability of predicting the context within sequences of words—or, in this case, amino acids. By leveraging three lists of shifted non-overlapping words, a technique also applied in other studies within this thesis (Section 4.3.2 and Section 5.2.3), ProtVec effectively groups biologically similar words, which share physical and chemical properties, in close proximity within the vector space. Upon evaluation using k-Nearest Neighbors (kNN) with a two-fold CV approach, a window size of three has been identified as optimal.

ProtVec's strength lies in its ability to detect and encapsulate local patterns within protein sequences into compact vector representations. These embeddings are adept at retaining some of the local contextual information surrounding each tripeptide, providing valuable insights for various predictive tasks. More details of the model's training procedure can be referred to [185].

Table 5.2 shows the comparison of pre-trained protein language models with their respective embedding dimensions, granularity, and training datasets.

Table 5.2: Comparison of pre-trained protein language models.

Pre-trained Models	Dimension	Granularity	Training Set
ESM-2 (30)	640	Amino Acid	UniRef50 (sample UniRef90)
ProtBert	1,024	Amino Acid	UniRef100
ProtT5	1,024	Amino Acid	UniRef50
ProtVec	100	Tripeptides	Swiss-Prot

5.2.4 Implementation and Evaluation

In this work, we used Convolutional Neural Network (CNN), Bidirectional Gated Recurrent Unit (BiGRU) and Transformer. A detailed description of each algorithm and its corresponding theory can be found in Section 2.2.5. In this section, we will focus on discussing the parameter settings and architecture of the models applied in our experiments.

All the models in this study were built using `Keras` [176] and trained on pre-20 datasets. They were then tested in both post-20 and incomplete datasets. The architecture of the Multi-Channel Neural Network (MC-NN) used in this study is illustrated in Fig. 5.3. The Transformer architecture used here is the encoder presented in [123].

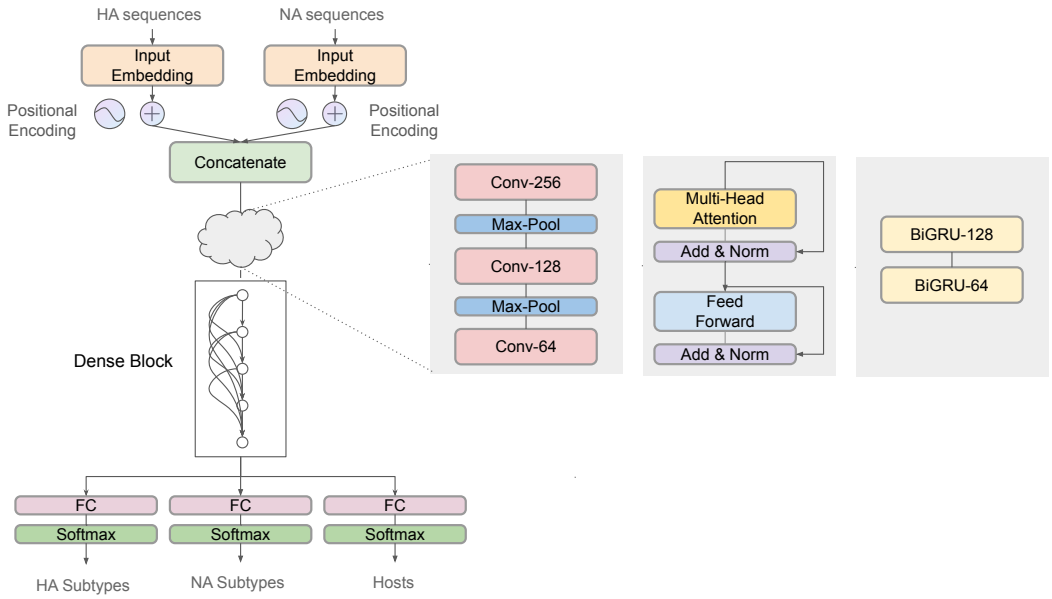


Figure 5.3: The multi-channel neural network architecture: positional encoding is only employed along with the Transformer.

We used nested k-fold Cross-validation (CV), and the outer fold k_{outer} was set to 5, and the inner fold k_{inner} was set to 4. The details about nested k-fold CV can be found in Section 2.2.6. The hyperparameters settings for the Neural Network (NN) architectures used in this study are presented in Table 5.3.

Table 5.3: Hyperparameter settings.

Models	Hyperparameters
CNN	kernel size = 3, 4, 5 embedding size = 50, 100, 150, 200 learning rate = 0.01, 0.005, 0.001, 0.0001
BiGRU	embedding size = 50, 100, 150, 200 learning rate = 0.01, 0.005, 0.001, 0.0001
Transformer	embedding size = 32, 64, 128 learning rate = 0.01, 0.005, 0.001, 0.0001 num heads = 1, 2, 3, 4, 5

The overall performance of the models was evaluated by comparing their results to those from the Basic Local Alignment Search Tool (BLAST) [169, 189, 190], which was used with its default settings. BLAST is a widely used sequence analysis tool in computational biology and bioinformatics. F_1 score (F_1) and Area Under Precision-Recall Curve (AUPRC) were selected as the evaluation metrics for this study.

5.3 Results and Discussions

In this study, we conducted various experiments to evaluate the performance of our models. These evaluations included assessments of overall performance, performance on single protein sequence inputs, and performance across different feature sets.

Our findings indicate that both CNN and Transformer models exhibit superior overall performance. Custom word embeddings are found to have performance comparable to that of features generated by pre-trained models. When only partial inputs are provided, the Transformer model demonstrates exceptional performance with HA-only inputs. Conversely, the BiGRU model shows enhanced performance with NA-only inputs.

For detailed insights into these findings, please refer to the subsequent sections.

5.3.1 Overall Performance

The performance of the model in various datasets was shown from Fig.5.4 to Fig.5.6. The BLAST results were obtained through 5-fold CV and were indicated by the solid black line in the figures.

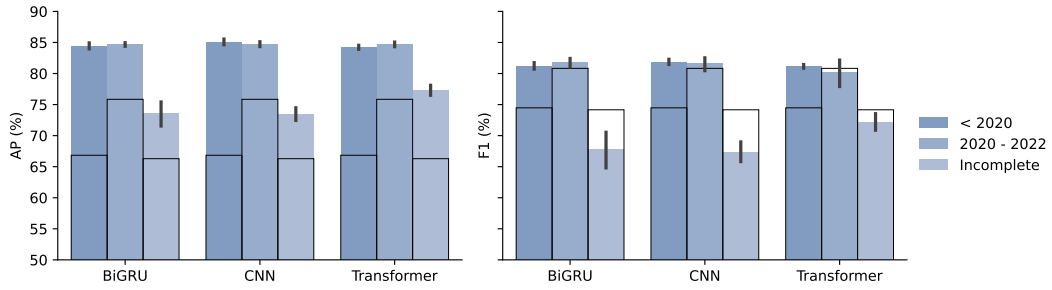


Figure 5.4: Comparison of overall performance between models for host prediction tasks: the baseline results with BLAST are highlighted with a solid black outline.

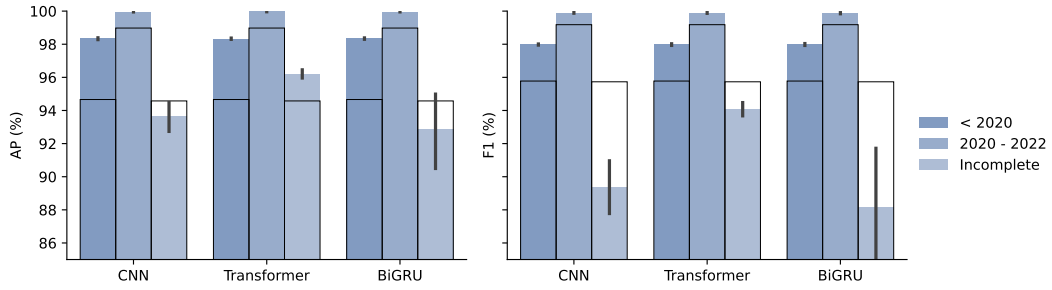


Figure 5.5: Comparison of overall performance between models for HA subtype prediction task: the baseline results with BLAST are highlighted with a solid black outline.

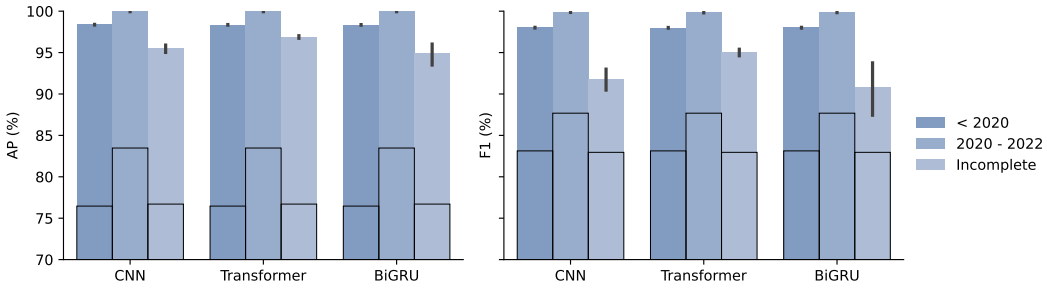


Figure 5.6: Comparison of overall performance between models for NA subtype prediction task: the baseline results with BLAST are highlighted with a solid black outline.

The models were trained exclusively on the pre-20 dataset (labelled as "*< 2020*") and were tested on the post-20 (labelled as "*2020 - 2022*") and incomplete (labelled as "*incomplete*") datasets. The pre-20 and post-20 datasets only contained complete sequences, while the incomplete dataset included incomplete sequences, as detailed in Table 5.1.

In evaluating the models on the pre-20 dataset, it was evident that each surpassed the baseline performance. However, the differences in their respective performances were subtle, with negligible differences observed. The best-performing model across all prediction tasks was the MC-CNN, which achieved an average AP of 93.92% (93.62%, 94.21%), and an average F_1 score of 92.62% (92.38%, 92.85%).

When we evaluated the models on the post-20 dataset, we observed a general improvement in performance compared to the pre-20 dataset. MC-BiGRU exhibited a 1.43% rise in its average F_1 score, while MC-Transformer exhibited a 1.23% increase in its average AP score. Nevertheless, MC-CNN was still the top performer across all prediction tasks, achieving an average AP of 94.88% (94.76%, 94.99%) and an average F_1 score of 93.81% (93.41%, 94.21%).

The incomplete dataset brought about its challenges, with a discernible decline in performance across all models, and the MC-BiGRU showed increased variation. The MC-Transformer model achieved superior performance, with an average AP of 90.13% (89.87%, 90.39%), and an average F_1 score of 87.09% (86.49%, 87.69%). Additionally, the Transformer also had the most consistent performance across all datasets and all prediction tasks. We also observed that the task of predicting the host across all models was significantly more complex compared to predicting the subtype.

5.3.2 Overall Performance on Single Sequence Input

The proposed MC-NN took two inputs, but obtaining paired HA and NA for every strain wasn't always feasible in practical scenarios. To align with this reality, we

conducted additional experiments on two distinct test sets: the first comprised 23,802 HA protein sequences (labelled as "*HA only*"), and the second contained 5,142 NA protein sequences (labelled as "*NA only*"). The models were still trained using the pre-20 dataset and were tested on these datasets. The results of these experiments were presented from Fig.5.7 to Fig.5.9.

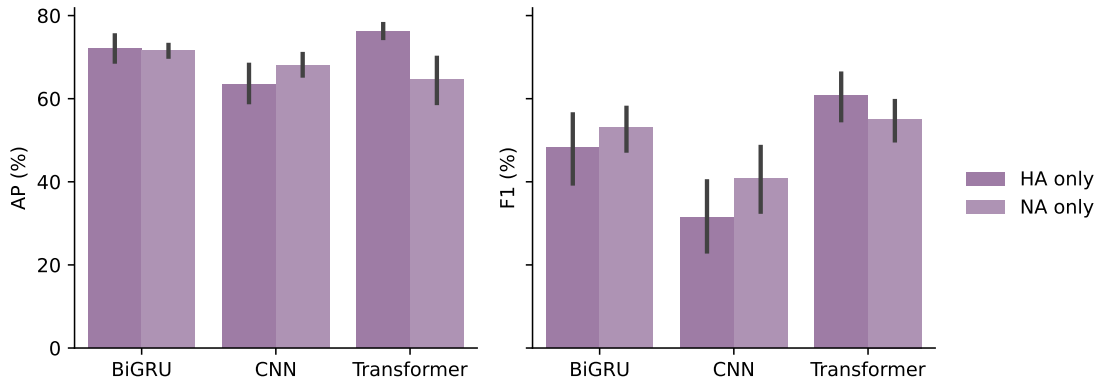


Figure 5.7: Comparison of overall performance between models for host prediction task using single sequence inputs.

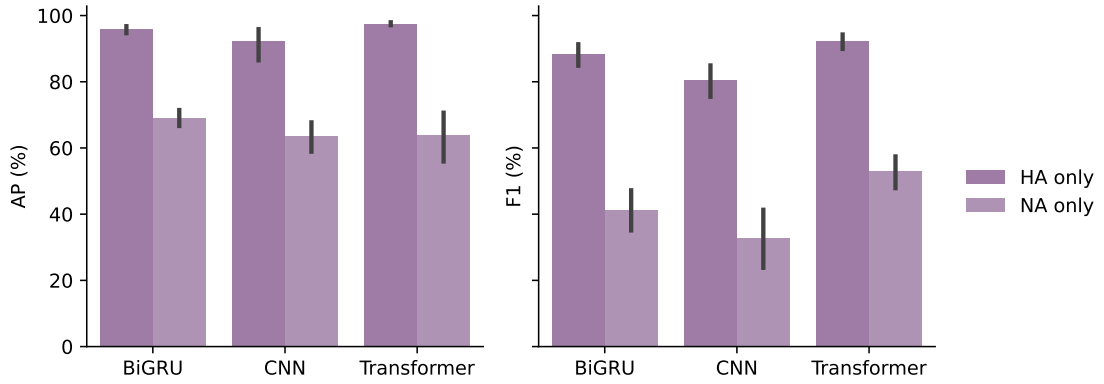


Figure 5.8: Comparison of overall performance between models for HA subtype prediction task using single sequence inputs.

When presented with inputs that lacked the corresponding H/N sequence pairs, there was a discernible decline in performance across all models, particularly for the host prediction tasks. When using the HA-only dataset, the MC-Transformer led in overall performance across all prediction tasks, achieving an average AP of 85.91% (83.14%, 88.67%) and an average F_1 score of 70.59% (63.38%, 77.79%). In contrast, for the NA-only dataset, MC-BiGRU was the top performer, with an average AP of 78.18% (75.50%, 80.87%) and an average F_1 score of 60.89% (54.51%, 67.26%).

