Development of Machine Learning Models for Predicting Mutation Likelihood in SARS-CoV-2 M^{PRO} and the NSP10-NSP16 Complex Using Molecular Dynamics Simulation Data

Submitted in partial fulfilment

of the requirements for the degree of

Master of Science in Bioinformatics (Coursework and Thesis)

of Rhodes University

Emily Morgan

Supervisor: Prof. Özlem Tastan Bishop

Grahamstown, South Africa

April 9, 2025

Declaration of Authorship

I, Emily Morgan, declare that "Development of Machine Learning Models for Predicting Mutation Likelihood in SARS-CoV-2 M^{pro} and the NSP10-NSP16 Complex Using Molecular Dynamics Simulation Data" is my own work and that it has not been submitted, in whole or in part, for any degree in any other University. I have indicated by reference and acknowledgement the areas that are not my own work.

Date: 6 February 2025

Abstract

The COVID-19 pandemic, caused by the SARS-CoV-2 virus, highlights the critical need for innovative methods to understand virus evolution and develop effective treatments. Mutations in SARS-CoV-2 proteins can increase virulence, prevent virus detection, and reduce the efficacy of treatments and vaccines. While SARS-CoV-2 mutation research generally focuses on the spike protein, some non-structural proteins (NSPs) warrant attention, such as NSP10, NSP16 and main protease (M^{pro}), also known as the 3C-like protease ($3CL^{pro}$). These proteins are essential to the replication and immune capabilities of viruses, making them valuable targets for viral therapies.

This study begins with an extension of the residue mutation predictions performed in Barozi *et al.* (2024), where the Python artificial neural network (ANN) and random forest (RF) models we had developed were fine-tuned and additional support vector machine (SVM) models were produced. All models were trained using the original M^{pro} dataset from Barozi *et al.* (2024), achieving moderate performance with an average accuracy of up to 76% on test subsets. In an attempt to improve the mutation prediction performance, an alternative dataset using raw M^{pro} MD trajectory coordinates was processed using convolutional neural networks (CNNs). However, the CNNs performed worse than the models trained on the processed M^{pro} trajectory data. Finally, the generalisibility of the ANN, RF and SVM models when applied to other SARS-CoV-2 protein data was investigated using the NSP10-NSP16 complex. To obtain a comparable dataset, molecular dynamics (MD) simulations of the NSP10-NSP16 complex were conducted with and without the SAM ligand. Stable trajectories were analysed through dynamic residue network (DRN) analysis, root mean square fluctuation (RMSF), solvent accessible surface area (SASA),

B-factor, and BLOcks SUbstitution Matrix (BLOSUM) metrics to create machine learning (ML) input datasets for NSP10 and NSP16. These datasets were tested using the M^{pro}-trained models, resulting in a decline in performance compared to the M^{pro} test sets, indicating limited transferability.

This study identified critical ML-based residue mutation prediction limitations, including small datasets, class imbalances, and structural instabilities during molecular dynamics simulations. However, it established a foundation for further research by demonstrating the importance of feature selection and the potential of ML models to predict viral residue mutations.

Acknowledgements

I would like to express my gratitude to those whose support and expertise have been invaluable throughout the course of this project. Their contributions have been fundamental to the successful completion of this research.

First and foremost, I thank Prof. Özlem Tastan Bishop, my supervisor, for her insightful guidance, constructive feedback, and unwavering support, which have been instrumental in shaping this research.

I am also deeply appreciative of Assistant Prof. Prashant Singh for his invaluable expertise and guidance in developing the workflow for testing the machine-learning models. His input for designing and building the CNN model significantly enhanced the computational aspects of this study.

Special thanks go to Prof. Kevin Lobb, who generously offered his expertise in parameterising the zinc-binding sites of the NSP10 protein, a critical component of this work. I also extend my appreciation to Rabelani Ramahala for her assistance in this parameterisation work, which supported the progress of this study.

A big thanks also goes to Dr. Allan Sanyanga for his assistance in parameterising the SAM ligand and ensuring the accuracy of this essential step. I am also grateful to Rehema Mukami and Dr Sanyanga for their invaluable input after their meticulous review of the molecular dynamics simulation parameters.

I would also like to extend my sincere thanks to the AI team of the Research Unit in Bioinformatics (RUBi) at Rhodes University. I have had the privilege of building on the

valuable work previously conducted together, including the ML models and M^{pro} dataset created. In particular, I wish to thank Shaylyn Govender for providing me with the scripts necessary for running the DRN analysis and providing further detail on the methodology used when processing the M^{pro} data.

I am profoundly grateful to my colleagues in RUBi at Rhodes University for fostering a collaborative and intellectually stimulating environment. I also cherish the time I spent at the Scientific Computing Division at Uppsala University, where I not only gained valuable insights but also created fond memories with the program's participants.

I acknowledge the Centre for High Performance Computing (CHPC), South Africa, for providing computational resources to this research project. In addition, the CNN model implementation was enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

Lastly, I thank my family for their unwavering encouragement and support throughout this journey. Their belief in me has been a constant source of motivation.

Contents

List of Figures

List of Tables

Declaration of Publications

The following research paper has been accepted and presented for publication, and parts of its materials are included in the thesis:

1. V. Barozi, S. Chakraborty, S. Govender, **E. Morgan**, R. Ramahala, S. C. Graham, N. T. Bishop, and Ö. T. Bishop. Revealing SARS-CoV-2 Mpro mutation cold and hot spots: Dynamic residue network analysis meets machine learning. *Computational and Structural Biotechnology Journal*, 23:3800–3816, 2024. ISSN 2001-0370. doi:10.1016/j.csbj.2024.10.031.

For this publication, my contributions included:

- Creation of the Python artificial neural networks
- Assisting in the application of statistical tests to analyze the distribution of mutating residues based on dynamic residue network metrics

List of Abbreviations

ANN	Artificial Neural Network
AUC	Area Under the ROC Curve
ACPYPE	Antechamber Python Parser Interface
BC	Betweenness Centrality
BLOSUM	Blocks Substitution Matrix
CC	Closeness Centrality
CHPC	Centre for High Performance Computing
CNN	Convolutional Neural Network
DC	Degree Centrality
DRN	Dynamic Residue Network
EC	Eigencentrality
ECC	Eccentricity
GISAID	Global Initiative on Sharing All Influenza Data
KC	Katz Centrality
L	Farness
MD	Molecular Dynamics
ML	Machine Learning
M^{pro}	Main Protease
NSP	Non-Structural Protein
PCA	Principal Component Analysis
PDB	Protein Data Bank
PR	Page Rank

RF	Random Forest
RIN	Residue Interaction Network
RMSD	Root Mean Square Deviation
RMSF	Root Mean Square Fluctuation
ROC	Receiver Operating Characteristic
SAM	S-Adenosyl Methionine
SARS-CoV-2	Severe Acute Respiratory Syndrome Coronavirus 2
SASA	Solvent Accessible Surface Area
SNV	Single Nucleotide Variant
SVM	Support Vector Machine
UniProt	Universal Protein Resource
VoC	Variant of Concern

Chapter 1

Concepts and Literature Review

1.1 Background

The COVID-19 pandemic, triggered by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2), has impacted both global health and socio-economic stability (Gorbalenya *et al.*, 2020). As of April 2024, over 755 million cases and more than 7 million deaths have been reported worldwide[1]. This pandemic underscores the need for advanced scientific approaches to understand the virus, develop effective treatments, and adapt to its evolving nature. During the course of the Covid-19 pandemic, the virus has mutated due to replication errors, resulting in numerous SARS-CoV-2 variants and variants of concern (VOCs) (Saldivar-Espinoza *et al.*, 2022). These variants may result in increased virulence or more severe symptoms, detection of the virus may be compromised, or even reduce the effectiveness of treatments and vaccines. To create antiviral therapies that will remain effective for the current and future variants of SARS-CoV-2, potential amino acid mutations of proteins targeted by antiviral agents must be predicted.