For the host prediction task, MC-BiGRU showed comparable performance between the NA-only and NA-only datasets. In contrast, MC-CNN tended to

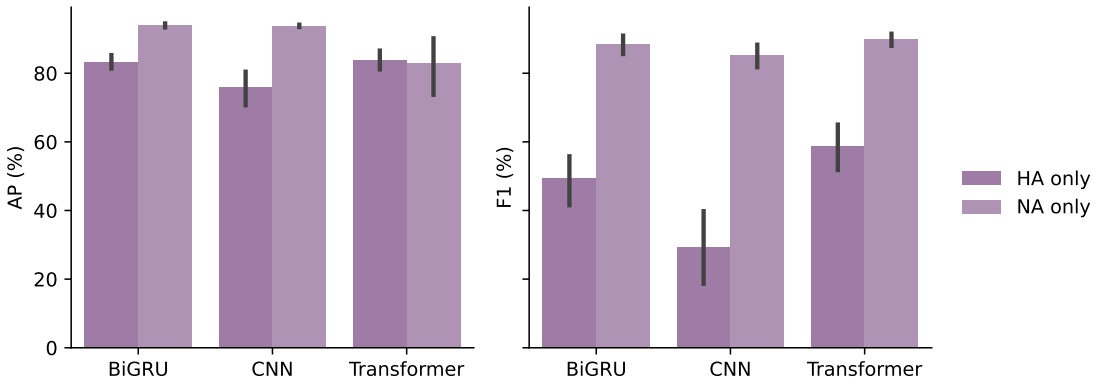


Figure 5.9: Comparison of overall performance between models for NA subtype prediction task using single sequence inputs.

excel with the NA-only dataset, while MC-Transformer performed more favourably with the NA-only dataset. Regarding HA subtype predictions, all models generally displayed better and more consistent performance with the NA-only dataset than with the NA-only dataset. For NA subtype prediction tasks, both MC-CNN and MC-BiGRU demonstrated superior results using the NA-only dataset, whereas MC-Transformer's performance remained consistent between the NA-only and NA-only datasets, as indicated by its AP score.

In summary, the MC-Transformer model consistently outperformed the other models when predicting using single sequence datasets. Furthermore, for the host prediction task, the MC-Transformer favoured the HA-only dataset over the NA-only dataset, while other models preferred the opposite. It was unsurprising that all models exhibited superior prediction accuracy with HA sequences for the HA subtype prediction task and with NA sequences for the NA subtype prediction task.

5.3.3 Performance on Different Feature Sets

From Section 5.3.1 to Section 5.3.2, we discussed the performance of the models under various practical scenarios. However, we had not delved into the impact of different features on model performance. Therefore, the previously discussed model performance was averaged across all feature sets.

Fig. 5.10 to Fig. 5.12 show the model performance for different prediction tasks based on various features.

In the host prediction task, MC-Embedding-CNN stood out, achieving the highest performance on both the pre-20 dataset and post-20 dataset. It reached an AP of 86.90% (85.99%, 87.82%) and an F_1 score of 83.52% (82.71%, 84.34%) for the pre-20 dataset, and an AP of 85.86% (85.58%, 86.14%) and an F_1 score of

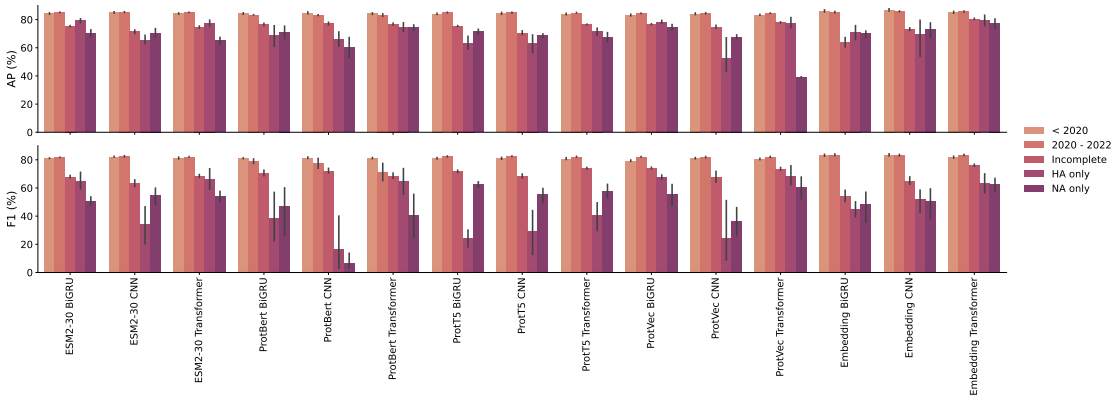


Figure 5.10: Comparison of model performance for hosts prediction task based on diverse features.

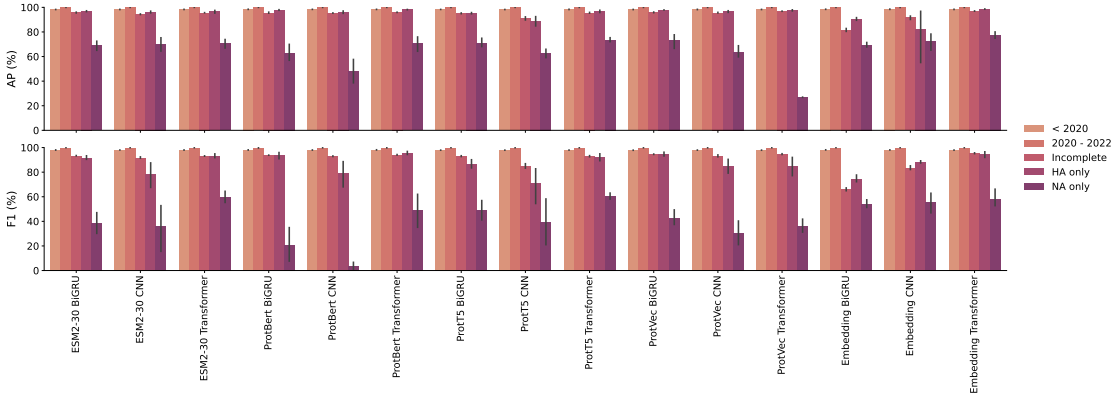


Figure 5.11: Comparison of model performance for HA subtype prediction task based on diverse features.

83.30% (82.76%, 83.84%) for the post-20 dataset. Meanwhile, for the incomplete dataset, MC-Embedding-Transformer had the best performance with an AP of 80.55% (80.06%, 81.03%) and an F_1 score of 76.13% (75.29%, 76.97%). When only HA sequences were present without their corresponding NA pairs, MC-ProtVec-Transformer was the top-performing model based on the average of AP and F_1 score, achieving an AP of 77.59% (73.57%, 81.61%) and an F_1 score of 68.59% (60.90%, 76.28%). However, MC-ProtVec-BiGRU showcased less variability in its performance metrics, with an AP of 78.39% (77.12%, 79.66%) and an F_1 score of 67.59% (65.86%, 69.32%). When inputs comprised only NA sequences without corresponding HA pairs, the MC-Embedding-Transformer showed optimal performance, achieving an AP of 77.41% (73.57%, 81.25%) and an F_1 score of 62.85% (57.98%, 67.73%). However, MC-ProtT5-BiGRU exhibited a more consistent performance, with an AP of 71.56% (69.84%, 73.28%) and an F_1 score of 62.62% (60.56%, 64.69%).

For the HA subtype prediction task, all models generally performed well on

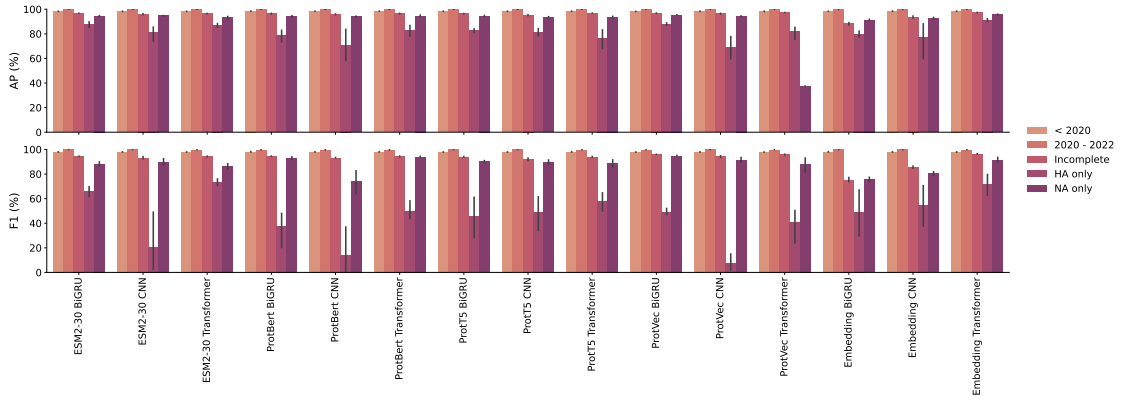


Figure 5.12: Comparison of model performance for NA subtype prediction task based on diverse features.

the pre-20 dataset and the post-20 dataset, with negligible performance differences between them. Therefore, our analysis narrowed down to evaluating performance metrics on the incomplete, HA-only, and NA-only datasets. For the incomplete dataset, MC-Embedding-Transformer was superior, with an AP of 96.99% (96.86%, 97.12%) and an F_1 score of 95.37% (94.94%, 95.81%). For the HA-only dataset, MC-ProBert-Transformer was the top performer, showcasing an AP of 98.19% (98.08%, 98.30%) and an F_1 score of 95.77% (94.39%, 97.14%). Meanwhile, for the NA-only dataset, MC-Embedding-Transformer was the best performer, with an AP of 77.53% (74.62%, 80.45%) and an F_1 score of 58.39% (50.94%, 65.85%), though MC-ProtT5-Transformer showed more stable results, with an AP of 73.53% (71.75%, 75.30%) and an F_1 score of 60.79% (57.86%, 63.72%).

Regarding the NA subtype prediction task, our focus remained on the performance across the incomplete, HA-only, and NA-only datasets. For the incomplete set, MC-Embedding-Transformer was the dominant model, with an AP of 97.35% (97.30%, 97.41%) and an F_1 score of 96.37% (96.18%, 96.56%). On the HA-only dataset, while MC-Embedding-Transformer again was the standout model, with an AP of 91.29% (90.26%, 92.33%) and an F_1 score of 72.02% (62.39%, 81.65%), MC-ESM2-30-Transformer had a smaller variance, with an AP of 87.20% (85.40%, 89.00%) and an F_1 score of 73.38% (70.37%, 76.39%). For the NA-only dataset, MC-ProtVec-BiGRU was the best, with an AP of 95.34% (95.16%, 95.53%) and an F_1 score of 94.87% (94.58%, 95.16%).

In terms of model variance, MC-CNN exhibited the least fluctuation across all feature sets for every prediction task on both the pre-20 and post-20 datasets. In contrast, MC-Transformer showed consistent results when utilising various feature sets for tasks on the incomplete and Single HA datasets. For the NA-only dataset,

MC-BiGRU exhibited stability in terms of AP, while MC-Transformer remained stable in its F_1 scores. This variance was determined by calculating the average scores of the models across different feature sets and tasks.

Fig. 5.13 to Fig. 5.15 show the comparison of AP and F_1 scores across different feature sets for different prediction tasks.

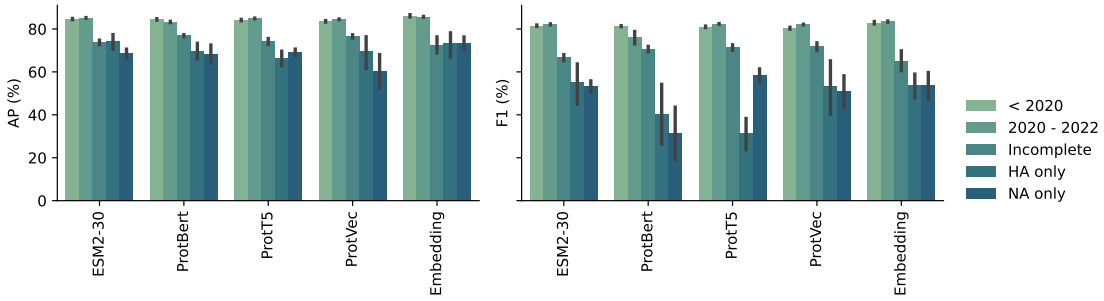


Figure 5.13: Comparison of AP and F_1 scores across different feature sets for the host prediction task.

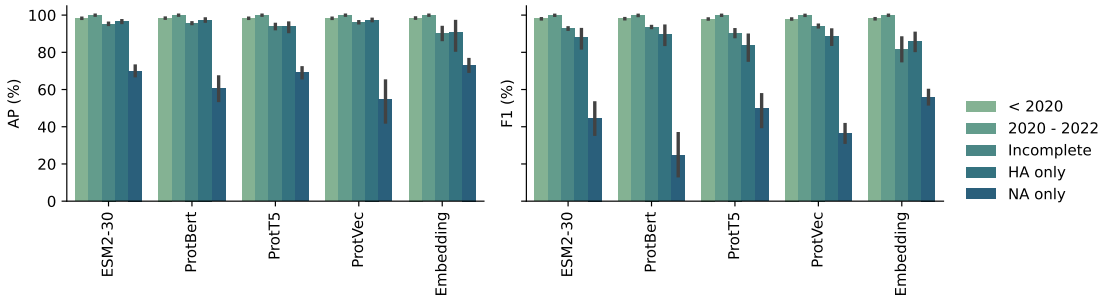


Figure 5.14: Comparison of AP and F_1 scores across different feature sets for the HA subtype prediction task.

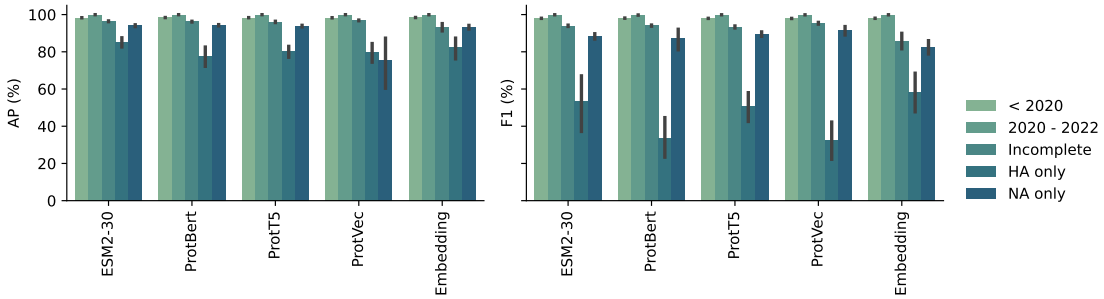


Figure 5.15: Comparison of AP and F_1 scores across different feature sets for the NA subtype prediction task.

The "Embedding" feature set in the figures refers to features derived from the word embedding layer. Specifically, these embeddings are generated based on the

input training data. For the purposes of our study, only the pre-20 dataset was employed to train the model. We implemented 5-fold CV, which implies that the test set sequences were encoded based on the vocabulary and distribution of the training set's corpus.

All feature sets consistently demonstrated strong performance across all models when trained on the pre-20 dataset, with only minimal performance differences. This trend was similar to that seen in the post-20 dataset. However, when evaluating the average metric scores obtained from various models, the ProtVec feature set exhibited a slightly weaker performance in the host prediction task compared to other feature sets.

The variance in performance of the feature sets, once fed into the models, was predominantly observed in the incomplete, HA-only, and NA-only datasets. For the host prediction task, ProtVec and ProtBert outperformed others on the incomplete dataset, ESM2-30 stood out on the HA-only dataset, while Embedding and ProtT5 were the top performers on the NA-only dataset.

For the HA prediction task, ProtVec excelled on the incomplete dataset. Both ProtVec and ProtBert stood out on the HA-only dataset, with Embedding taking the top spot on the NA-only dataset. For the NA prediction task, ProtVec was top-tier in the incomplete dataset. ESM2-30 and Embedding shared the spotlight on the HA-only dataset, while ProtVec and ProtBert were the best performers on the NA-only dataset.

In summary, across both pre-20 and post-20 datasets, all feature sets delivered a commendable performance with minimal variations. ProtVec, while generally robust, showed a slight decline in its host prediction metrics. ProtVec and ProtBert consistently shined in host and HA predictions on the incomplete dataset. ESM2-30 was the best for host prediction on the HA-only dataset and equally shared the top position with Embedding for NA predictions. Embedding, besides its accolade with ESM2-30, Embedding was also the pinnacle performer for HA predictions on the NA-only dataset. ProtT5, along with Embedding, dominated in host predictions for the NA-only dataset. Lastly, ProtVec and ProtBert took the lead for NA predictions on the NA-only dataset.

5.4 Chapter Summary

This chapter delves into developing and evaluating an efficient approach for predicting the hosts and antigenic types of the Influenza A Virus (IAV) using a Multi-Channel Neural Network (MC-NN) model. The necessity of this research stems

from the potential of IAVs to transmit across species, emphasising the importance of early detection and management of influenza outbreaks, as well as the identification of emerging novel subtypes for effective public health planning and outbreak control.

The model introduced in this chapter marks a departure from conventional approaches that typically involve training separate models for different aspects of IAVs' prediction. This approach utilises an end-to-end MC-NN. By harnessing the power of deep learning, it delves into the rich information encapsulated within sequence data, offering a more comprehensive and efficient prediction model.

In terms of data handling, the chapter builds upon methodologies used in previous research, employing complete Hemagglutinin (HA) and Neuraminidase (NA) sequences gathered from the Influenza Research Database (IRD) and Global Initiative on Sharing All Influenza Data (GISAID) databases. After stringent filtering for uniqueness and completeness, the resulting dataset comprises 46,172 unique pairs of HA and NA sequences. These are further divided into pre-2020 and 2020-2022 sets, with an additional subset for incomplete sequences. This segregation aims to enhance the model's performance evaluation, with the pre-2020 set used for training and the 2020-2022 set for testing the model's adaptability to new data. In a deviation from previous chapters, this study segregates incomplete sequences to more accurately determine their impact on the model's efficacy.

The methodology for assigning sequence host labels has undergone a meticulous refinement process. The approach to sequence representation moves away from the reliance on Position-Specific Scoring Matrix (PSSM)-based sequence representations, as used in the study previously discussed in this thesis. Instead, the focus shifts to word embedding techniques and the utilisation of features extracted from pre-trained models, following findings that suggest these methods offer performance comparable to PSSM-based approaches. This shift involves creating dictionaries and embeddings from input data and utilising pre-trained models trained on extensive databases. The objective is to explore whether these pre-trained model features could further enhance model performance or offer new insights.

The chapter details the architecture of the multi-channel neural network used in these experiments. This architecture is unique in its dual input channels (for HA and NA sequences) and its triple output system (predicting host, HA, and NA labels). Given the reality that not all IAV strains have matching pairs of H/N sequences, the model's performance is rigorously evaluated in both typical scenarios and in special cases where only HA or NA sequences are available. This thorough evaluation approach ensures the model's robustness and applicability across various real-world situations, thereby contributing to the field of IAVs' prediction and

public health surveillance. For a brief summary of the results from these extensive analyses, please refer to Table 5.4.

Table 5.4: Summary of results and findings of Chapter 5.

Evaluation Criteria	**Dataset**	**Best Performing Model / Feature Set**	**Observations**
Overall Performance	Pre-20	MC-CNN	All models surpass the baseline performance, with MC-CNN leading with an average AP of 93.92% and an F_1 score of 92.62%.
	Post-20	MC-CNN	A general improvement is observed, with MC-CNN maintaining top performance with an average AP of 94.88% and an F_1 score of 93.81%.
	Incomplete	MC-Transformer	Performance declines across models, with MC-Transformer achieving superior performance. Challenges are noted in predicting hosts.
Overall Performance on Single Sequence Input	HA-only	MC-Transformer	MC-Transformer leads in performance, indicating its effectiveness in HA-only input scenarios.
	NA-only	MC-BiGRU	MC-BiGRU excels, showing strength in NA-only input scenarios.

Continued on next page.

Table 5.4 continued from previous page

Evaluation Criteria	Dataset	Best Performing Model / Feature Set	Observations
Performance Across Models with Varied Feature Sets	Pre-20	MC-Embedding-CNN	MC-Embedding-CNN outperforms others in host prediction tasks. All models generally perform well on the pre-20 dataset for the HA subtype task.
	Post-20	MC-Embedding-CNN	The high performance of MC-Embedding-CNN continues in the post-20 dataset for host prediction. All models generally perform well on the post-20 dataset for the HA subtype task.
	Incomplete	MC-Embedding-Transformer	MC-Embedding-Transformer shows the best performance in the incomplete dataset for all prediction tasks.
	HA-only	MC-ProtVec-Transformer, MC-ProBert-Transformer, MC-Embedding-Transformer	MC-ProtVec-Transformer excels in the host prediction task. MC-ProBert-Transformer is the top performer in the HA subtype prediction task. MC-Embedding-Transformer is the top performer in the NA subtype prediction task.

Continued on next page.

Table 5.4 continued from previous page

Evaluation Criteria	Dataset	Best Performing Model / Feature Set	Observations
	NA-only	MC-Embedding-Transformer, MC-ProtVec-BiGRU	MC-Embedding-Transformer is the top performer for host and HA subtype prediction tasks. MC-ProtVec-BiGRU outperforms others in the NA subtype prediction task.
Feature Set Performance	Pre-20, Post-20	Varied	Feature sets demonstrate strong performance across all models with minimal differences. ProtVec is slightly weaker in host prediction.
	Incomplete	ProtVec, ProtBert	ProtVec and ProtBert stand out in host prediction. ProtVec is best for subtype predictions.
	HA-only	ESM2-30, ProtVec, ProtBert, Embedding	ESM2-30 stands out in host prediction. ProtVec and ProtBert are best for HA predictions. ESM2-30 and Embedding stand out in the NA prediction task.

Continued on next page.

Table 5.4 continued from previous page

Evaluation Criteria	**Dataset**	**Best Performing Model / Feature Set**	**Observations**
	NA-only	Embedding, ProtT5, ProtVec, ProtBert	Embedding and ProtT5 excel in host predictions. ProtVec and ProtBert are best for NA predictions. Embedding is best for HA predictions.

6

Evaluating the Influence of Semi-supervised Learning in Predicting Antigenicity of Influenza A Viruses

6.1 Introduction

This chapter focuses on the crucial task of accurately predicting the antigenicity of Influenza A Viruses (IAVs), a critical factor in developing effective vaccines and anticipating future outbreaks. The measurement of antigenic distance, traditionally reliant on extensive laboratory work, results in a large amount of unlabelled data that remains underutilised. To address this, this chapter delves into the efficacy of semi-supervised learning approaches, specifically label spreading and self-training, to enhance predictive accuracy in estimating antigenic distances. These semi-supervised learning methods effectively combine labelled and unlabelled data, represent a significant advancement in computational biology, and are particularly valuable in contexts where labelled data are scarce.

The data used in this study are sourced from various previous literature, enriching the diversity and complexity of the dataset. The semi-supervised methods employed are foundational yet powerful in exploiting the vast reservoir of unlabelled data. By improving learning algorithms and developing more comprehensive models, these methods are ideally suited for the complex task of flu antigenicity study. The integration of detailed patterns from the limited labelled data with broader trends from the unlabelled data is expected to yield a more in-depth understanding of antigenic evolution.

For encoding the protein sequences, the study employs features derived from pre-trained models, similar to methodologies discussed in Section 5.2.3. This approach

allows for a sophisticated representation of the sequences, facilitating the extraction of relevant patterns and features critical for the prediction tasks.

The comparative analysis conducted in this chapter evaluates the performance of traditional supervised methods against semi-supervised techniques. The dataset comprises high-quality labelled instances and a larger pool of unlabelled influenza virus sequences. The focus is on assessing how well the semi-supervised algorithms leverage the abundance of unlabelled data and their effectiveness in scenarios marked by data scarcity and imbalance, which are common challenges in antigenic distance research.

This research explores the potential of semi-supervised learning to provide more accurate estimations of antigenic distances. Such improvements in predictive accuracy are crucial for the timely development of vaccines and effective public health planning. The findings from this study may provide valuable insights for future research directions and contribute to optimising the application of machine learning techniques in the field of influenza studies.

Our analysis reveals that semi-supervised learning models exhibit performance comparable to that of fully supervised models in most of the evaluated scenarios. In situations where only 25% and 50% of the data are labelled, semi-supervised learning significantly enhances the models' effectiveness, particularly when utilising ProtVec feature sets. Moreover, all models showed enhanced performance on H1N1 and H9N2 viruses as opposed to H3N2 and H5N1. These results underscore the potential of semi-supervised learning to improve the accuracy of antigenic distance estimations, thereby contributing valuable insights for the advancement of vaccine development and public health strategies in the face of influenza virus diversity and evolution.

6.2 Experiments

6.2.1 Data Collection

The data were sourced from previous literature and are publicly available [22, 60, 63, 66, 191, 192]. The original dataset includes 5,311 pairs of Influenza A Viruses (IAVs) and 2,179 HA1 sequences with measured antigenic distance, covering four subtypes.

The Hemagglutinin (HA) protein, composed of three identical subunits, comprises two chains: HA1 and HA2, with lengths of 329 and 175 residues, respectively. The HA1 undergoes mutations more frequently than HA2 and is under significant selective pressure for new variants [193–195].

The HI titers (i.e., the results of the HI assay) are derived from the reaction

between viral hemagglutinins and host antibodies and are often used to understand antigenic differences. If the HI titers of a virus pair differ by four dilutions or more, then this pair is considered antigenically different. However, this data can be challenging to interpret directly due to its complex nature. Therefore, instead of using raw HI values to delineate antigenic differences, the antigenic difference was quantified using HI data, based on Archetti-Horsfall definition [106, 109, 196, 197] providing a more structured and interpretable approach to understanding antigenic relationships. The antigenic distance between the viral strain D and the strain V can be calculated using the Archetti-Horsfall distance formula, as shown below:

$$d_{DV} = \sqrt{\frac{H_{DD} \times H_{VV}}{H_{DV} \times H_{VD}}}$$

where H_{DV} represents the HI titer of viral strain D in relation to antisera produced against strain V.

A pair of viruses was deemed antigenically similar if their antigenic distance was less than 4; otherwise, they were categorised as antigenically different.

The antigenic distance calculated using the Archetti-Horsfall definition can be asymmetric. We excluded viral pairs that showed ambiguous antigenic relationships and those with redundant HA1 sequences. The final dataset comprises distinct viral pairs, unique HA1 sequences, and unambiguous antigenic relationships, as detailed in Table 6.1. The unlabelled data refer to pairs within the same subtype in the dataset that have not been measured. The final dataset contains around 87% unlabelled.

Table 6.1: Summary statistics of the dataset.

Subtypes	# Seqs	# Pairs	# Similar Pairs	# Variant Pairs	# Unlabelled Pairs
H1N1	152	11,448	483	851	10,114
H3N2	61	1,826	125	142	1,559
H5N1	87	3,687	186	265	3,236
H9N2	29	400	31	87	282
Total	329	17,453	825	1,345	15,283

6.2.2 Sequence Representations

We used four state-of-the-art pre-trained models to generate protein sequence embeddings, consistent with the approach outlined in Section 5.2.3. In particular, for the ESM-2 model, we applied a distinct checkpoint featuring a 33-layer architecture that covers 650M parameters (esm2_t33_650M_UR50D). Each protein sequence is encoded into an embedding of 1280 dimensions in this configuration.

6.2.3 Semi-supervised Learning Algorithms

In this work, we used two basic and well-established semi-supervised algorithms: label spreading and self-training.

Label Spreading

Semi-supervised learning leverages the idea that points in the input space sharing the same label, or those residing within the same structure or manifold in the input space, should be close to one another [198]. One popular semi-supervised technique constructs a graph that establishes connections among examples in the training dataset and then propagates known labels through the graph's edges to label unlabelled examples. This approach, known as label spreading [198], draws inspiration from experimental psychology's spreading activation networks.

Label spreading constructs a similarity graph where each node corresponds to a data point. The data points in this graph are interconnected based on their relative distances within the input space. The edges of this graph are assigned weights that reflect the similarity between nodes, typically calculated using a distance metric within the feature space. As the algorithm progresses, it propagates information throughout this graph to capture the input space's inherent structure. The algorithm then uses the spread of labels across this network to effectively classify unlabelled data points, taking advantage of the established relationships and similarities within the graph.

The strength of label spreading lies in its ability to incorporate the underlying structure of both labelled and unlabelled data, aiding in uncovering inherent groupings within the dataset. This attribute makes it particularly useful in scenarios where the labelled data is limited or may not represent the full diversity of the input space, as is often the case in biological data sets. The propagation of labels through the graph enables the model to learn from the vast, untapped information present in the unlabelled data, potentially enhancing classification performance and deepening the understanding of the underlying patterns and structures within the data.

Self-training

Self-training [199] begins with a pre-existing classifier, often termed the pseudo-labeller, which is initially trained on a small set of labelled data. This pseudo-labeller is then used to generate predictions, known as pseudo-labels, on a larger pool of unlabelled data. The predictions made with the highest confidence are selected according to specific thresholds or criteria, and their pseudo-labels are added to

the training dataset. The model undergoes retraining with this expanded dataset, progressively integrating these new examples.

Self-training has the potential to substantially improve model accuracy, even though it depends on self-generated predictions instead of true labels. This is partly due to the model's ability to progressively refine itself through exposure to a broader array of training examples in each iteration. The approach is particularly beneficial in contexts where labelled data is limited or costly to obtain, allowing the model to capitalise on the abundance of unlabelled data.

However, the effectiveness of self-training hinges on the initial model's accuracy. If the pseudo-labeler's predictions are erroneous, these errors can be perpetuated in subsequent iterations, potentially compromising the model's overall performance. Therefore, strategically determining confidence thresholds and stopping criteria is crucial to ensure that self-training contributes positively to the model's predictive accuracy. This method's efficacy in expanding the training set and enhancing generalisation capabilities, particularly in data-scarce environments, makes it a valuable tool in the semi-supervised learning toolkit.

6.2.4 Implementation and Evaluation

In this study, we applied traditional supervised learning algorithms, including Convolutional Neural Network (CNN), Bidirectional Gated Recurrent Unit (BiGRU), Support Vector Machine (SVM), and Random Forest (RF), in a fully supervised setting, to provide a comparison with semi-supervised learning models. Specifically, RF and SVM were chosen as the base model for self-training due to their simplicity and computational efficiency, which are beneficial for iterative retraining processes in semi-supervised learning. A detailed description of each algorithm and its corresponding theory can be found in Section 2.2.5. In this section, we will focus on discussing the parameter settings and architecture of the models applied in our experiments.

Table 6.2 details the hyperparameter settings used in this study, and Fig. 6.1 illustrates the architecture of CNN and BiGRU.

Consistent with the methodologies applied in the preceding two studies of this thesis, we used a nested k-fold Cross-validation (CV) approach (the details of nested k-fold CV can be found in Section 2.2.6), using 5 outer folds ($k_{outer} = 5$) and 4 inner folds ($k_{inner} = 4$), with the F_1 score as the performance metric.

In contrast to the previous studies in this thesis, which primarily used labelled data, thereby enabling straightforward data partitioning for evaluation, this study

Table 6.2: Hyperparameter settings.