SARS-CoV-2 is a positive-sense single-stranded RNA virus belonging to the *Coronaviridae* family (Li *et al.*, 2023). Its genome encodes a range of structural, non-structural (NSP), and accessory proteins essential for its replication and pathogenicity. Among these, the

[1]https://data.who.int/dashboards/covid19/cases?n=c

NSP10-NSP16 complex is crucial to the virus's ability to evade the host immune system (Chen *et al.*, 2011). NSP16 is an RNA methyltransferase that, in collaboration with its cofactor NSP10, catalyses the 2'-O-methylation of the 5' cap structure of viral mRNA. This methylation is vital as it mimics the host mRNA cap structure, preventing the viral RNA from being recognised and degraded by the host's immune system (Sk *et al.*, 2020).

In addition to NSP16, the main protease (M^{pro}) plays a critical role in SARS-CoV-2 replication (Hu *et al.*, 2022). M^{pro} is a cysteine hydrolase, which is responsible for cleaving viral polyproteins into functional non-structural proteins required for the formation of the replication-transcription complex. Without M^{pro} activity, the polyproteins of the SARS-CoV-2 virus cannot be processed, effectively halting replication. Due to the essential role of M^{pro} and the absence of homologous proteins in humans, it is a prime target for antiviral drug development.

When designing treatments for viral infections, the two main targets are the viral proteins essential to the viral life-cycle or host proteins involved in the viral life-cycle (Li *et al.*, 2023). The SARS-CoV-2 viral proteins investigated as potential targets are the spike protein, nucleocapsid, M^{pro}, papain-like protease, and NSP12 to NSP16. While mutations in the spike protein have been extensively studied due to their direct impact on viral entry into host cells and implications for vaccine efficacy, mutations in other regions, such as the M^{pro} and NSP16 protein, also warrant attention due to their potential to alter the replication and immune evasion capabilities of the virus. Understanding how residue mutations affect these proteins can provide valuable insights into the adaptability of viruses and lead to the development of more robust antiviral treatments.

1.1.1 SARS-CoV-2

SARS-CoV-2 is a virus belonging to the *Coronaviridae* family, characterised by their positive, single-stranded RNA genomes (Li *et al.*, 2023). Through phylogenetic analysis, it has been determined that SARS-CoV-2 belongs to the betacoronavirus genus, as it is closely related to other betacoronaviruses such as SARS-CoV and MERS-CoV. The

SARS-CoV-2 genome is approximately 30 000 nucleotides long and encodes a diverse range of proteins. Open reading frame ORF1ab encodes a polyprotein that, when cleaved, produces sixteen non-structural proteins (NSP1-NSP16), which play various roles in viral replication and transcription. There are four structural proteins, which are the spike (S), nucleocapsid (N), membrane (M), and envelope (E) proteins. Structural proteins are essential for viral assembly and entry into host cells. Lastly, the genome encodes nine accessory proteins (ORF3a, ORF3b, ORF6, ORF7a, ORF7b, ORF8b, ORF9b and ORF14) (Yang and Rao, 2021) that assist in manipulating the host cell environment to favour viral replication and evade the immune response (Li *et al.*, 2023).

1.1.1.1 Structure and Function of the M^{pro} Protein

The M^{pro} protein of SARS-CoV-2 is a key enzyme responsible for cleaving large viral polyproteins into functional NSPs, which are essential for viral replication (Hu *et al.*, 2022). The polyproteins involved are pp1a and pp1ab, which are cleaved at 11 distinct sites, producing NSP4 through NSP16. Structurally, M^{pro} consists of three domains, which are Domain I, Domain II, and Domain III.

The cleavage process of SARS-CoV-2 main protease (M^{pro}) begins with the binding of the substrate at the catalytic site of the enzyme, which is located between Domain I and Domain II (Hu *et al.*, 2022). This catalytic site contains five subpockets (S1 to S5), each of which interacts with specific positions on the polyprotein substrate (P1, P1', P2, P3, P3', P4). The P1 position must contain a glutamine (G) residue, which is recognized by the residues in the S1 subpocket, forming stabilizing hydrogen bonds via the beta-oxygen atom. While P1, P1' and P2 primarily determine substrate specificity, P3, P3' and P4 contribute to substrate recognition and stability. This precise binding ensures the correct positioning of the substrate for cleavage.

Once the substrate is correctly positioned, the catalytic dyad of H41 and C145 carries out the proteolytic cleavage (Hu *et al.*, 2022). H41 deprotonates the thiol group of C145, activating the thiol and making it highly nucleophilic. The activated C145 thiol then

attacks the C-terminal carbon atom of the glutamine residue at P1, forming a temporary covalent intermediate. This intermediate destabilizes the peptide bond between the P1 and P1' residues, leading to the release of the N-terminal segment. The C-terminal segment, beginning at the P1' residue, remains covalently attached to C145. To finalize the cleavage, a water molecule, activated by H41, breaks the bond between the C-terminal segment and C145, releasing the C-terminal product and restoring M^{pro} to its original state. This cycle enables M^{pro} to cleave multiple polyprotein molecules, generating the functional NSPs required for SARS-CoV-2 replication.

1.1.1.2 5'-end Cap Formation and Methylation of SARS-CoV-2 RNA

SARS-CoV-2, like many other viruses and eukaryotes, protects its RNA at the 5'-end using a capping mechanism (Sk *et al.*, 2020). Capping is useful as it allows for efficient RNA splicing, intracellular transport (Sk *et al.*, 2020) and RNA stability (including the prevention of 5'-3' exonuclease activity that would result in RNA degradation) (Ferron *et al.*, 2012). This capping process is shown in Figure 1.1, reflecting the enzymatic modifications to the viral RNA. The first step consists of the hydrolysis of the γ-phosphate from the 5'-triphosphate end of the RNA strand by an RNA triphosphatase (TPase) (Krafcikova *et al.*, 2020), forming a 5'-diphosphate end. Next, a guanylyltransferase (GTase) catalyses the transfer of a guanosine monophosphate (GMP) molecule to the 5'-diphosphate end, forming the primary cap structure. Following this are two methylation steps. The first is N-7 methylation of the GTP nucleobase, catalysed by an N-7 methyltransferase to form a cap-0 structure (Chen *et al.*, 2011). In SARS-CoV-2, the N-7 methyltransferase is encoded by the viral genome and is named NSP14. NSP16 (also called the 2'-O-methyltransferase) is the protein which attaches a second methyl group to the second nucleotide by ribose C2'-O-methylation, forming a cap-1 structure. NSP16 is m7GpppA-specific, meaning that ribose 2'-O-methylation using this protein can only occur when the cap-0 structure has been formed on the RNA molecule of SARS-CoV-2 (Rosas-Lemus *et al.*, 2020). The addition of the secondary methyl group mimics the capping process of eukaryotes, which helps the viral mRNA to remain undetected by the host immune system (Sk *et al.*, 2020).