Algorithms	Hyperparameters
SVM	C = $[10^{-6}, 10^{1}]$ gamma = $[10^{-6}, 10^{1}]$
RF	n_estimators = 10, 50, 100, 150, 200 max_depth = 5, 10, 15, 20, None
CNN	num_filters = 32, 64, 128, 512 learning_rate = 1e-5,1e-4,1e-3,1e-2 batch_size = 32, 64, 128 epochs = 500 kernel_size = 3, 5
BiGRU	units = 32, 64, 128, 256, 512 learning_rate = $10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}$ batch_size = 32, 64, 128 epochs = 500
Label Spreading	kernel = 'knn' alpha = 0.1, 0.2, 0.3 n_neighbors = 3, 5, 7, 11, 20, 30, 40, 50, 75, 100 max_iter = 20, 25, 30, 35, 40, 45, 50
Self-training	threshold = 0.5, 0.55, 0.6, ..., 0.95, 0.99 criterion = 'threshold', 'k_best' k_best = 3, 5, 10, 15 max_iter = 5, 10, 15, 20

employed a significant proportion of unlabelled data. Therefore, we added additional steps before CV, as shown in Fig. 6.2. Each inner fold is composed exclusively of labelled data with an equitable distribution of labels across all folds. To evaluate the impact of the ratio of labelled data on the model's performance, we also adjusted the proportion of labelled data within the training set to 25%, 50%, 75%, and 100%, respectively.

6.3 Results and Discussion

In this study, several experiments were carried out to evaluate the effectiveness of our models. The evaluations spanned a range of aspects, such as general performance, the efficacy of each feature set, performance across various subtypes, and performance in relation to different proportions of labelled data.

Our findings reveal that the semi-supervised models exhibit performance comparable to that of fully supervised models in most of the evaluated scenarios. Specifically, in scenarios where 25% and 50% of labelled data were available, semi-supervised learning significantly improved the efficacy of models employing the ProtVec feature sets. Moreover, all models showed superior performance on H1N1 and H9N2 subtypes when compared to H3N2 and H5N1.

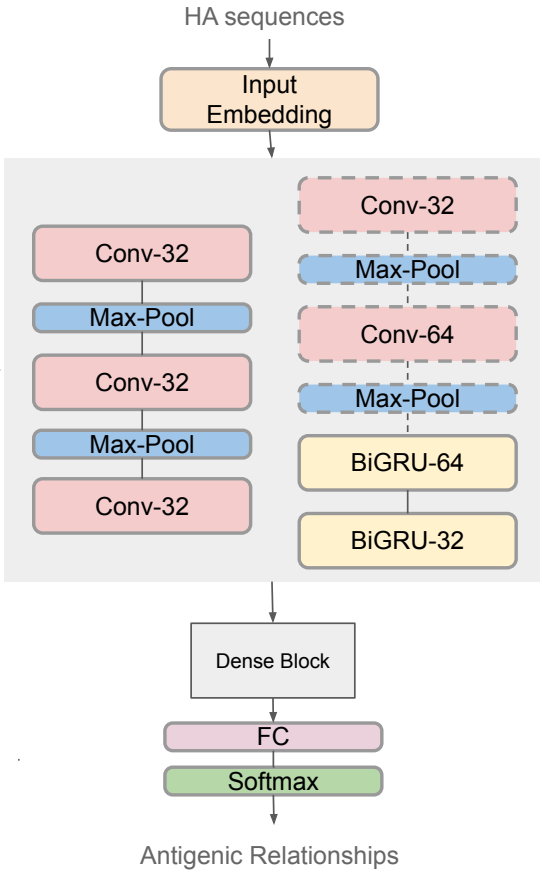


Figure 6.1: Model architecture for CNN and BiGRU.

For detailed insights into these findings, please refer to the subsequent sections.

6.3.1 Overall Performance

Figure 6.3 compares fully supervised and semi-supervised learning approaches across various ratios of labelled data, using features derived from different pre-trained models. As expected, there is a general trend of improved model performance with an increased ratio of labelled data, since more labelled data typically helps achieve better model performance. Ideally, semi-supervised learning methods would surpass the performance of their fully supervised counterparts, but this was not a constant outcome in our study. The semi-supervised models demonstrated comparable performance to the fully supervised models across most evaluated scenarios.

For simplicity and clarity in our discussion, each model within our study is identified using a structured format: [*Training Approach*] - [*Pre-trained Model*] - [*ML Algorithm*]. For example, the model's name "FullySupervised-ESM2_33-RF" breaks down as follows: "FullySupervised" indicates that the model undergoes

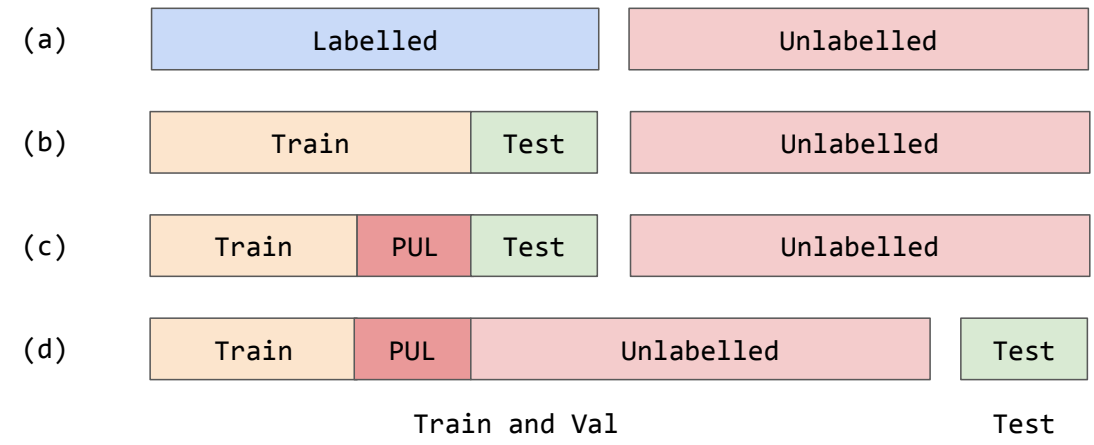


Figure 6.2: Incorporation of unlabelled data in cross-validation. (a) the initial dataset employed in the experiment; (b) the labelled data is partitioned into training and test sets for model training and performance evaluation, respectively, while the unlabelled data is reserved for subsequent steps; (c) a fraction of the training data is divided as pseudo-unlabelled data (PUL) based on a specific labelled ratio, simulating an unlabelled state, while the test data remains intact for unbiased performance assessment; (d) the training set is fused with the pseudo-unlabelled data and untouched unlabelled data to form a final training set used for model fitting and validation. The test data is preserved for the final evaluation of the model's performance on new data.

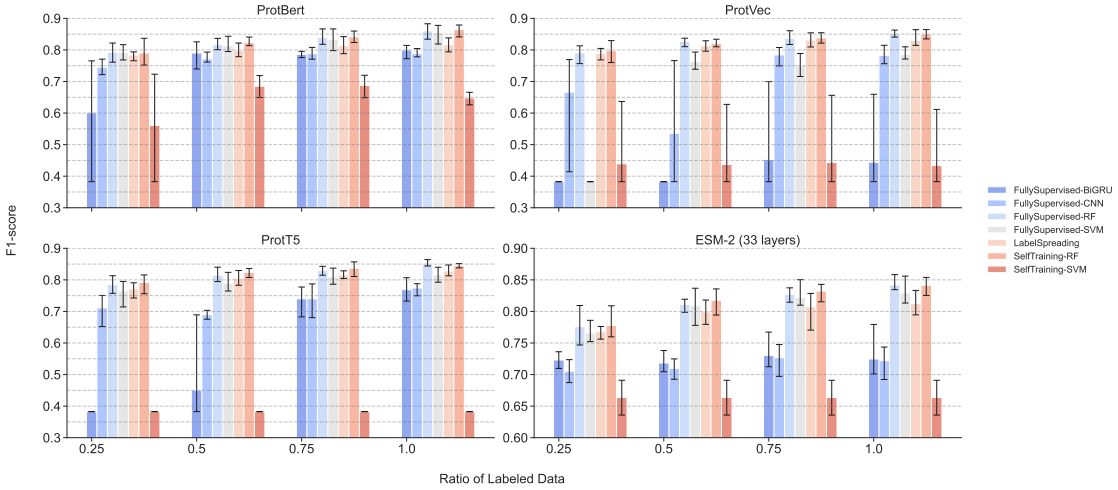


Figure 6.3: Comparing supervised and semi-supervised methods with different pre-trained model features at different labelled data ratios.

training in a fully supervised manner, "ESM2_33" identifies the specific pre-trained model used for generating features, and "RF" represents the machine learning algorithm we applied. The "FullySupervised-RF" collectively denotes models trained in a fully supervised way, incorporating various pre-trained models for feature extraction and using the RF algorithm.

We will next examine which models, both fully supervised and semi-supervised, perform optimally at different ratios of labelled data. At 25% labelled data, the best performers are SelfTraining-ProtVec-RF and FullySupervised-ProtBert-RF, achieving an F_1 score of 0.797 (CI: 0.760, 0.829) and 0.792 (CI: 0.761, 0.821) respectively. With 50% labelled data, FullySupervised-ProtVec-RF and SelfTraining-ProtT5-RF lead, achieving an F_1 score of 0.826 (CI: 0.810, 0.837) and 0.823 (CI: 0.807, 0.836) respectively. At 75% labelled data availability, SelfTraining-ProtBert-RF and FullySupervised-ProtBert-RF emerge as the top models, with an F_1 score of 0.841 (CI: 0.823, 0.859) and 0.837 (CI: 0.817, 0.866) respectively. Finally, when 100% of the labelled data is provided, SelfTraining-ProtBert-RF and FullySupervised-ProtBert-RF again perform best, with an F_1 score of 0.864 (CI: 0.841, 0.878) and 0.860 (CI: 0.834, 0.882), respectively. We also found that both the SelfTraining-SVM and FullySupervised-BiGRU models exhibited their poorest performance when using features extracted either from ProtT5 or ProtVec.

6.3.2 Performance for Pre-trained Features

We fed features from four different pre-trained models into the models. In this section, we evaluated the performance of feature sets extracted from different pre-trained models, namely ESM-2, ProtBert, ProtVec, and ProtT5, by averaging their F_1 scores across a range of machine learning models, as the results are shown in Fig. 6.4.

Our findings indicate that feature sets from ProtVec generally yielded the lowest performance. Conversely, feature sets from ProtBert showed promising results, particularly when the proportion of labelled data was 50% or higher. Additionally, the ESM-2 feature sets demonstrated remarkable consistency in performance across varying ratios of labelled data and presented the least variance compared to other feature sets. ProtBert feature sets were the next most stable in terms of variance.

The analysis reveals a trend where richer labelled data typically leads to higher F_1 scores across the different feature sets, indicating improved model efficacy. However, some pre-trained model feature sets, like those from ESM-2, exhibit strong performance even with fewer labelled data, offering potential advantages in situations where labelled data is limited.

Fig. 6.5 presents a comparative analysis of feature sets extracted from various pre-trained models under different learning paradigms. In contexts where 25% and 50% labelled data are available, semi-supervised learning significantly enhances the efficacy of models using ProtVec feature sets. On the other hand, this approach seems to reduce the performance of models built on ProtBert feature sets. Models

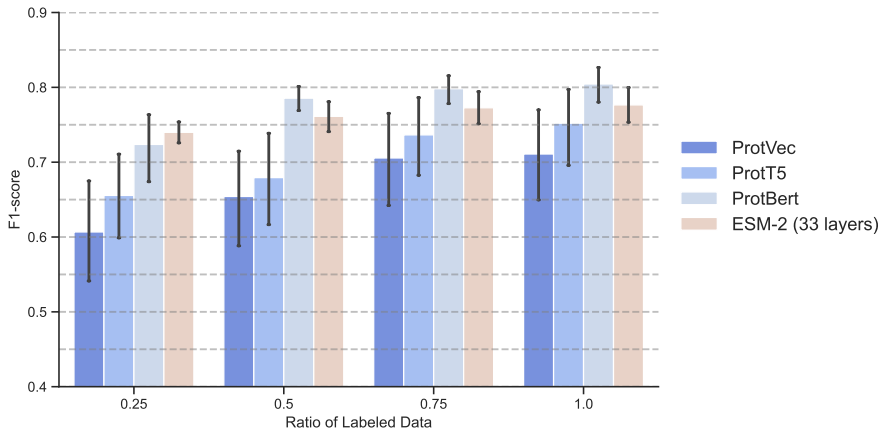


Figure 6.4: Comparing various pre-trained model features at different labelled data ratios.

that utilised other feature sets show similar performance levels in supervised and semi-supervised learning.

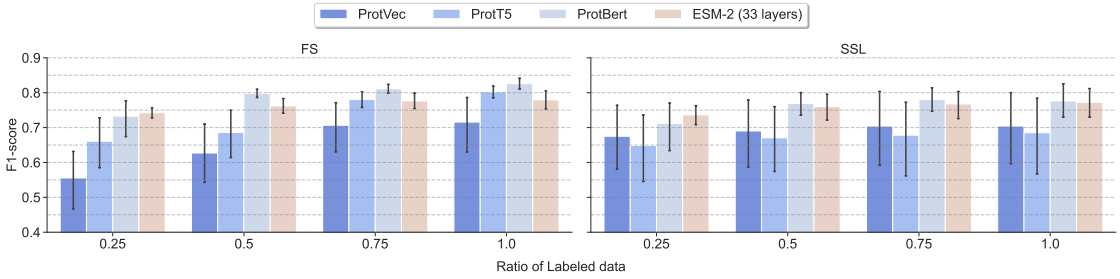


Figure 6.5: Comparing various pre-trained model features in fully supervised and semi-supervised settings at different labelled data ratios.

6.3.3 Performance in Each Subtype

Figures 6.6 and 6.7 display the performance outcomes for the FullySupervised-RF and SelfTraining-RF models, using features extracted from ESM-2 and ProtBert, respectively. Each figure specifically illustrates these models' performance across various subtypes.

All models demonstrate superior performance on H1N1 and H9N2 compared to H3N2 and H5N1. Specifically, for H5N1, all models show good results with more than 50% labelled data. Notably, semi-supervised learning does not significantly outperform supervised learning in this context, with supervised learning even surpassing semi-supervised learning when only 20% of labelled data is available. Regarding H3N2, semi-supervised learning does not perform markedly better than supervised learning with 100% labelled data. However, when 75% of labelled

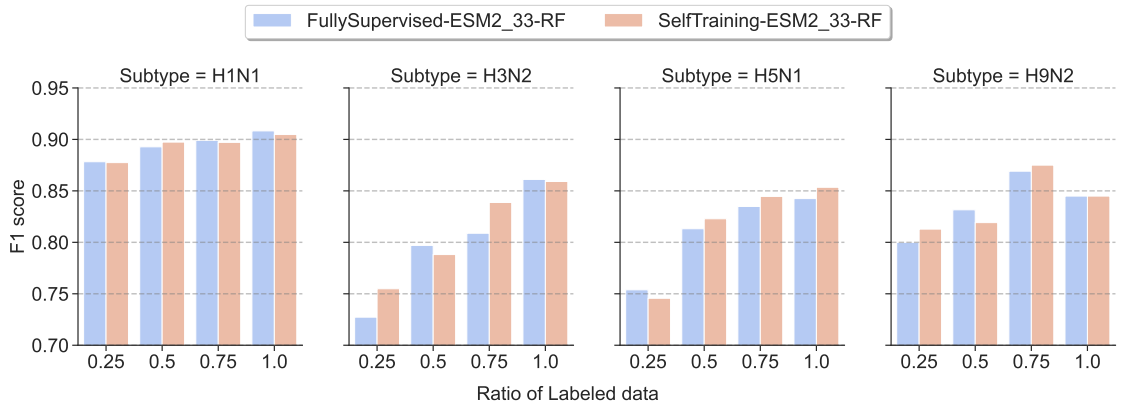


Figure 6.6: Comparison of fully supervised and self-training methods using ESM-2 (33 layers) features in predicting antigenicity of influenza A viruses across different subtypes with varied levels of labelled data.

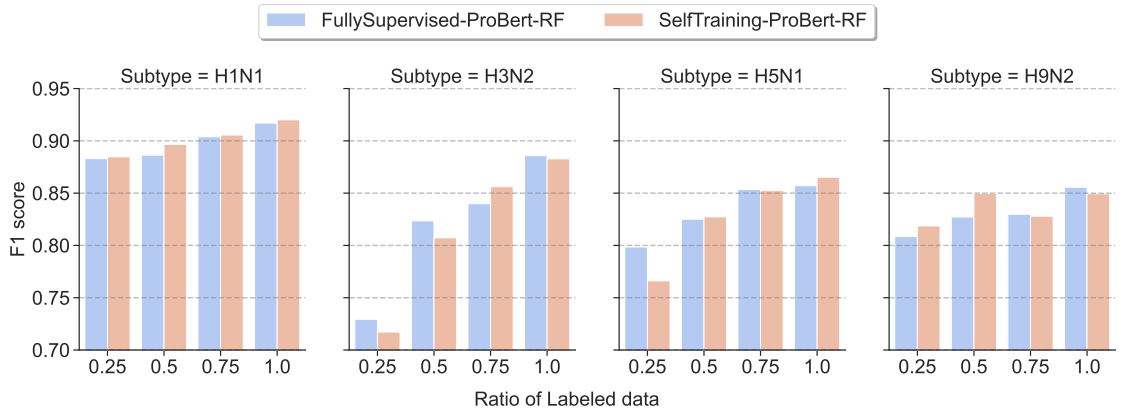


Figure 6.7: Comparison of fully supervised and self-training methods using ProtBert features in predicting antigenicity of influenza A viruses across different subtypes with varied levels of labelled data.

data is provided, semi-supervised learning tends to surpass supervised learning. At a 25% labelled data level, the SelfTraining-ESM2_33-RF model outshines its fully supervised counterparts, whereas SelfTraining-ProtBert-RF exhibits the opposite trend.

6.4 Chapter Summary

This chapter presents an exhaustive examination of semi-supervised learning methodologies and their impact on predicting antigenicity of Influenza A Viruses (IAVs). It aims to illuminate the potential of these techniques to mitigate the challenges that arise from the scarcity of labelled data in virological studies, with a particular focus on the influenza virus.

Traditional supervised learning paradigms face limitations due to their reliance on large labelled data sets, which are often not readily available. This is attributed to the swift mutation rates of IAVs, and the practical difficulties encountered in laboratory tests.

In an effort to address these limitations, this study delves into the applicability of semi-supervised learning strategies, with a special emphasis on label spreading and self-training algorithms. These methods are comprehensively described, underscoring their ability to utilise both labelled and unlabelled data to improve predictive accuracy. The label spreading algorithm is expounded for its proficiency in transferring information from labelled to unlabelled data points, thus enhancing the training dataset. In parallel, the self-training approach is depicted as an iterative process that enriches the labelled dataset with unlabelled instances predicted with high certainty.

Additionally, this chapter applies features extracted from four distinct pre-trained models. This inclusion aims to further refine the predictive capability of the semi-supervised learning approaches. The chapter's empirical segment utilises various supervised learning models, such as Support Vector Machine (SVM) and Random Forest (RF), as baselines to gauge the improvements offered by semi-supervised methods. A comparative analysis is conducted between models trained solely on supervised learning and those augmented with semi-supervised techniques.

Results from these experiments are rigorously scrutinised to determine the contexts in which semi-supervised learning is most effective. The chapter elucidates the metrics employed for evaluation and discusses the implications of these findings for IAVs' antigenicity prediction. For a brief summary of the results of these extensive analyses, please refer to Table 6.3.

Table 6.3: Summary of results and findings of Chapter 6.

Aspect	**Fully Supervised Best Performers**	**Semi-Supervised Best Performers**
Comparison across various ratios of labelled data.	At 25% labelled data: FullySupervised-ProtBert-RF (F_1: 0.792) At 50% labelled data: FullySupervised-ProtVec-RF (F_1: 0.826) At 75% labelled data: FullySupervised-ProtBert-RF (F_1: 0.837) At 100% labelled data: FullySupervised-ProtBert-RF (F_1: 0.860)	At 25% labelled data: SelfTraining-ProtVec-RF (F_1: 0.797) At 50% labelled data: SelfTraining-ProtT5-RF (F_1: 0.823) At 75% labelled data: SelfTraining-ProtBert-RF (F1: 0.841) At 100% labelled data: SelfTraining-ProtBert-RF (F1: 0.864)
Averaged (F_1 scores across models for different pre-trained models.	ProtBert features perform best, especially at higher data ratios.	Significant enhancement with ProtVec features in semi-supervised learning.
Performance across H1N1, H9N2, H3N2, H5N1 subtypes.	Varies; H1N1 and H9N2 see superior performance	

7

Conclusion and Future Work

7.1 Introduction

In bioinformatics, augmented by machine learning, we have witnessed remarkable advancements in addressing critical challenges in virology. This work has involved a deep exploration into predicting Influenza A Viruses (IAVs), utilising both established and emerging methods. Our engagement with Position-Specific Scoring Matrices (PSSMs), multi-channel neural networks (MC-NNs), and other advanced techniques has illuminated the capabilities of computational models to unravel the complexities of biological systems.

This fusion of extensive virological knowledge with sophisticated computational techniques has not only sharpened the accuracy of predicting hosts of IAVs but also revealed intricate patterns and relationships. These achievements indicate our progress in comprehending the IAV and forming a solid foundation for a more informed and proactive approach to managing future outbreaks.

This work is underpinned by a philosophy similar to that of Google's GraphCast [200], an open-source Artificial Intelligence (AI) weather forecasting system. GraphCast epitomises a paradigm shift in predictive modelling, moving from traditional methods to a data-driven AI approach. Rather than adhering to meteorological principles or physical laws, GraphCast uses four decades of weather data to train its model. Its success is rooted in a "black box" approach, prioritising utility over an understanding of the underlying complexities of the natural world.

This thesis embraces a data-driven philosophy, indicating that an exhaustive comprehension of the IAV might not be essential to design effective predictive models. The core of our approach is to exploit existing protein data, which is the sole basis for most of our experiments. This focus on protein data allows our models to learn and make predictions based solely on these datasets without

additional information. Consequently, while the finer details of the IAV's behaviour may remain elusive to models, our approach still yields pragmatic outcomes by capitalising on the insights gleaned from this protein data.

7.2 Summary of Results and Findings

The key results and findings from each study conducted in this thesis will be presented as follows. This will showcase how these approaches have contributed to our understanding and prediction capabilities in the context of the IAV research.

Chapter 4 This chapter presents an extensive analysis of the results obtained from various bioinformatics and machine learning methods used to predict host origins of IAVs. The performance of these methods is evaluated at different taxonomic levels and in varying conditions, including the presence of incomplete sequences and the choice of reference databases. The chapter also explores the effectiveness of ensemble results, integrating multiple models for a holistic assessment.

Sequence Representations and Performance:

- Word embeddings (especially 3-grams) outperformed Position-Specific Scoring Matrix (PSSM)-based methods in most cases.
- The performance of all methods was generally better at higher taxonomic levels than lower ones.
- At lower taxonomic levels, the mean score drops for all sequence representations, ranging from 10% to 15%.

Machine Learning Algorithms:

- Transformer and XGBoost algorithms performed best at higher and lower taxonomic levels, respectively.
- A decline in performance was observed for all classifiers when moving from higher to lower taxonomic levels, with mean score reductions between 9% and 30%.

Impact of Incomplete Sequences:

- Models showed a notable decrease in performance and stability on dataset 2, influenced by noisy data from incomplete sequences.
- However, the impact on their effectiveness was not overwhelmingly large, indicating a certain level of robustness.

Ensemble Results:

- The ensemble of models achieved a 97.56% match between predicted and true labels, with 2.44% of sequences incorrectly predicted.
- Some sequences, especially those collected during outbreak periods or possessing strong cross-species characteristics, were consistently mispredicted.

Effect of Reference Database:

- The choice of reference database (partial NR vs. UniRef50) played a significant role in the PSSM-based model's predictive capabilities.
- The performance of PSSM-based models, which used PSSMs generated by different reference databases, varied. Thus, the choice of the reference database for PSSM calculation is important.

Overall Observations:

- Models tended to perform less effectively on datasets that incorporated incomplete sequences.
- The adaptability and efficiency of UniRef50-PSSM in managing complex classification scenarios were observed.

This comprehensive analysis underscores the complexity and challenges in predicting influenza host origins using computational techniques. The results highlight the importance of considering various factors such as data quality, taxonomic level, and the choice of reference database to optimise the performance of bioinformatics and machine learning models in real-world scenarios. The integration of ensemble results provides a holistic perspective on the efficacy of combining multiple models, which can offer valuable insights for future research and practical applications in influenza surveillance and response efforts.

Chapter 5 This chapter provides a comprehensive analysis of the performance of various models in predicting hosts and antigenic types of IAVs, focusing on different datasets, single sequence inputs, and the impact of diverse feature sets.

Overall Model Performance Across Datasets

1. Pre-2020 and Post-2020 Datasets:
    - The models were trained on the pre-2020 dataset and tested on both the pre-2020 and post-2020 datasets, containing only complete sequences.
    - The best-performing model was MC-CNN, achieving an average AP of 93.92% and an average F_1 score of 92.62%.
    - In the post-2020 dataset, there was a general improvement in performance, with MC-CNN remaining the top performer.
2. Incomplete Dataset:
    - A decline in performance was observed across all models in the incomplete dataset.
    - The MC-Transformer model showed superior performance with an average AP of 90.13% and an average F_1 score of 87.09%.

Performance on Single Sequence Input

1. HA-only Dataset: MC-Transformer led in overall performance, achieving an average AP of 85.91% and an average F_1 score of 70.59%.
2. NA-only Dataset: MC-BiGRU was the top performer, with an average AP of 78.18% and an average F_1 score of 60.89%.
3. Comparative Analysis:
    - MC-CNN excelled with NA-only inputs.
    - For the host prediction task, the MC-Transformer performed better with HA-only inputs, while others preferred the opposite.
    - For HA subtype predictions, all models generally performed better with HA-only inputs.
    - For NA subtype predictions, all models generally performed better with NA-only inputs.

Performance Based on Different Feature Sets

1. Host Prediction Task:
    - MC-Embedding-CNN and MC-Embedding-Transformer exhibited the highest performance across different datasets.
    - ProtVec and ProtBert outperformed other feature sets across different datasets.
2. HA Subtype Prediction Task:
    - MC-ProBert-Transformer was the top performer in the HA-only dataset.
    - ProtVec excelled on the incomplete dataset and shared the top spot with Embedding on the NA-only dataset.
3. NA Subtype Prediction Task:
    - MC-Embedding-Transformer dominated in the incomplete and HA-only datasets.
    - ProtVec and ProtBert led in the NA-only dataset.
4. Model Variance:
    - MC-CNN exhibited the least fluctuation across feature sets for all prediction tasks.
    - MC-Transformer showed consistent results with various feature sets, especially in incomplete and HA-only datasets.

This chapter highlights the robustness of the models in predicting influenza A virus hosts and subtypes across various scenarios and datasets. The performance varied depending on the dataset and feature set used, with MC-CNN, MC-Transformer, and MC-BiGRU emerging as significant models across different tests.

Chapter 6 The chapter provides a detailed analysis of the performance of both supervised and semi-supervised learning models in predicting influenza antigenicity. The models were evaluated based on various ratios of labelled data, using features derived from different pre-trained models, and their performance was assessed across various influenza subtypes.

Model Performance with Different Ratios of Labelled Data:

1. 25% Labelled Data: The best performers were SelfTraining-ProtVec-RF and FullySupervised-ProtBert-RF, with F_1 scores of 0.797 and 0.792, respectively.

2. 50% Labelled Data: FullySupervised-ProtVec-RF and SelfTraining-ProtT5-RF led with F_1 scores of 0.826 and 0.823, respectively.

3. 75% Labelled Data: SelfTraining-ProtBert-RF and FullySupervised-ProtBert-RF emerged as top models with F_1 scores of 0.841 and 0.837, respectively.

4. 100% Labelled Data: SelfTraining-ProtBert-RF and FullySupervised-ProtBert-RF again performed best with F_1 scores of 0.864 and 0.860, respectively.

5. Overall Trend: A general trend of improved model performance was observed with an increased labelled data ratio. Semi-supervised models demonstrated comparable performance to the fully supervised models across most scenarios.

Performance Based on Pre-trained Model Features:

1. ProtVec Features: Generally yielded the lowest performance.

2. ProtBert Features: Showed promising results, particularly when the proportion of labelled data was 50% or higher.

3. ESM-2 Features: Stood out at a 25% labelled data ratio, exhibiting the best performance and demonstrating remarkable consistency across varying ratios of labelled data.

4. Overall Trend: Richer labelled data typically led to higher F_1 scores across different feature sets. Some pre-trained model feature sets, such as those from ESM-2, exhibited strong performance even with fewer labelled data, offering advantages in situations where labelled data is limited.

Performance Across Various Influenza Subtypes:

1. Subtypes H1N1 and H9N2: All models performed better than H3N2 and H5N1.

2. Subtype H5N1: Good results were observed with more than 50% labelled data.

3. Subtype H3N2: Semi-supervised learning tended to surpass supervised learning when 75% of labelled data was provided.

4. Specific Trends: The SelfTraining-ESM2_33-RF model outperformed its fully supervised counterparts at a 25% labelled data level, while SelfTraining-ProtBert-RF exhibited the opposite trend.

5. General Observation: semi-supervised learning did not consistently outperform supervised learning, with the latter even surpassing semi-supervised learning in some scenarios with limited labelled data.

The chapter reveals nuanced insights into the efficacy of various models and feature sets in predicting influenza antigenicity, emphasising the importance of the proportion of labelled data and the choice of feature sets in achieving optimal performance. The analysis also underscores the varying effectiveness of the model across different influenza subtypes.

7.3 Summary of Limitations

This thesis offers valuable insights and advances in the field but also faces limitations that warrant discussion. In this section, we describe the primary constraints encountered in our research:

One limitation arises from the evaluation methodologies employed in our experiments, which may not fully capture models' ability to predict the cross-species transmission potential of IAVs, given that the data used are confined to a single class. Future studies should aim to develop or integrate more specialised evaluation methods that directly assess and validate model efficacy in predicting cross-species transmission scenarios.

The challenge of identical protein sequences yielding varied functionalities underscores another limitation. Relying solely on protein sequences to predict functional outcomes does not account for the complexities underlying these variances, suggesting a need for more nuanced analytical approaches.