This indicates that NSP16 plays a role in the SARS-CoV-2 escape from the host immune response (Chen *et al.*, 2011).

For ribose 2'-O-methylation to occur, a methyl group must be transferred from one methyl donor molecule to the nucleotide of the RNA molecule (Chen *et al.*, 2011). NSP16 is a class I methyltransferase which is S-adenosyl methionine (SAM) dependent, meaning that the SAM cofactor is a required methyl donor during the methylation process (Krafcikova *et al.*, 2020). Additionally, the SARS-CoV-2 Apo-NSP16 protein is not stable enough to bind a SAM molecule. SAM binding becomes possible only when NSP16 forms a complex with NSP10, which is characteristic of coronaviruses (Chen *et al.*, 2011). The overall 3D structure of the NSP10-NSP16 complex, along with the SAM cofactor, is shown in Figure 1.2.

NSP10 is a stable protein which stimulates both NSP14 N-7 methyltransferase activity and NSP16 2'-O-methyltransferase activity through allosteric interactions with these proteins Rosas-Lemus *et al.* (2020). NSP10 has been found to self-dimerise and form a dodecamer when extended by NSP11. No specific enzymatic activity has been found for NSP10 alone. However, it has been classified as a zinc-binding protein, which binds non-specifically to RNA molecules.

1.1.1.3 Structure of NSP16, NSP10 and the NSP10-NSP16 Complex

The SARS-CoV-2 NSP16 protein is composed of 298 amino acid residues and features a 2'-O-methyltransferase catalytic core (Rosas-Lemus *et al.*, 2020). This catalytic core consists of a Rossmann-like fold with eleven α-helices, seven β-strands and loops. These β-strands are arranged in a 3-2-1-4-5-6-7 pattern to form a β-sheet, surrounded by the α-helices and loops. The SAM molecule binds to the SAM-binding pocket of NSP16, which is formed between residues 71-79, 100-108 and 130-148 according to Rosas-Lemus *et al.* (2020). Residues N43, Y47, G71, G81, D99, N101, C115, and D130 are identified as key residues that interact with the SAM molecule through hydrogen bonding by Lin *et al.* (2020). Rosas-Lemus *et al.* (2020) indicate that residues K46, D130, K170 and E203

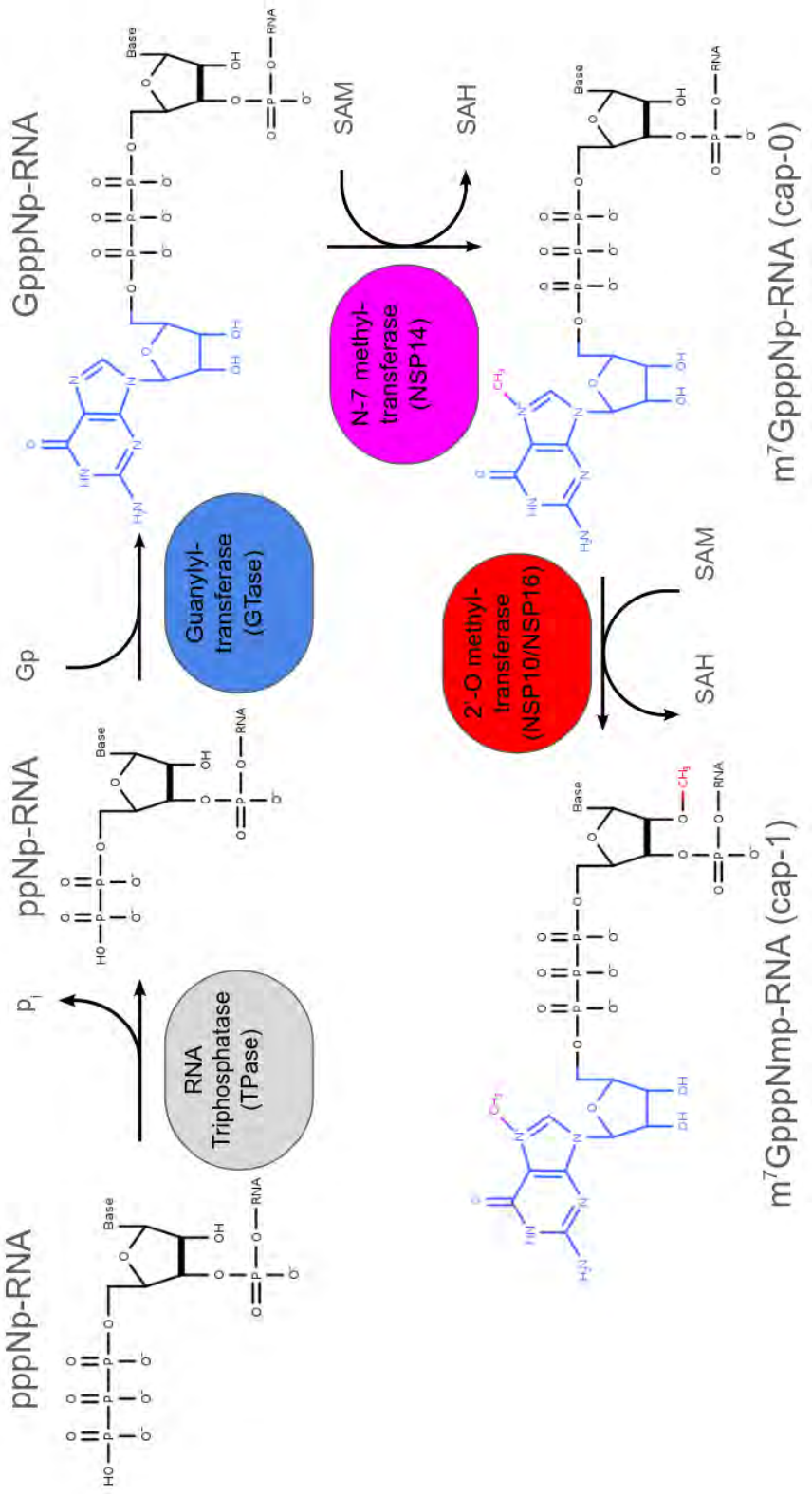

Figure 1.1: 5' cap formation and methylation processes for SARS-CoV-2

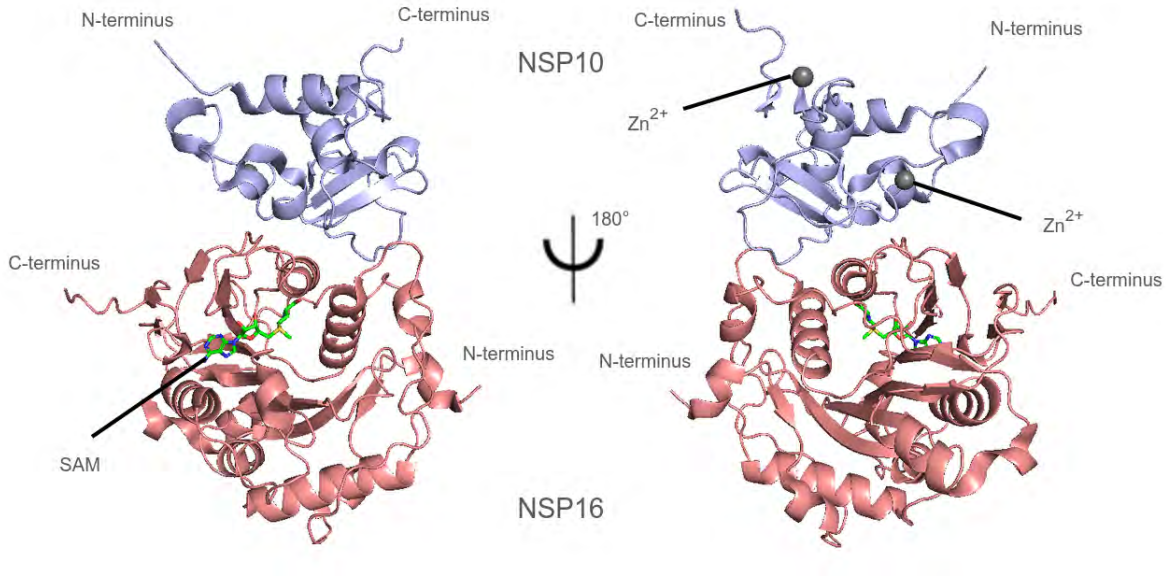

Figure 1.2: 3D cartoon representation of the NSP10-NSP16 complex with the SAM cofactor (PDB ID: 6WH4)

are found close to the SAM molecule methyl group and suggest that these residues may play an essential role in the methylation of the SARS-CoV-2 RNA molecule. These four residues form the highly conserved KDKE motif characteristic of 2'-O-methyltransferases (Rosas-Lemus *et al.*, 2020). This motif is believed to be crucial in binding the first adenine nucleotide, which receives the methyl group transferred from SAM (Chen *et al.*, 2011).

The SARS-CoV-2 NSP10 protein is 139 amino acids long and is divided into two subdomains (Rosas-Lemus *et al.*, 2020). The first is an α-subdomain, which consists of multiple α-helices and the two zinc fingers of NSP10 (Krafcikova *et al.*, 2020). The second domain is the β-subdomain, which includes a pair of anti-parallel β-strands (β1 and β2) and loops. The zinc fingers are important for the structural stability of NSP10 and are involved in mediating interactions with other viral proteins or RNA. The first zinc cofactor is coordinated with residues C74, C77, H83 and C90 and is thought to stabilise helices α2 and α3. The second zinc cofactor is found in a zinc-binding site produced by residues C117, C120, C128, and C130, which stabilises the C-terminus of NSP10.

NSP10 has a hydrophobic surface with positive charges that interact with the negatively charged hydrophobic pockets of NSP16, allowing the NSP10-NSP16 complex to form (Rosas-Lemus *et al.*, 2020). For example, NSP10 V42 is anchored in an NSP16 hydropho-

bic pocket formed by residues M41, V44, A73, V78, and P80 (Krafcikova *et al.*, 2020). Another NSP10 residue, L45, is anchored in another deep hydrophobic pocket created by the NSP16 residues P37, I40, V44, T48, L244, and M247. The anchoring of the NSP10 residues within the NSP16 hydrophobic binding pockets is thought to be essential in forming a stable NSP10-NSP16 complex. This is due to the hydrogen bonds that form directly between the residue atoms or are facilitated by water molecules. For example, a hydrogen bond forms directly between the NSP10 residue L45 and NSP16 residue G87, while a water bridge between NSP10 L45, NSP16 T48 and a water molecule was observed.

1.1.1.4 Residue Mutations of SARS-CoV-2 NSP16

It has been observed that the 2'-O-methyltransferase activity of NSP16 is essential to the viral immune evasion of coronaviruses (Schindewolf *et al.*, 2023). The KDKE motif of NSP16 forms part of the active site for 2'-O-methyltransferase activity. This motif is highly conserved across both viral and mammalian methyltransferase genomes, and no naturally occurring mutations have been found for these residues in any SARS-CoV-2 genomes (Deng *et al.*, 2024). However, studies have been conducted in which a single mutation is experimentally introduced to a residue of this motif to determine the extent to which the 2'-O-methylation functionality of the protein is affected. In the study by Schindewolf *et al.* (2023), the D130 residue was mutated to A130, as this mutation has previously been shown to reduce MTase activity in SARS-CoV (Bouvet *et al.*, 2010). The replication of the mutant virus was significantly reduced, and the virus showed attenuation in human respiratory cells (Schindewolf *et al.*, 2023). This mutation disrupted the formation of the cap-1 structure, hindering the virus's ability to mimic host RNA caps.

Interestingly, improved RNA binding affinity in NSP16 has been observed when comparing SARS-CoV-2 to SARS-CoV, even though the KDKE motif is conserved in both NSP16 proteins. It has been postulated by Deng *et al.* (2024) that this increase was caused by the mutations of residue 36, 138 and 153 of NSP16 between SARS-CoV and SARS-CoV-2. Residues 36 and 138 are found with gate loop 1 and 2 respectively which are located at the surface of the RNA binding pocket, whereas residue 153 is found more deeply within this

pocket. In their study, it was found that combining mutations L36I, N138H, and I153L resulted in the most significant decrease in the 2'-O-methylation activity of SARS-CoV-2 NSP16, as well as impaired replication and reduced pathogenicity.

Viswanathan *et al.* (2020) aligned the NSP16 sequences from nine coronavirus strains from the α, β and γ subclasses to identify any naturally acquired mutations in SARS-CoV-2. In particular, eleven notable mutations were found which occurred between SARS-CoV-1 and SARS-CoV-2, which are E32D, N33S, K135R, S188A, A209C, T223V, N265G, Y272L, E276S, V290I, and I294V. E32D, N33S and K135R are the mutations that occur on residues found in two loop regions of NSP16, which are thought to play a role in the binding of the RNA cap-0 molecule to NSP16. In addition, the S188A, A209C and E276S mutations are found in the region that forms a binding site for the first adenosine of the RNA molecule.

1.2 Related Studies

1.2.1 Related Studies - Mutation Frequency Predictions using Machine-Learning (ML) Techniques

As of June 2024, only one study has utilised ML models to predict mutation hotspots and coldspots in SARS-CoV-2 proteins. Mutation hotspots are the set of residues which are most likely to mutate within a viral protein, whereas the residues that will not mutate are in mutation coldspot regions (Barozi *et al.*, 2024). The study by Rawat *et al.* (2022) focused on using structural and sequence information to predict high and low mutability residue sites for all SARS-CoV-2 proteins. However, this study trained a classification model only, meaning that no attempts were made to predict actual mutation frequencies for the residues.

Other ML studies are more broadly related to mutation frequency predictions, such as predicting protein residue mutations not from SARS-CoV-2 or predicting genome mutations. For example, studies conducted by Yan and Wu predict the probability of amino

acid residues from specific Influenza A proteins having a high or low chance of mutation (Yan and Wu, 2020a,b). Notably, the methods used for this study were highlighted as applicable to the proteins translated from the SARS-CoV-2 genome by the authors (Yan and Wu, 2020a). The study by Saldivar-Espinoza *et al.* (2022) used artificial neural networks (ANNs) to predict mutations associated with the SARS-CoV-2 virus. However, rather than predicting mutation hotspots on SARS-CoV-2 proteins, this study focused on the nucleotide mutations in the SARS-CoV-2 genes that recurred across multiple lineages.

Multiple studies have also been conducted on either predicting mutations associated with proteins or the effects of mutations on the overall stability of those proteins. For example, a study conducted by Zhang *et al.* (2024) used pathogenicity and mutation data associated with human proteins to predict the pathogenicity of allosteric protein mutations and the sites at which these mutations occur.