The use of Basic Local Alignment Search Tool (BLAST) also presents limitations and complexities. For example, the "`max_target_seqs`" parameter might not return the highest scoring sequences but, instead, the initial sequences exceeding a certain E-value threshold, potentially compromising result accuracy [201]. Moreover, BLAST's array of configurations, including different scoring matrices and gap penalties, can

significantly influence search outcomes, necessitating careful consideration when using it as a standard benchmark [202].

The models we implemented are designed to produce a probability distribution for various labels, with a default setting that assigns labels based on a 50% probability threshold. This means that a label is assigned to a given sample if the model's predicted probability for that label exceeds 50%. This method assumes equal importance and error cost for all classifications, which may not always align with practical scenarios. Therefore, it highlights the need for model calibration to tailor the threshold more effectively to the specific requirements of our application. Calibration could adjust this threshold to better reflect the varying significance of different labels and the differing misclassification costs, thereby improving the model's practical accuracy and applicability to real-world data.

In the second and third of our study, we employed features from popular pre-trained protein language models to convert sequence data into a numerical format. However, this introduces the risk of data leakage, as sequences for validation and testing might overlap with those used in training the pre-trained models. This overlap can inadvertently lead to overly optimistic performance evaluations, as the models may recognise and predict these sequences with higher accuracy than genuinely unseen data.

The second study also introduced a MC-NN designed to predict influenza hosts and subtypes. The separate input channels for HA and NA sequences merge early (i.e., right after embedding layers) in the process to optimise training time. They may capture the interactions or relationships between proteins. However, this early merging strategy potentially limits the model's capacity to learn variations specific to either protein, as well as its ability to extract high-level features from the protein sequences. Thus, this approach may risk bias towards one protein's features over the other.

Furthermore, we explored fundamental self-training algorithms for IAVs' antigenicity prediction. However, such basic semi-supervised learning algorithms may not fully capture the complex nature of protein sequence data.

Throughout most studies, we applied standard stratified nested k-fold Cross-validation (CV) to mitigate overfitting, but this method does not account for the evolutionary trajectory of IAVs, which progresses linearly rather than cyclically [203].

Furthermore, our subtype prediction analysis may not fully encompass the potential evolutionary trajectories of the H1N1 subtype into various forms, thus revealing a limitation in our approach to subtype prediction. As viral genomes continuously undergo mutations and reassortments, there is a possibility for new

variants or subtypes emerging beyond those considered in our study. Consequently, the predictive scope of our analysis may be constrained by this inherent variability and unpredictability in viral evolution. Therefore, while our study has found models capable of accurately classifying subtypes, future research needs to incorporate mechanisms for anticipating and adapting to potential shifts in viral subtypes, ensuring the continued relevance and accuracy of predictive models in combating infectious diseases.

7.4 Future Work

The advancements chronicled in this work open avenues for further exploration. A prominent direction lies in using machine learning for epidemic decision-making. The current frameworks predominantly harness protein sequences to extract protein-level information. However, our findings suggest that identical protein sequences can sometimes lead to varied protein functionalities. Relying solely on protein sequences presents limitations in unravelling the underlying reasons for such variances.

We presented a multi-channel neural network to process HA and NA protein sequences simultaneously. This approach allowed us to capture more complex information from these key influenza proteins, shedding light on the complex interplay of viral components. The promising results of this approach highlight the potential to expand the number of input channels of this model to handle all influenza virus proteins, while reducing the need for sequences from each virus to contribute all proteins. Another future work in this area lies in exploring embedding features from multi-sources instead of using the same word embedding scheme for different protein inputs. For example, each input channel can be used with a different pre-trained embedding.

Gene regulations also play an important role in influencing protein functionalities. For example, variations in codon usage can profoundly impact protein synthesis and folding. Certain codons are translated more rapidly than others, and these translation rate discrepancies can influence how an amino acid sequence folds into a protein. Slow translation processes can even initiate Ribonucleic Acid (RNA) degradation, impeding protein production altogether, as evidenced by research from [204].

Moreover, the ambient environment of a protein, characterised by factors such as temperature, pH levels, and molecular interactions, can significantly dictate protein folding patterns. A notable example is that of prion proteins, which can manifest in multiple structural forms. While one is typically observed in

healthy cells, its misfolded counterpart is implicated in prion diseases, including Creutzfeldt-Jakob disease.

Additionally, the scope for exploring other biological data types, such as spatial-temporal patterns or host immune responses, remains vast. Transfer learning, where insights derived from related viral families could enhance our comprehension of influenza, also beckons exploration. Furthermore, the ever-evolving field of artificial intelligence paves the way for formulating dynamic models that can adapt in real-time to the mutable nature of viral evolution.

Future endeavours should also prioritise the development of intuitive tools and platforms grounded in these insights, facilitating wider scientific community engagement, fostering collaborative initiatives, and promoting shared discoveries. The confluence of biology and technology ushers in a promising future where our understanding, readiness, and response to infectious diseases are more informed, prompt, and efficacious.

In furthering these aims, future research could incorporate *few-shot learning* to enhance the adaptability of machine learning models to rapidly evolving viral mutations. This approach can significantly reduce the data required to train models effectively, enabling quicker responses to new threats. Additionally, the exploration of *continual learning* techniques can enable models to continuously update their knowledge base as new data become available, while also retaining previously learnt information, thus mitigating the risk of obsolescence and ensuring their relevance over time. Leveraging *multi-label data* can provide a more comprehensive understanding of the complex interactions between viral proteins and host cells, which could lead to more effective intervention strategies. Furthermore, the application of *meta-learning* approaches holds promise in developing models capable of generalising across different viral families and quickly adapting to novel pathogens, thus bolstering our preparedness for future epidemics and pandemics.

Bibliography

[1] Vaccination — ourworldindata.org. https://ourworldindata.org/vaccination. [Accessed 2023-11].

[2] Immunization coverage — who.int. https://www.who.int/en/news-room/fact-sheets/detail/immunization-coverage. [Accessed 2023-11].

[3] Can Chen, Xiaoxiao Liu, Danying Yan, Yuqing Zhou, Cheng Ding, Lu Chen, Lei Lan, Chenyang Huang, Daixi Jiang, Xiaobao Zhang, et al. Global influenza vaccination rates and factors associated with influenza vaccination. *International Journal of Infectious Diseases*, 125:153–163, 2022.

[4] Flu vaccination coverage, united states, 2020-21 influenza season | fluvaxview | seasonal influenza (flu) | cdc — cdc.gov. https://www.cdc.gov/flu/fluvaxview/coverage-2022estimates.htm. [Accessed 2023-11].

[5] Amit Dang and Jitendar Sharma. Assessing the low influenza vaccination coverage rate among healthcare personnel in india: A review of obstacles, beliefs, and strategies. *Value in Health Regional Issues*, 21:100–104, 2020.

[6] World bank open data — data.worldbank.org. https://data.worldbank.org/indicator/NY.GDP.PCAP.CD?locations=IN. [Accessed 2023-11].

[7] Janice C Pedersen. Hemagglutination-inhibition assay for influenza virus subtype identification and the detection and quantitation of serum antibodies to influenza virus. *Animal influenza virus*, pages 11–25, 2014.

[8] Overview of Influenza Testing Methods | CDC — cdc.gov. https://www.cdc.gov/flu/professionals/diagnosis/overview-testing-methods.htm. [Accessed 24-01-2024].

[9] Lisa R Clayville. Influenza update: a review of currently available vaccines. *Pharmacy and Therapeutics*, 36(10):659, 2011.

[10] Christine LP Eng, Joo Chuan Tong, and Tin Wee Tan. Predicting host tropism of influenza a virus proteins using random forest. *BMC medical genomics*, 7:1–11, 2014.

[11] Florian Mock, Adrian Viehweger, Emanuel Barth, and Manja Marz. Vidhop, viral host prediction with deep learning. *Bioinformatics*, 37(3):318–325, 2021.

[12] Pavan K Attaluri, Zhengxin Chen, and Guoqing Lu. Applying neural networks to classify influenza virus antigenic types and hosts. In *2010 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pages 1–6. IEEE, 2010.

[13] Reza Ahsan and Mansour Ebrahimi. The first implication of image processing techniques on influenza a virus sub-typing based on ha/na protein sequences, using convolutional deep neural network. *bioRxiv*, page 448159, 2018.

[14] Dan Scarafoni, Brian A Telfer, Darrell O Ricke, Jason R Thornton, and James Comolli. Predicting influenza a tropism with end-to-end learning of deep networks. *Health security*, 17(6):468–476, 2019.

[15] Yuan-Ling Xia, Weihua Li, Yongping Li, Xing-Lai Ji, Yun-Xin Fu, Shu-Qun Liu, et al. A deep learning approach for predicting antigenic variation of influenza a h3n2. *Computational and mathematical methods in medicine*, 2021, 2021.

[16] Guoqing Lu, Shunpu Zhang, and Xiang Fang. An improved string composition method for sequence comparison. *BMC bioinformatics*, 9(6):1–8, 2008.

[17] Anna Fabijańska and Szymon Grabowski. Viral genome deep classifier. *IEEE Access*, 7:81297–81307, 2019.

[18] Charalambos Chrysostomou, Floris Alexandrou, Mihalis A Nicolaou, and Huseyin Seker. Classification of influenza hemagglutinin protein sequences using convolutional neural networks. *arXiv preprint arXiv:2108.04240*, 2021.

[19] Fayroz F Sherif, NOURHAN Zayed, and MAHMOUD Fakhr. Classification of host origin in influenza a virus by transferring protein sequences into numerical feature vectors. *Int J Biol Biomed Eng*, 11(2), 2017.

[20] Eunmi Kwon, Myeongji Cho, Hayeon Kim, and Hyeon S Son. A study on host tropism determinants of influenza virus using machine learning. *Current Bioinformatics*, 15(2):121–134, 2020.

[21] Rui Yin, Xinrui Zhou, Shamima Rashid, and Chee Keong Kwoh. Hopper: an adaptive model for probability estimation of influenza reassortment through host prediction. *BMC medical genomics*, 13(1):1–13, 2020.

[22] Xinrui Zhou, Rui Yin, Chee-Keong Kwoh, and Jie Zheng. A context-free encoding scheme of protein sequences for predicting antigenicity of diverse influenza a viruses. *BMC genomics*, 19(10):145–154, 2018.

[23] Mansour Ebrahimi, Parisa Aghagolzadeh, Narges Shamabadi, Ahmad Tahmasebi, Mohammed Alsharifi, David L Adelson, Farhid Hemmatzadeh, and Esmaeil Ebrahimie. Understanding the underlying mechanism of ha-subtyping in the level of physic-chemical characteristics of protein. *PloS one*, 9(5):e96984, 2014.

[24] Fahad Humayun, Fatima Khan, Nasim Fawad, Shazia Shamas, Sahar Fazal, Abbas Khan, Arif Ali, Ali Farhan, and Dong-Qing Wei. Computational method for classification of avian influenza a virus using dna sequence information and physicochemical properties. *Frontiers in Genetics*, 12:599321, 2021.

[25] Rui Yin, Xinrui Zhou, Jie Zheng, and Chee Keong Kwoh. Computational identification of physicochemical signatures for host tropism of influenza a virus. *Journal of bioinformatics and computational biology*, 16(06):1840023, 2018.

[26] Jingxuan Qiu, Xinxin Tian, Yaxing Liu, Tianyu Lu, Hailong Wang, Zhuochen Shi, Sihao Lu, Dongpo Xu, and Tianyi Qiu. Univ-flu: A structure-based model of influenza a virus hemagglutinin for universal antigenic prediction. *Computational and Structural Biotechnology Journal*, 20:4656–4666, 2022.

[27] Jesse Vig, Ali Madani, Lav R Varshney, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani. Bertology meets biology: Interpreting attention in protein language models. *arXiv preprint arXiv:2006.15222*, 2020.

[28] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[29] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[30] Beibei Xu, Zhiying Tan, Kenli Li, Taijiao Jiang, and Yousong Peng. Predicting the host of influenza viruses based on the word vector. *PeerJ*, 5:e3579, 2017.

[31] Samia Tasnim Sara, Md Mehedi Hasan, Ahsan Ahmad, and Swakkhar Shatabda. Convolutional neural networks with image representation of amino acid sequences for protein function prediction. *Computational Biology and Chemistry*, 92:107494, 2021.

[32] Han Li and Fengzhu Sun. Comparative studies of alignment, alignment-free and svm based approaches for predicting the hosts of viruses based on viral sequences. *Scientific reports*, 8(1):10032, 2018.

[33] Christine LP Eng, Joo Chuan Tong, and Tin Wee Tan. Predicting zoonotic risk of influenza a viruses from host tropism protein signature using random forest. *International journal of molecular sciences*, 18(6):1135, 2017.

[34] Fatemeh Kargarfard, Ashkan Sami, Manijeh Mohammadi-Dehcheshmeh, and Esmaeil Ebrahimie. Novel approach for identification of influenza virus host range and zoonotic transmissible sequences by determination of host-related associative positions in viral genome segments. *BMC genomics*, 17(1):1–10, 2016.

[35] Pavan K Attaluri, Zhengxin Chen, Aruna M Weerakoon, and Guoqing Lu. Integrating decision tree and hidden markov model (hmm) for subtype prediction of human influenza a virus. In *International Conference on Multiple Criteria Decision Making*, pages 52–58. Springer, 2009.

[36] Mahmoud ElHefnawi and Fayroz F Sherif. Accurate classification and hemagglutinin amino acid signatures for influenza a virus host-origin association and subtyping. *Virology*, 449:328–338, 2014.

[37] Mohamed Amine Remita, Ahmed Halioui, Abou Abdallah Malick Diouara, Bruno Daigle, Golrokh Kiani, and Abdoulaye Banire Diallo. A machine learning approach for viral genome classification. *BMC bioinformatics*, 18:1–11, 2017.

[38] Daniel Struck, Glenn Lawyer, Anne-Marie Ternes, Jean-Claude Schmit, and Danielle Perez Bercoff. Comet: adaptive context-based modeling for ultrafast hiv-1 subtype identification. *Nucleic acids research*, 42(18):e144–e144, 2014.

[39] DS Hunnisett and WJ Teahan. Context-based methods for text categorisation. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 578–579, 2004.

[40] Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Peter Chen, John Canny, Pieter Abbeel, and Yun Song. Evaluating protein transfer learning with tape. *Advances in neural information processing systems*, 32, 2019.

[41] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[42] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. *Advances in neural information processing systems*, 28, 2015.

[43] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[44] Yanping Chen, Kai Wang, Weizhe Yang, Yongbin Qing, Ruizhang Huang, and Ping Chen. A multi-channel deep neural network for relation extraction. *IEEE Access*, 8:13195–13203, 2020.

[45] Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. Classifying relations via long short term memory networks along shortest dependency paths. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1785–1794, 2015.

[46] Linlin Wang, Zhu Cao, Gerard De Melo, and Zhiyuan Liu. Relation classification via multi-level attention cnns. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1298–1307, 2016.

[47] Yilong Yang, Qingfeng Wu, Ming Qiu, Yingdong Wang, and Xiaowei Chen. Emotion recognition from multi-channel eeg through parallel convolutional recurrent neural network. In *2018 international joint conference on neural networks (IJCNN)*, pages 1–7. IEEE, 2018.

[48] Yixin Cao, Zhiyuan Liu, Chengjiang Li, Juanzi Li, and Tat-Seng Chua. Multi-channel graph neural network for entity alignment. *arXiv preprint arXiv:1908.09898*, 2019.