These studies are detailed below, focusing on the type of prediction being performed in the study, the features and models chosen, and the overall performance of the trained models.

1.2.1.1 Prediction of Top and Bottom Mutability Sites in Proteins of SARS-CoV-2

Rawat *et al.* (2022) used Support Vector Machine (SVM) models to predict low and high mutability sites of all SARS-CoV-2 proteins. The non-synonymous single nucleotide polymorphism (SNP) entries stored in the 2019 Novel Coronavirus Resource until June 2021 were accessed to classify residues as high and low mutability sites. The number of isolates containing mutations occurring at a particular site was calculated using this data. The residue sites within the top 30% of the isolate count were labelled high mutability sites, while those within the bottom 30% were labelled as low mutability sites. The remaining sites were ambiguous (neither high nor low mutability sites).

The study tested 21 features describing either the sequence or protein structure. Some of these features include *relative accessible surface area*, *residue depth*, *surrounding hydropho-*

bicity, *residue type* (polar, non-polar, positive, etc.), *unfolding enthalpy of the chain*, and the *average position weight matrix*. However, to prevent overfitting, a maximum of 6 of the 21 features were considered for use in the SVM models. In addition, a forward feature selection approach was utilised, where the features that produced the highest area under the ROC curve (AUC) for the SVM models were chosen. The final four chosen features were the *residue at the mutation site*, *residues flanking the site*, *relative accessible surface area* and the *average position weight matrix*.

An SVM model was created for each selection threshold (5% to 30%) and evaluated using accuracy, sensitivity, specificity and the area under the ROC curve (AUC). The model with the best performance used a threshold of 5%, resulting in 76.7% accuracy, 76.5% sensitivity, 77.0% specificity and an AUC value of 0.84. As the threshold percentage increased, the performance of the model decreased. This was because higher percentage thresholds required less stringent criteria to classify sites as either low or high mutability, leading to a greater number of sites being incorrectly classified, thus reducing the model's accuracy.

1.2.1.2 Prediction of Mutation Positions in Influenza A Proteins

The occurrence or non-occurrence of mutations for each amino acid in various Influenza A proteins through evolutionary changes were predicted by Yan and Wu using ANNs (Yan and Wu, 2020a) (Yan and Wu, 2020b). For example, the protein sequences for a specific protein, such as H7 hemagglutinins, were retrieved from the Influenza virus resources database. Phylogenetic analysis was then performed to cluster the sequences and determine direct mutation relationships, referred to as father-daughter relationships, in this study. These relationships enabled the identification of amino acid differences between the sequences, highlighting the mutations between the father and daughter sequences.

The three features used for mutation prediction were *amino-acid pair predictability*, *amino-acid distribution predictability* and the *ratio of composition of amino acids*. The sequences and their features were split into training and testing data by date, where sequences submitted before 2000 were used for training of the ANN, while the remaining sequences were

part of the validation and testing datasets. The ANN model that was used was a 3-6-1 feedforward-backpropagation model, where the tan-sigmoid, tan-sigmoid and log-sigmoid transfer functions were used respectively for the three layers, and resilient backpropagation was used as the training algorithm. If the model returned a value over 0.5 for an amino acid in the protein, it indicated that a mutation was likely to occur at this position. This model was evaluated using specificity, sensitivity and total correct rate (defined by the authors as the number of true positives and true negatives divided by the number of amino acids in the sequence). The model had a very high specificity and total correct rate, both over 95%, but the sensitivity of the model was only around 20% for the test set. A logistic regression model was also used, but while it had similar specificity and total correct rates to the ANN model, the sensitivity was even lower than that of the ANN model.

1.2.1.3 Prediction of Recurrent Mutations in SARS-CoV-2

Saldivar-Espinoza *et al.* (2022) used ANN ML models to predict the positions where recurrent mutations occur and which mutations are recurrent in SARS-CoV-2. To identify all recurrent mutations, all high-coverage variant sequences submitted up until 19 April 2021 were retrieved from the Global Initiative on Sharing All Influenza Data (GISAID) database (Shu and McCauley, 2017). In particular, the mutations on the genome that did not result in insertions or deletions, along with the genome ID and date, were captured (Saldivar-Espinoza *et al.*, 2022). In addition, the nucleotide sequences were run through the Pangolin ML tool to classify the sequences into lineages based on Pango nomenclature. The individual lineages were grouped, and the number of distantly related lineages (NDRL) was calculated for each mutation. Each mutation was then classified as a recurrent mutation or not based on whether the NDRL exceeded the chosen thresholds of 1, 5, 10 and 15.

The features used to train these models were the *genome sequence*, the *predicted secondary structure of the genome*, *SHAPE-Seq reactivity values*, and the *amino acid sequences* translated from the coding sections of the genome sequence. However, these variables

were split into 13-nucleotide windows, with the centre nucleotide being at the position of interest for predicting the recurrent mutation. The testing set consisted of the M^{pro}, spike, PL^{pro} and RNA_pol genes, which were the genes of interest to the researchers. The validation set consisted of the helicase, NSP6, endoRNAse and M genes, as they were of a similar length to the genes in the testing set. The leader, NSP2, NSP4, NSP7, NSP8, NSP9, NSP10, exonuclease, methyltransferase, ORF3a, E, ORF6, ORF7a, ORF7b, ORF8, N and ORF10 genes were chosen for the training dataset.

The first ANN models were trained to predict whether a nucleotide at a position on the SARS-CoV-2 genome would be altered by a recurrent mutation. The predictions were evaluated using sensitivity, specificity, and area under the curve (AUC), where sensitivity was prioritised in choosing the best prediction model. As the NDRL threshold increased, the performance of the ANN model increased for all sets, with an AUC score of 0.81 for the testing set of NDRL 15. The model's specificity also increased from 0.46 to 0.69, while the sensitivity decreased slightly from 0.84 to 0.79 as the NDRL threshold increased. This indicates that the model could predict the position of recurrent mutations to some extent. Shapely Additive exPlanations (SHAP) were used to determine feature importance, where the nucleotide in the centre position was the most important. The nucleotides at other positions in the 13-nucleotide window and the SHAPE-Seq reactivity values were also considered important features.

1.2.1.4 Prediction of Pathogenic Mutations of Allosteric Proteins from SARS-CoV-2

Zhang *et al.* (2024) investigated the extent to which pathogenic mutations could be predicted for a set of allosteric proteins using a variety of ML models. All human protein-encoding genes were retrieved from the UniProt database (UniProt Consortium, 2018), which had missense mutations, clinical disease information recorded in ClinVar, and predictive results from the CADD, REVEL, DEOGEN2, gMVP, ESM1b, AlphaScore, and AlphaMissence ML methods. These sequences were assigned to the training set. The gene test set (TestSet_Gene) consisted of all genes from the Allosteric Database, which had

clinical information recorded in ClinVar and predictive results for all the ML methods. The mutations in these sets were then mapped to the allosteric-protein-encoded regions and allosteric sites to form the allosteric protein test set (TestSet_AlloProt) and allosteric site test set (TestAlloSite), respectively.

In addition to the predicted values from the ML methods chosen, the features used for predictions were the one-hot encoded protein mutations, B-factor and solvent accessibility. The ML algorithms compared during training included linear regression, SVM, random forest, AdaBoost and XGBoost. XGBoost had the highest predictive performance and was chosen as the model for pathogenic mutation predictions.

1.2.2 Related Studies - Residue Mutation Predictions for SARS-CoV-2 M^{pro}

As of June 2024, no attempts had been made to predict the mutation frequencies of SARS-CoV-2 proteins residues. Consequently, in December 2024, the article by Barozi *et al.* (2024) was published, in which we focused on predicting mutation frequencies of SARS-CoV-2 M^{pro} protein residues using ML models. By integrating molecular dynamics (MD) simulations and dynamic residue network (DRN) analysis as ML model features, we attempted to predict which residues were more prone to mutations and which remained stable over time.

Single nucleotide variants (SNVs) were identified from the GISAID database using SARS-CoV-2 sequences collected between 1 December 2019 and 24 February 2024. Only sequences with high coverage and detailed patient status metadata were included to ensure data reliability. Mutation frequencies per residue were predicted using regression models, which were then converted into binary classification labels. Residues with mutation frequencies above 20 were classified as mutating, while those below this threshold were considered non-mutating.

A total of 31 features were used in the study. Six independent MD simulations of the M^{pro} protein were conducted, from which DRN metrics and physicochemical properties

were derived. The selected features included eight centrality metrics from DRN analysis, three physicochemical properties—root mean squared fluctuation (RMSF), solvent-accessible surface area (SASA), and B-factor—and 20 BLOSUM62 matrix values to capture sequence-based mutation tendencies.

For predictions, ANN and RF models were implemented using MATLAB and Python. I developed the Python-based ANN model for this study. Model performance was assessed using R-values for regression and confusion matrices and AUC scores for classification. The regression models achieved test set R-values between 0.421 and 0.664, while the classification models attained AUC values ranging from 0.720 to 0.820. The results demonstrated moderate predictive power, though none of the models provided highly accurate mutation frequency predictions for the M^{pro} residues.

1.3 Problem Statement

Limited research has been conducted in accurately predicting the mutation frequencies of protein residues, especially in the context of SARS-CoV-2 proteins. Existing approaches, such as those outlined in Barozi *et al.* (2024), show only moderate performance in predicting residue mutations, which limits their broader applicability. To better understand viral evolution and design effective treatments, there is a need for more accurate models that can predict mutation hotspots and coldspots in viral proteins. In addition, it is essential that these models be generalisable across multiple SARS-CoV-2 proteins, enabling the prediction of mutations in different protein targets and enhancing model utility for broader antiviral drug development.

1.4 Aims

The three main aims of this study are listed below:

Develop and fine-tune ML models to accurately predict the likelihood of residue mutations in the SARS-CoV-2 M^{pro} protein.

Apply these optimised models to predict mutations in the SARS-CoV-2 NSP10-NSP16 complex as a case study, evaluating their generalisability to a different protein target.

Identify any models best suited for predicting SARS-CoV-2 protein mutations or determine if inherent limitations make accurate mutation prediction infeasible.

1.5 Hypotheses

The hypotheses for the three main aims are provided below:

- It should be possible to create a reasonably accurate ML model which can classify SARS-CoV-2 M^{pro} residues as highly mutable or not.
- Limited accuracy will be achieved for predicting mutating residues from the SARS-CoV-2 NSP10-NSP16 complex for the ML models trained only on M^{pro} data.
- Some models will have better accuracy than others due to the nature of the input dataset and the mechanics of the model.

1.6 Objectives

The objectives associated with these aims are as follows:

- To train and fine-tune supervised ML models with GISAID data, as well as MD and DRN analysis metrics to predict the mutability of SARS-CoV-2 M^{pro} residues

- To conduct MD simulations and DRN analysis on the wild-type NSP10-NSP16 complex, processing this data to extract features relevant to residue mutation predictions
- To apply the optimised models to predict mutation sites in the NSP10-NSP16 complex, evaluating the performance to assess model generalisability
- To conduct a comparative analysis of model performance, identifying potential predictive limitations and documenting factors that may challenge accurate mutation prediction

Chapter 2

Prediction of Mutation Sites in SARS-CoV-2 M^{pro} using Processed Trajectory Data

2.1 Introduction

Previously, both artificial neural network (ANN) and random forest (RF) models were implemented to predict both the mutation frequency of M^{pro} residues and whether a given M^{pro} residue would mutate or not (Barozi *et al.*, 2024). The highest accuracy achieved from these ANN and RF models was between 78% and 84% for the individual training, validation, and test dataset splits. This indicates that models with high performance have not yet been achieved. In addition, it was observed that the model performance varied significantly.

This chapter will focus on fine-tuning the previously implemented ANN and RF models and introducing SVM as an additional model to evaluate whether improved results can be obtained for SARS-CoV-2 M^{pro} residue prediction.

2.2 Literature Review

2.2.1 Machine Learning (ML) Techniques

ML models are algorithms that allow computers to learn from and make decisions or predictions based on data (Müller and Guido, 2016). Unlike traditional programming, where decisions are defined by the programmer, ML models identify patterns and relationships in data to derive their own rules and conclusions. These models can be broadly categorised based on their learning approach and the type of tasks they are designed to perform. The two primary learning approaches in ML are supervised and unsupervised learning, each with distinct characteristics and applications.

Supervised learning involves training a model on a labelled dataset, meaning the data includes both input features and the corresponding correct output (Müller and Guido, 2016). The model aims to learn a mapping from inputs to outputs that can be applied to new, unseen data. Supervised learning models can be further separated into regression models, which predict continuous output values, and classification models, which predict a specific value representing a particular class from a set of predefined classes. Unsupervised learning deals with unlabeled data, meaning the model tries to learn the patterns and the structure of the data without any specific output variable to guide the learning process. Some examples of unsupervised learning tasks are clustering and dimensionality reduction.

The following sections detail some of the supervised ML models, which are decision trees and their ensembles, support vector machines (SVMs) and ANNs.

2.2.1.1 Decision Trees

Decision trees are ML models that can be used for both classification and regression predictions (Müller and Guido, 2016). The main idea behind the model is partitioning the provided dataset into smaller and smaller groups based on differences in the provided features until all objects have the same class or value in each group. This results in a

tree-like structure with nodes, branches and leaves. Each node represents a test applied to one of the input features, while branches represent the possible outputs from this test. An output decision can then be linked to the subsequent node in the tree, which tests for a different condition based on the previous output. A leaf is the predicted output, either a specific label if it is a classification model or a continuous value for a regression model, found at the end of the tree.

To construct an optimal decision tree, the goal is to create nodes that separate the data into groups that are as pure as possible (Müller and Guido, 2016). This means each group contains instances of only one class (in classification) or minimises the variance (in regression). Construction of decision trees with the highest purity possible can be done by employing techniques that determine the most optimal way to split the data, with each technique using a different measure for determining the highest purity. These techniques are called impurity measures, where Gini impurity and entropy are the most commonly used.

Gini impurity is defined as "the frequency at which a randomly chosen element would be incorrectly labelled if it was randomly labelled according to the distribution of labels in the subset" (Breiman *et al.*, 1984). Entropy, however, is used to determine the homogeneity of the data. The lower the Gini impurity or entropy value returned for a particular test used in the node, the better the data split into separate classes, indicating a higher purity. The test which produces the highest increase in purity is chosen as the node when constructing the tree. Impurity measures are crucial in guiding splits by identifying the optimal feature and threshold for data partitioning. The goal is to minimise error by reducing impurity as much as possible with each split.