[49] Weijiang Li, Fang Qi, Ming Tang, and Zhengtao Yu. Bidirectional lstm with self-attention mechanism and multi-channel features for sentiment classification. *Neurocomputing*, 387:63–77, 2020.

[50] Anjith George, Zohreh Mostaani, David Geissenbuhler, Olegs Nikisins, André Anjos, and Sébastien Marcel. Biometric face presentation attack detection with multi-channel convolutional neural network. *IEEE Transactions on Information Forensics and Security*, 15:42–55, 2019.

[51] Matthias Kerzel, Moaaz Ali, Hwei Geok Ng, and Stefan Wermter. Haptic material classification with a multi-channel neural network. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 439–446. IEEE, 2017.

[52] Yoon Kim. Convolutional neural networks for sentence classification, 2014.

[53] Haowei Zhang, Jin Wang, Jixian Zhang, and Xuejie Zhang. Ynu-hpcc at semeval 2017 task 4: Using a multi-channel cnn-lstm model for sentiment classification. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 796–801, 2017.

[54] Sebastian Ruder, Parsa Ghaffari, and John G. Breslin. Character-level and multi-channel convolutional neural networks for large-scale authorship attribution, 2016.

[55] Hyejung Chung and Kyung-shik Shin. Genetic algorithm-optimized multi-channel convolutional neural network for stock market prediction. *Neural Computing and Applications*, 32(12):7897–7914, 2020.

[56] Elmar Messner, Melanie Fediuk, Paul Swatek, Stefan Scheidl, Freyja-Maria Smolle-Jüttner, Horst Olschewski, and Franz Pernkopf. Multi-channel lung sound classification with convolutional recurrent neural networks. *Computers in Biology and Medicine*, 122:103831, 2020.

[57] Md Sirajus Salekin, Ghada Zamzmi, Dmitry Goldgof, Rangachar Kasturi, Thao Ho, and Yu Sun. Multi-channel neural network for assessing neonatal

pain from videos. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pages 1551–1556. IEEE, 2019.

[58] Hoang Quynh Le, Duy Cat Can, Tien Sinh Vu, Thanh Hai Dang, Mohammad Taher Pilehvar, and Nigel Collier. Large-scale exploration of neural relation classification architectures. 2018.

[59] Xiaolong Zheng, Peng Zheng, Liang Zheng, Yang Zhang, and Rui-Zhi Zhang. Multi-channel convolutional neural networks for materials properties prediction. *Computational Materials Science*, 173:109436, 2020.

[60] Rui Yin, Nyi Nyi Thwin, Pei Zhuang, Zhuoyi Lin, and Chee Keong Kwoh. Iav-cnn: a 2d convolutional neural network model to predict antigenic variants of influenza a virus. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 19(6):3497–3506, 2021.

[61] Mi Liu, Jingze Liu, Wenjun Song, Yousong Peng, Xiao Ding, Lizong Deng, and Taijiao Jiang. Development of predac-h1pdm to model the antigenic evolution of influenza a/(h1n1) pdm09 viruses. *Virologica Sinica*, 2023.

[62] Mi Liu, Xiang Zhao, Sha Hua, Xiangjun Du, Yousong Peng, Xiyan Li, Yu Lan, Dayan Wang, Aiping Wu, Yuelong Shu, et al. Antigenic patterns and evolution of the human influenza a (h1n1) virus. *Scientific reports*, 5(1):14171, 2015.

[63] Rui Yin, Viet Hung Tran, Xinrui Zhou, Jie Zheng, and Chee Keong Kwoh. Predicting antigenic variants of h1n1 influenza virus based on epidemics and pandemics using a stacking model. *PloS one*, 13(12):e0207777, 2018.

[64] Fujun Peng, Yuanling Xia, and Weihua Li. Prediction of antigenic distance in influenza a using attribute network embedding. *Viruses*, 15(7):1478, 2023.

[65] Yuhua Yao, Xianhong Li, Bo Liao, Li Huang, Pingan He, Fayou Wang, Jiasheng Yang, Hailiang Sun, Yulong Zhao, and Jialiang Yang. Predicting influenza antigenicity from hemagglutintin sequence data based on a joint random forest method. *Scientific reports*, 7(1):1545, 2017.

[66] Yousong Peng, Dayan Wang, Jianhong Wang, Kenli Li, Zhongyang Tan, Yuelong Shu, and Taijiao Jiang. A universal computational model for predicting antigenic variants of influenza a virus based on conserved antigenic structures. *Scientific reports*, 7(1):42051, 2017.

[67] Jing Li, Sen Zhang, Bo Li, Yi Hu, Xiao-Ping Kang, Xiao-Yan Wu, Meng-Ting Huang, Yu-Chang Li, Zhong-Peng Zhao, Cheng-Feng Qin, et al. Machine learning methods for predicting human-adaptive influenza a viruses based on viral nucleotide compositions. *Molecular biology and evolution*, 37(4):1224–1236, 2020.

[68] Xiaoli Qiang and Zheng Kou. Prediction of interspecies transmission for avian influenza a virus based on a back-propagation neural network. *Mathematical and computer modelling*, 52(11-12):2060–2065, 2010.

[69] Christine LP Eng, Joo Chuan Tong, and Tin Wee Tan. Distinct host tropism protein signatures to identify possible zoonotic influenza a viruses. *PloS one*, 11(2):e0150173, 2016.

[70] Yanhua Xu and Dominik Wojtczak. Predicting influenza a viral host using pssm and word embeddings. In *2021 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 1–10. IEEE, 2021.

[71] Yanhua Xu. and Dominik Wojtczak. End-to-end multi-channel neural networks for predicting influenza a virus hosts and antigenic types. In *Proceedings of the 14th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management - KDIR,*, pages 40–50. INSTICC, SciTePress, 2022.

[72] Yanhua Xu and Dominik Wojtczak. Dive into machine learning algorithms for influenza virus host prediction with hemagglutinin sequences. *Biosystems*, 220:104740, 2022.

[73] Yanhua Xu and Dominik Wojtczak. Mc-nn: An end-to-end multi-channel neural network approach for predicting influenza a virus hosts and antigenic types. *SN Computer Science*, 4(5):435, 2023.

[74] Who influenza (seasonal). https://www.who.int/news-room/fact-sheets/detail/influenza-(seasonal)#:~:text=There%20are%20around%20a%20billion,650%20000%20respiratory%20deaths%20annually. [Accessed 2023-11].

[75] Wang Jae Lee and Wang Jae Lee. Common cold and flu. *Vitamin C in Human Health and Disease: Effects, Mechanisms of Action, and New Guidance on Intake*, pages 89–100, 2019.

[76] Ron Eccles. Understanding the symptoms of the common cold and influenza. *The Lancet infectious diseases*, 5(11):718–725, 2005.

[77] Sebastian Johnston and Stephen Holgate. Epidemiology of viral respiratory tract infections. *Viral and other infections of the human respiratory tract*, pages 1–38, 1996.

[78] Yoshihiro Kawaoka et al. *Influenza virology: current topics.* Caister Academic Press, 2006.

[79] CDC. Types of Influenza Viruses — cdc.gov. https://www.cdc.gov/flu/about/viruses/types.htm. [Accessed 2023-11].

[80] Suxiang Tong, Xueyong Zhu, Yan Li, Mang Shi, Jing Zhang, Melissa Bourgeois, Hua Yang, Xianfeng Chen, Sergio Recuenco, Jorge Gomez, et al. New world bats harbor diverse influenza a viruses. *PLoS pathogens*, 9(10):e1003657, 2013.

[81] Mark B Carascal, Rance Derrick N Pavon, and Windell L Rivera. Recent progress in recombinant influenza vaccine development toward heterosubtypic immune response. *Frontiers in Immunology*, 13:878943, 2022.

[82] Mark B Carascal, Rance Derrick N Pavon, and Windell L Rivera. Corrigendum: Recent progress in recombinant influenza vaccine development toward heterosubtypic immune response. *Frontiers in Immunology*, 13:948031, 2022.

[83] Influenza A Subtypes and the Species Affected | Seasonal Influenza (Flu) | CDC — cdc.gov. https://www.cdc.gov/flu/other/animal-flu.html. [Accessed 11-11-2023].

[84] Yuelong Shu and John McCauley. Gisaid: Global initiative on sharing all influenza data–from vision to reality. *Eurosurveillance*, 22(13):30494, 2017.

[85] Gerardo Chowell, Sherry Towers, Cécile Viboud, Rodrigo Fuentes, Viviana Sotomayor, Lone Simonsen, Mark A Miller, Mauricio Lima, Claudia Villarroel, Monica Chiu, et al. The influence of climatic conditions on the transmission dynamics of the 2009 a/h1n1 influenza pandemic in chile. *BMC infectious diseases*, 12(1):1–12, 2012.

[86] Porter Hoagland, Di Jin, Lara Y Polansky, Barbara Kirkpatrick, Gary Kirkpatrick, Lora E Fleming, Andrew Reich, Sharon M Watkins, Steven G Ullmann, and Lorraine C Backer. The costs of respiratory illnesses arising from

florida gulf coast karenia brevis blooms. *Environmental Health Perspectives*, 117(8):1239–1243, 2009.

[87] Stephen Corson, Chris Robertson, Arlene Reynolds, and Jim McMenamin. Modelling the population effectiveness of the national seasonal influenza vaccination programme in scotland: The impact of targeting all individuals aged 65 years and over. *Influenza and Other Respiratory Viruses*, 13(4):354–363, 2019.

[88] Christa K Irwin, Kyoung-Jin Yoon, Chong Wang, Steven J Hoff, Jeffrey J Zimmerman, T Denagamage, and AM O'Connor. Using the systematic review methodology to evaluate factors that influence the persistence of influenza virus in environmental matrices. *Applied and environmental microbiology*, 77(3):1049–1060, 2011.

[89] Jeffery K Taubenberger and John C Kash. Influenza virus evolution, host adaptation, and pandemic formation. *Cell host & microbe*, 7(6):440–451, 2010.

[90] Rina Fajri Nuwarda, Abdulsalam Abdullah Alharbi, and Veysel Kayser. An overview of influenza viruses and vaccines. *Vaccines*, 9(9):1032, 2021.

[91] Jeffery K Taubenberger, John C Kash, and David M Morens. The 1918 influenza pandemic: 100 years of questions answered and unanswered. *Science translational medicine*, 11(502):eaau5485, 2019.

[92] Jason S Long, Bhakti Mistry, Stuart M Haslam, and Wendy S Barclay. Host and viral determinants of influenza a virus species specificity. *Nature Reviews Microbiology*, 17(2):67–81, 2019.

[93] John Steel and Anice C Lowen. Influenza a virus reassortment. *Influenza Pathogenesis and Control-Volume I*, pages 377–401, 2014.

[94] George K Hirst. The agglutination of red cells by allantoic fluid of chick embryos infected with influenza virus. *Science*, 94(2427):22–23, 1941.

[95] Antigenic Characterization | CDC — cdc.gov. https://www.cdc.gov/flu/about/professionals/antigenic.htm. [Accessed 2024-02].

[96] figdraw.com. https://www.figdraw.com. [Accessed 2023-10].

[97] Mary Zacour, Brian J Ward, Angela Brewer, Patrick Tang, Guy Boivin, Yan Li, Michelle Warhuus, Shelly A McNeil, Jason J LeBlanc, and Todd F Hatchette. Standardization of hemagglutination inhibition assay for influenza serology allows for high reproducibility between laboratories. *Clinical and Vaccine Immunology*, 23(3):236–242, 2016.

[98] Iain Stephenson, Alan Heath, Diane Major, Robert W Newman, Katja Hoschler, Wang Junzi, Jacqueline M Katz, Jerry P Weir, Maria C Zambon, and John M Wood. Reproducibility of serologic assays for influenza virus a (h5n1). *Emerging infectious diseases*, 15(8):1250, 2009.

[99] Food, Drug Administration, et al. Guidance for industry: clinical data needed to support the licensure of seasonal inactivated influenza vaccines. *Rockv. US Dep. Heal. Hum. Serv*, 2007.

[100] Iain Stephenson, Rose Gaines Das, John M Wood, and Jacqueline M Katz. Comparison of neutralising antibody assays for detection of antibody to influenza a/h3n2 viruses: an international collaborative study. *Vaccine*, 25(20):4056–4063, 2007.

[101] JM Wood, RE Gaines-Das, J Taylor, and P Chakraverty. Comparison of influenza serological techniques by international collaborative study. *Vaccine*, 12(2):167–174, 1994.

[102] David E Swayne, David L Suarez, Erica Spackman, Samadhan Jadhao, Gwenaelle Dauphin, Mia Kim-Torchetti, James McGrane, John Weaver, Peter Daniels, Frank Wong, et al. Antibody titer has positive predictive value for vaccine protection against challenge with natural antigenic-drift variants of h5n1 high-pathogenicity avian influenza viruses from indonesia. *Journal of virology*, 89(7):3746–3762, 2015.

[103] R Webster, N Cox, and K Stohr. Who manual on animal influenza diagnosis and surveillance, who/ cds/ csr/ ncs/ 2002.5. 2002.

[104] Danuta M Skowronski, Travis S Hottes, Naveed Z Janjua, Dale Purych, Suzana Sabaiduc, Tracy Chan, Gaston De Serres, Jennifer Gardy, Janet E McElhaney, David M Patrick, et al. Prevalence of seroprotection against the pandemic (h1n1) virus after the 2009 pandemic. *Cmaj*, 182(17):1851–1856, 2010.

[105] Michael W Deem. Entropy, disease, and new opportunities for chemical engineering research. *AIChE journal*, 51(12):3086–3090, 2005.

[106] Vishal Gupta, David J Earl, and Michael W Deem. Quantifying influenza vaccine efficacy and antigenic distance. *Vaccine*, 24(18):3881–3888, 2006.

[107] Enrique T Munoz and Michael W Deem. Epitope analysis for influenza vaccine design. *Vaccine*, 23(9):1144–1148, 2005.

[108] Jonas E Salk et al. A critique of serologie methods for the study of influenza viruses. *Archiv fur die gesamte Virusforschung*, 4(4):476–84, 1951.

[109] Wilfred Ndifon, Jonathan Dushoff, and Simon A Levin. On the use of hemagglutination-inhibition for influenza surveillance: surveillance data are predictive of influenza vaccine effectiveness. *Vaccine*, 27(18):2447–2452, 2009.

[110] Kotone Sano and Haruko Ogawa. Hemagglutination (inhibition) assay. *Lectins: Methods and Protocols*, pages 47–52, 2014.

[111] Erica Spackman and Ioannis Sitaras. Hemagglutination inhibition assay. *Animal Influenza Virus: Methods and Protocols*, pages 11–28, 2020.

[112] Christoph Molnar. *Interpretable Machine Learning*. 2 edition, 2022.

[113] Ipc. https://ipc.mit.edu/research/work-of-the-future/, Oct 2023.

[114] Andrew P Hederman and Margaret E Ackerman. Leveraging deep learning to improve vaccine design. *Trends in Immunology*, 2023.

[115] NN SVG — alexlenail.me. http://alexlenail.me/NN-SVG/index.html. [Accessed 2023-10].

[116] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[117] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA, 2016. Association for Computing Machinery.

[118] Chris Seiffert, Taghi M Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. Rusboost: Improving classification performance when training data is skewed. In *2008 19th International Conference on Pattern Recognition*, pages 1–4. IEEE, 2008.