The main advantage of decision trees is that the algorithms do not require scaling of features (Müller and Guido, 2016). No scaling techniques such as normalisation or standardisation are required, and continuous and discrete data can easily be included without preprocessing. In addition, interpretation of single decision trees can be easier than other ML models, as they can be converted to flow charts that non-experts can understand. However, decision trees are prone to overfitting. This can occur even when using tech-

niques such as pre-pruning, which reduces model complexity. Overfitting can be combated by using decision tree ensemble methods. These methods include RF, AdaBoost and extreme gradient boosting (XGBoost), which are described below.

2.2.1.2 Decision Tree Ensembles

While decision trees are a single tree-like structure, decision tree ensembles combine multiple decision trees to form new ML models (Müller and Guido, 2016). The predictions of each tree within the ensemble are aggregated to produce an overall result. There are two main ensemble aggregation methods, which are boosting and bagging. RF employs the bagging method, whereas AdaBoost and XGBoost employ the boosting method.

In the RF bagging method, each tree is trained independently on a specific subset of the training data, with data points selected with replacement (Müller and Guido, 2016). After training, the predicted outputs of each tree for a data point are aggregated with equal weighting, either averaging the output for regression or returning the most frequent class for classification.

The boosting method trains all trees within the ensemble sequentially and gives uneven weighting to the outputs of each tree depending on the data point and how difficult it is to predict (Müller and Guido, 2016). When using the AdaBoost learning algorithm, data points that are predicted incorrectly by the tree have their weighting increased compared to those predicted correctly. This forces the subsequent tree to train on the data with the increased weight. In this way, individual weak tree learners are combined to create a more robust ensemble model, which is a strong learner. In contrast, the XGBoost algorithm tries to correct errors by minimising a loss function using gradient descent.

RF models averaging many decision tree models are less likely to overfit, compared to decision trees (Müller and Guido, 2016). An increase in trees reduces the variance of the model while keeping bias around the same, meaning there is less chance of overfitting. However, RF models do not perform as well when trained on high-dimensional, sparse

data compared to models using linear algorithms. In addition, they require more memory and training time than linear models.

2.2.1.3 SVMs

SVMs are another class of ML models, which are also applicable for both classification and regression tasks (Müller and Guido, 2016). These models are effective not only for linearly separable datasets but also for those with complex, non-linear relationships. This is because SVMs implement the kernel method, in which the input data is transformed to a new, higher-dimensional space, potentially allowing the data to become linearly separable (Cortes, 1995). This transformation is performed using one of the various kernel functions.

The chosen kernel function depends on the type of dataset provided to the SVM model (Müller and Guido, 2016). For example, the linear kernel is used for linearly separable data in the original feature space. In contrast, the Radial Basis Function (RBF) kernel is applied to handle non-linear datasets. The polynomial and sigmoid kernels are also popular choices. These kernel functions are also important in that they allow for the modelling of complex, non-linear relationships within the input data without explicitly computing the transformation to the higher-dimensional space, as the kernel trick is employed to reduce the computational burden.

The main goal of classification SVMs is to find the optimal decision boundary (also called a hyperplane) that best separates data points into distinct classes in the transformed high-dimensional space (Cortes, 1995). A key concept is maximising the margin, which is the distance between the hyperplane and the nearest data points from each class, known as support vectors (Kecman, 2005). By maximising this margin, SVMs improve the model's generalisation, ensuring strong performance on unseen data. In regression SVMs, the goal is slightly different, where the models aim to find a hyperplane (or a regression function) that best fits the data by minimising prediction error within a specified margin. In this case, the margin is a threshold within which errors are tolerated, and points outside the margin are penalised.

Both classification and regression SVMs are trained by minimising a loss function penalising misclassifications or prediction errors (Kecman, 2005). The loss function penalises points that fall within the margin for classification as they do not meet the required separation criteria. The regularisation parameter (C) is crucial in balancing the trade-off between maximising the margin and minimising errors (Müller and Guido, 2016). A higher value of C emphasises reducing errors, possibly at the expense of a smaller margin, while a lower C prioritises a wider margin, allowing some misclassifications or prediction errors.

Some advantages of SVMs are that they are effective predictors for both datasets with high-dimensional data and those smaller in size (Müller and Guido, 2016). In addition, they can handle non-linear data, which is not possible for some other ML models, and margin maximisation tuning helps to reduce the overfitting of the model. However, SVMs can be computationally expensive to run, particularly for larger datasets. The performance of SVMs is also dependent on the choice of kernel and hyperparameters, such as C and kernel-specific parameters. This means careful hyperparameter tuning is required for a successful model.

2.2.1.4 ANNs

ANNs are ML models that mimic the networks of neurons found in the human brain (Müller and Guido, 2016). They are composed of interconnected nodes or artificial neurons that process information, where each neuron receives inputs from other neurons, performs a computation on these inputs, and then sends the output to other neurons. This process allows the network to learn and model complex, non-linear relationships.

The basic structure of an ANN consists of an input layer, an output layer, and one or more hidden layers, where each layer contains at least one node (Müller and Guido, 2016). Each node in the input layer corresponds to a specific input data feature. The nodes in one layer are connected to every other node in the next layer. These connections between the nodes have a specific weight, where the more weighted connections have a greater

influence on the network output. The nodes in each layer perform a computation on the inputs they receive, which is usually a weighted sum of the inputs followed by an activation function. The output of each node is then passed to the next layer, and this process continues until the final output is calculated and returned from the output layer.

ANNs are trained by adjusting the weights associated with each connection between nodes (Müller and Guido, 2016). This is typically done using a loss function that measures the difference between the network's output and the desired output. The weights are adjusted to minimise this difference, which allows the network to learn and improve its performance over time.

The main advantage of using ANNs is their ability to form large, complex models to capture the patterns in large datasets (Müller and Guido, 2016). This can improve performance compared to other regression and classification models, such as RF and SVMs. However, achieving high overall performance with the ANN model requires carefully tuning its parameters and ensuring adequate training time. If the data is particularly large and complex, training time may be too long to be feasible. The data used to train the ANN model must also be appropriately preprocessed, and better performance is achieved when the features are similar in both measurement units and distribution. ANNs have also been found to be prone to overfitting.

2.2.2 Principal Component Analysis

Principal Component Analysis (PCA) is a dimensionality reduction technique that transforms high-dimensional data into a lower-dimensional space by identifying orthogonal components, called principal components, that capture the greatest variance in the dataset (Abdi and Williams, 2010). The first principal component accounts for the most variance, with subsequent components accounting for progressively less variance. This method is widely used to simplify data, reduce noise, and address multicollinearity, making it particularly valuable for high-dimensional datasets. PCA assumes linear relationships between variables and requires normalisation when variables are measured on different scales.

2.2.3 Evaluation of Regression and Classification Model Performance

It is important to determine and compare ML model performances to understand the differences between the models and identify the best model for predictions. There are numerous metrics for establishing a model's performance, and they are different for regression and classification tasks. The most commonly used performance metrics are described below.