[119] Nitesh V Chawla, Aleksandar Lazarevic, Lawrence O Hall, and Kevin W Bowyer. Smoteboost: Improving prediction of the minority class in boosting. In *European conference on principles of data mining and knowledge discovery*, pages 107–119. Springer, 2003.

[120] Robert Gove and Jorge Faytong. Machine learning and event-based software testing: classifiers for identifying infeasible gui event sequences. In *Advances in Computers*, volume 86, pages 109–135. Elsevier, 2012.

[121] Marvin Minsky and Seymour A Papert. *Perceptrons: An introduction to computational geometry*. MIT press, 2017.

[122] dair-ai/ml-visuals. https://github.com/dair-ai/ml-visuals. [Accessed 06-11-2023].

[123] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[124] Sepp Hochreiter and Jürgen Schmidhuber. Lstm can solve hard long time lag problems. *Advances in neural information processing systems*, pages 473–479, 1997.

[125] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[126] Alex Graves, Nicole Beringer, and Juergen Schmidhuber. Rapid retraining on speech data with lstm recurrent networks. *Technical Report IDSIA-09–05, IDSIA*, 2005.

[127] Trias Thireou and Martin Reczko. Bidirectional long short-term memory networks for predicting the subcellular localization of eukaryotic proteins. *IEEE/ACM transactions on computational biology and bioinformatics*, 4(3):441–446, 2007.

[128] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches, 2014.

[129] Gavin C Cawley and Nicola LC Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *The Journal of Machine Learning Research*, 11:2079–2107, 2010.

[130] Tim Head, Manoj Kumar, Holger Nahrstaedt, Gilles Louppe, and Iaroslav Shcherbatyi. scikit-optimize/scikit-optimize, September 2020.

[131] Tom O'Malley, Elie Bursztein, James Long, François Chollet, Haifeng Jin, Luca Invernizzi, et al. Kerastuner. https://github.com/keras-team/keras-tuner, 2019.

[132] Apoorv Agnihotri and Nipun Batra. Exploring bayesian optimization. *Distill*, 2020. https://distill.pub/2020/bayesian-optimization.

[133] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2015.

[134] Eric Brochu, Vlad M Cora, and Nando De Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

[135] Paula Branco, Luís Torgo, and Rita P Ribeiro. A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys (CSUR)*, 49(2):1–50, 2016.

[136] Razvan Bunescu, Ruifang Ge, Rohit J Kate, Edward M Marcotte, Raymond J Mooney, Arun K Ramani, and Yuk Wah Wong. Comparative experiments on learning information extractors for proteins and their interactions. *Artificial intelligence in medicine*, 33(2):139–155, 2005.

[137] Joseph Bockhorst and Mark Craven. Markov networks for detecting overlapping elements in sequence data. *Advances in Neural Information Processing Systems*, 17:193–200, 2005.

[138] Mark Goadrich, Louis Oliphant, and Jude Shavlik. Learning ensembles of first-order clauses for recall-precision curves: A case study in biomedical

information extraction. In *International Conference on Inductive Logic Programming*, pages 98–115. Springer, 2004.

[139] Jesse Davis, Elizabeth S Burnside, Inês de Castro Dutra, David Page, Raghu Ramakrishnan, Vitor Santos Costa, and Jude W Shavlik. View learning for statistical relational learning: With an application to mammography. In *IJCAI*, pages 677–683. Citeseer, 2005.

[140] Josephine Akosa. Predictive accuracy: A misleading performance measure for highly imbalanced data. In *Proceedings of the SAS Global Forum*, volume 12, 2017.

[141] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240, 2006.

[142] Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3):e0118432, 2015.

[143] Wanhua Su, Yan Yuan, and Mu Zhu. A relationship between the average precision and the area under the roc curve. In *Proceedings of the 2015 International Conference on The Theory of Information Retrieval*, pages 349–352, 2015.

[144] Influenza Virus Genome Sequencing and Genetic Characterization | CDC — cdc.gov. https://www.cdc.gov/flu/about/professionals/genetic-characterization.htm. [Accessed 2024-02].

[145] IUPAC Codes — bioinformatics.org. https://www.bioinformatics.org/sms/iupac.html. [Accessed 2023-11].

[146] Richard S Larson and Tudor I Oprea. *Bioinformatics and drug discovery*. Springer, 2006.

[147] Memorandaarestate-Les Memorandums. A revision of the system of nomenclature for influenza viruses: a who memorandum. *Bulletin ofthe World Health Organization*, 58(4):585–591, 1980.

[148] HU Mei, Zhi H Liao, Yuan Zhou, and Shengshi Z Li. A new set of amino acid descriptors and its application in peptide qsars. *Peptide Science: Original Research on Biomolecules*, 80(6):775–786, 2005.

[149] Maria Sandberg, Lennart Eriksson, Jörgen Jonsson, Michael Sjöström, and Svante Wold. New chemical descriptors relevant for the design of biologically active peptides. a multivariate characterization of 87 amino acids. *Journal of medicinal chemistry*, 41(14):2481–2491, 1998.

[150] Feifei Tian, Peng Zhou, and Zhiliang Li. T-scale as a novel vector of topological descriptors for amino acids and its application in qsars of peptides. *Journal of molecular structure*, 830(1-3):106–115, 2007.

[151] Li Yang, Mao Shu, Kaiwang Ma, Hu Mei, Yongjun Jiang, and Zhiliang Li. St-scale as a novel amino acid descriptor and its application in qsam of peptides and analogues. *Amino acids*, 38:805–816, 2010.

[152] Alexander G Georgiev. Interpretable numerical descriptors of amino acid space. *Journal of Computational Biology*, 16(5):703–723, 2009.

[153] Steven Henikoff and Jorja G Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, 89(22):10915–10919, 1992.

[154] Andrea Zaliani and Emanuela Gancia. Ms-whim scores for amino acids: a new 3d-description for peptide qsar and qspr studies. *Journal of chemical information and computer sciences*, 39(3):525–533, 1999.

[155] Gerard JP van Westen, Remco F Swier, Jörg K Wegner, Adriaan P IJzerman, Herman WT van Vlijmen, and Andreas Bender. Benchmarking of protein descriptor sets in proteochemometric modeling (part 1): comparative study of 13 amino acid descriptor sets. *Journal of cheminformatics*, 5(1):1–11, 2013.

[156] Gerard JP van Westen, Remco F Swier, Isidro Cortes-Ciriano, Jörg K Wegner, John P Overington, Adriaan P IJzerman, Herman WT van Vlijmen, and Andreas Bender. Benchmarking of protein descriptor sets in proteochemometric modeling (part 2): modeling performance of 13 amino acid descriptor sets. *Journal of cheminformatics*, 5:1–20, 2013.

[157] Zhen Chen, Pei Zhao, Fuyi Li, André Leier, Tatiana T Marquez-Lago, Yanan Wang, Geoffrey I Webb, A Ian Smith, Roger J Daly, Kuo-Chen Chou, et al. ifeature: a python package and web server for features extraction and selection from protein and peptide sequences. *Bioinformatics*, 34(14):2499–2502, 2018.

[158] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[159] Stephen F Altschul, Thomas L Madden, Alejandro A Schäffer, Jinghui Zhang, Zheng Zhang, Webb Miller, and David J Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic acids research*, 25(17):3389–3402, 1997.

[160] Stephen F Altschul and Eugene V Koonin. Iterated profile searches with psi-blast—a tool for discovery in protein databases. *Trends in biochemical sciences*, 23(11):444–447, 1998.

[161] Yi An, Jiawei Wang, Chen Li, Andre Leier, Tatiana Marquez-Lago, Jonathan Wilksch, Yang Zhang, Geoffrey I Webb, Jiangning Song, and Trevor Lithgow. Comprehensive assessment and performance improvement of effector protein predictors for bacterial secretion systems iii, iv and vi. *Briefings in Bioinformatics*, 19(1):148–161, 2018.

[162] Jiawei Wang, Bingjiao Yang, Jerico Revote, Andre Leier, Tatiana T Marquez-Lago, Geoffrey Webb, Jiangning Song, Kuo-Chen Chou, and Trevor Lithgow. Possum: a bioinformatics toolkit for generating numerical sequence feature descriptors based on pssm profiles. *Bioinformatics*, 33(17):2756–2758, 2017.

[163] Shandar Ahmad and Akinori Sarai. Pssm-based prediction of dna binding sites in proteins. *BMC bioinformatics*, 6:1–6, 2005.

[164] GISAID. Initiative.

[165] David JD Earn, Jonathan Dushoff, and Simon A Levin. Ecology and evolution of the flu. *Trends in ecology & evolution*, 17(7):334–340, 2002.

[166] BUTTER WORTHS. A one-letter notation for amino acid sequences.

[167] Limin Fu, Beifang Niu, Zhengwei Zhu, Sitao Wu, and Weizhong Li. Cd-hit: accelerated for clustering the next-generation sequencing data. *Bioinformatics*, 28(23):3150–3152, 2012.

[168] Weizhong Li and Adam Godzik. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22(13):1658–1659, 2006.

[169] Blast. http://ftp.ncbi.nih.gov/blast/executables.

[170] Tanping Li, Ke Fan, Jun Wang, and Wei Wang. Reduction of protein sequence complexity by residue grouping. *Protein Engineering*, 16(5):323–330, 2003.

[171] Vijayakumar Saravanan and Namasivayam Gautham. Harnessing computational biology for exact linear b-cell epitope prediction: a novel amino acid composition-based feature descriptor. *Omics: a journal of integrative biology*, 19(10):648–658, 2015.

[172] Taigang Liu, Xiaoqi Zheng, and Jun Wang. Prediction of protein structural class for low-similarity sequences using support vector machine and psi-blast profile. *Biochimie*, 92(10):1330–1334, 2010.

[173] Shuyan Ding, Yan Li, Zhuoxing Shi, and Shoujiang Yan. A protein structural classes prediction method based on predicted secondary structure and psi-blast profile. *Biochimie*, 97:60–65, 2014.

[174] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[175] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017.

[176] François Chollet et al. Keras. https://keras.io, 2015.

[177] Y Pan, S Cui, Y Sun, X Zhang, C Ma, W Shi, X Peng, G Lu, D Zhang, Y Liu, et al. Human infection with h9n2 avian influenza in northern china. *Clinical Microbiology and Infection*, 24(3):321–323, 2018.

[178] Varsha Potdar, Dilip Hinge, Ashish Satav, Eric AF Simões, Pragya D Yadav, and Mandeep S Chadha. Laboratory-confirmed avian influenza a (h9n2) virus infection, india, 2019. *Emerging Infectious Diseases*, 25(12):2328, 2019.

[179] Guo Zhao, Qunping Fan, Lei Zhong, Yanfang Li, Wenbo Liu, Xiaowen Liu, Song Gao, Daxin Peng, and Xiufan Liu. Isolation and phylogenetic analysis of pandemic h1n1/09 influenza virus from swine in jiangsu province of china. *Research in veterinary science*, 93(1):125–132, 2012.

[180] David T Jones. Protein secondary structure prediction based on position-specific scoring matrices. *Journal of molecular biology*, 292(2):195–202, 1999.

[181] R Burke Squires, Jyothi Noronha, Victoria Hunt, Adolfo García-Sastre, Catherine Macken, Nicole Baumgarth, David Suarez, Brett E Pickett, Yun Zhang, Christopher N Larsen, et al. Influenza research database: an integrated bioinformatics resource for influenza research and surveillance. *Influenza and other respiratory viruses*, 6(6):404–416, 2012.

[182] National center for biotechnology information. https://www.ncbi.nlm.nih.gov. [Accessed 2023-11].

[183] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

[184] Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, et al. Prottrans: Toward understanding the language of life through self-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(10):7112–7127, 2021.

[185] Ehsaneddin Asgari and Mohammad RK Mofrad. Continuous distributed representation of biological sequences for deep proteomics and genomics. *PloS one*, 10(11):e0141287, 2015.

[186] Facebookresearch. Facebookresearch/esm: Evolutionary scale modeling (esm): Pretrained language models for proteins. https://github.com/facebookresearch/esm#esmfold. [Accessed: 2023-09].

[187] facebook/esm2_t30_150m_ur50d - hugging face. https://huggingface.co/facebook/esm2_t30_150M_UR50D. [Accessed: 2023-09].

[188] Rostlab/prot_bert - hugging face. https://huggingface.co/Rostlab/prot_bert. [Accessed: 2023-09].

[189] Stephen F Altschul, Warren Gish, Webb Miller, Eugene W Myers, and David J Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.

[190] Christiam Camacho, George Coulouris, Vahram Avagyan, Ning Ma, Jason Papadopoulos, Kevin Bealer, and Thomas L Madden. Blast+: architecture and applications. *BMC bioinformatics*, 10:1–9, 2009.

[191] Abdoelnaser M Degoot, Emmanuel S Adabor, Faraimunashe Chirove, and Wilfred Ndifon. Predicting antigenicity of influenza a viruses using biophysical ideas. *Scientific reports*, 9(1):10218, 2019.

[192] Yu-Chieh Liao, Min-Shi Lee, Chin-Yu Ko, and Chao A Hsiung. Bioinformatics models for predicting antigenic variants of influenza a/h3n2 virus. *Bioinformatics*, 24(4):505–512, 2008.

[193] Robin M Bush, Catherine A Bender, Kanta Subbarao, Nancy J Cox, and Walter M Fitch. Predicting the evolution of human influenza a. *Science*, 286(5446):1921–1925, 1999.

[194] Robin M Bush, Walter M Fitch, Catherine A Bender, and Nancy J Cox. Positive selection on the h3 hemagglutinin gene of human influenza virus a. *Molecular biology and evolution*, 16(11):1457–1465, 1999.

[195] Joshua B Plotkin and Jonathan Dushoff. Codon bias and frequency-dependent selection on the hemagglutinin epitopes of influenza a virus. *Proceedings of the National Academy of Sciences*, 100(12):7152–7157, 2003.

[196] Min-Shi Lee and Jack Si-En Chen. Predicting antigenic variants of influenza a/h3n2 viruses. *Emerging infectious diseases*, 10(8):1385, 2004.

[197] Italo Archetti and Frank L Horsfall Jr. Persistent antigenic variation of influenza a viruses after incomplete neutralization in ovo with heterologous immune serum. *The Journal of experimental medicine*, 92(5):441–462, 1950.

[198] Dengyong Zhou, Olivier Bousquet, Thomas Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. *Advances in neural information processing systems*, 16, 2003.

[199] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd annual meeting of the association for computational linguistics*, pages 189–196, 1995.

[200] Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-

Rosen, Weihua Hu, et al. Graphcast: Learning skillful medium-range global weather forecasting. *arXiv preprint arXiv:2212.12794*, 2022.

[201] Nidhi Shah, Michael G Nute, Tandy Warnow, and Mihai Pop. Misunderstood parameter of ncbi blast impacts the correctness of bioinformatics workflows. *Bioinformatics*, 35(9):1613–1614, 2019.

[202] Alexander Pertsemlidis and John W Fondon. Having a blast with bioinformatics (and avoiding blastphemy). *Genome biology*, 2(10):1–10, 2001.

[203] Eleca J Dunham, Vivien G Dugan, Emilee K Kaser, Sarah E Perkins, Ian H Brown, Edward C Holmes, and Jeffery K Taubenberger. Different evolutionary trajectories of european avian-like and classical swine h1n1 influenza a viruses. *Journal of virology*, 83(11):5485–5494, 2009.

[204] Yi Liu. A code within the genetic code: codon usage regulates co-translational protein folding. *Cell Communication and Signaling*, 18(1):1–9, 2020.