2.2.3.1 R-value

The R-value, also called the correlation coefficient, is a measure of the strength and direction of a linear relationship between two variables (Schober *et al.*, 2018). The possible R-value for a given relationship is between -1 and 1, where a negative value indicates a negative correlation and a positive value represents a positive correlation. The strength of the relationship depends on the magnitude of the R-value, with low values labelled as weak, and R-values close to 1 as strong. When using the R-value in ML model evaluation, the two variables are the predicted output from the model and the actual target values. In this case, an R-value of 1 would suggest the model was a perfect predictor, a value of 0.5 would indicate a moderate predictor, while a value of 0 would suggest there is no linear relationship between the target and predicted values, meaning the model has no predictive power.

The formula for determining the R-value of variables x and y is the following:

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}$$

2.2.3.2 R^2-Value

The R^2-value, or the coefficient of determination, is a metric that calculates the proportion of the variance explained by one or more independent variables for the dependent variable.

When used to evaluate ML models, the R^2-value provides an indication of how well the predicted outputs correspond to the actual target values (Müller and Guido, 2016). Unlike the R-value, which measures the strength and direction of a linear relationship, R^2 is used to determine the goodness of fit of a model to the data.

The R^2-value ranges from 0 to 1, where a value of 1 indicates that the model perfectly explains all the variability in the target data, and a value of 0 suggests that the model fails to explain any of the variability beyond the mean of the target values (Müller and Guido, 2016).

The formula for calculating R^2 is given as (Asuero *et al.*, 2006):

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$

Here, y_i represents the actual target values, $\hat{y}_i$ represents the predicted values, and $\bar{y}$ is the mean of the actual target values. The numerator reflects the sum of squared residuals (errors), while the denominator represents the total sum of squares (variance) in the actual target values.

In the context of ML, a high R^2-value suggests that the model effectively captures the relationships between features and target outputs, while a low R^2-value indicates the model may require refinement (Müller and Guido, 2016). However, R^2 values alone do not provide insights into overfitting, underfitting, or error distribution.

2.2.3.3 Accuracy

Accuracy is defined as the proportion of test samples for which the actual label matches the predicted label (Müller and Guido, 2016). The formula for calculating accuracy is provided below:

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

where:

- TP (True Positive) - A positive case was correctly predicted as positive
- TN (True Negative) - A negative case was correctly predicted as negative
- FP (False Positive) - A negative case was incorrectly predicted as positive
- FN (False Negative) - A positive case was incorrectly predicted as negative

Accuracy is one of the most commonly used metrics in ML and is preferred for comparing classification models (Müller and Guido, 2016). However, it must be noted that accuracy is considered an optimal parameter when there is an even distribution of samples between the classes.

2.2.3.4 Precision, Recall, and F1 Score

Precision, recall, and the F1 score are evaluation metrics often used for classification models, particularly in scenarios where the dataset is imbalanced. These metrics provide a more nuanced view of a model's performance beyond accuracy.

Precision: Precision measures the proportion of true positive predictions among all the instances predicted as positive (Müller and Guido, 2016). It reflects the model's ability to avoid false positives and is calculated as

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

A high precision score indicates that the model makes few false positive predictions, meaning it is reliable in identifying positive cases. This metric is used for scenarios where the correctness of positive case predictions is important, meaning that false positives (such as testing positive for an illness when healthy) have a high cost.

Recall: Also known as sensitivity or true positive rate, recall measures the proportion of actual positive instances correctly identified by the model (Müller and Guido, 2016). It is calculated as

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

A high recall score indicates that the model can correctly predict the majority of actual positive cases and, therefore, minimise the number of false negatives (Müller and Guido, 2016). This is essential for prediction tasks where it is important to identify nearly all positive cases correctly. For example, in medical diagnostics, a high recall value is required for tasks such as identifying cancerous tissue samples from a set of biopsies. This is because failing to detect any cancerous samples could cause severe consequences for the patients.

F1 Score: The F1 score is the harmonic mean of precision and recall, providing a balanced metric that accounts for both false positives and false negatives (Müller and Guido, 2016). This metric is particularly useful when the classes are imbalanced. The formula for the F1 score is

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

The F1 score ranges between 0 and 1, where 1 indicates the model has perfect precision and recall (Müller and Guido, 2016). A higher F1 score indicates a better trade-off between precision and recall, making it a preferred metric for many classification tasks, especially when false positives and false negatives carry different consequences.

These metrics together allow for a comprehensive evaluation of a model's classification performance and help in comparing models tailored to specific needs or use cases.

2.3 Methodology

2.3.1 Dataset Choice

The M^{pro} dataset from Barozi *et al.* (2024) is used in this study for training and testing the ANN, RF and SVM models for predicting SARS-CoV-2 M^{pro} mutations. This dataset consists of 304 data points, each representing one of the first 304 residues from chain A of the M^{pro} homodimer (PDB ID: 5RFV). Each data point is described by 31 features, which are 8 DRN metric values (*averaged BC*, *CC*, *DC*, *EC*, *ECC*, *L*, *PR* and *Katz*), the 20 BLOSUM62 matrix values and three physicochemical properties for each residue (RMSF, B-factor and SASA).

The DRN centrality metrics are used to explore the impact of residues on the protein's structural dynamics, based on variations in distances observed during molecular dynamics simulations. The RMSF, B-factor, and SASA provide structural information about each residue, describing its flexibility, atomic mobility, and solvent exposure, respectively. The BLOSUM62 values capture the general evolutionary conservation of amino acid substitutions, enabling the assessment of how conserved each residue is across different protein sequences. Although these BLOSUM values do not directly account for the specific structural or functional role of a residue within a particular protein, they were included due to their potential usefulness. Specifically, they were expected to complement the dynamic and structural features (DRN metrics, RMSF, SASA, and B-factor), which are more sensitive to the local structural and dynamic environment of the residues.

The target variable in this dataset for regression tasks is the mutation frequency of each residue for all VoCs in the GISAID database. For classification tasks, the mutation frequency is converted to a binary classification of 0 for residues that do not mutate and 1 for residues considered to mutate in VoCs, with a cutoff value of 20 mutations. The class distribution for this dataset is 70% of samples are non-mutating residues, and 30% are mutating.

2.3.2 ML Phase - Training of Models using M^{pro} Dataset

Four experiments were conducted to improve the performance metric results and reduce the variation across different splits for the ANN and RF models compared to the models from Barozi *et al.* (2024). Each experiment was designed based on the results of the previous one. The initial experiment focused on hyperparameter tuning, which led to further preprocessing of the input data and, finally, to identify the cause of the variance observed between models. The SVM model was also included in all experiments to test the effect of a different ML approach on the overall performance of mutation prediction or reduce the variance between splits.

In Barozi *et al.* (2024), the original ANN and RF models were used for regression and classification tasks. This was done by constructing regression models from which predicted mutation frequencies for the M^{pro} residues were output. These predicted frequencies were then converted to a binary output, where class 1 represented mutated residues (frequency over 20) and class 0 represented the remaining non-mutated residues. In this study, separate regression and classification models were produced for the ANN, RF and SVM models to investigate whether separate hyperparameter tuning would improve performance for the different and separate models. Although both classification and regression models were investigated, the classification models were the focus of the investigation.

2.3.2.1 Feature Scaling

The target values for the regression models can be large (e.g. mutation frequencies ranging from 0 to 67139 for M^{pro} residues). A log transformation of these values allows for the compression of the range and reduces the skewness of the data, allowing for the potential improvement in model performance. Therefore, the log(mutation frequency + 1) for each residue was calculated and used as the actual values for the regression models.

The StandardScaler scaling technique from Scikit-learn was utilised for preprocessing the M^{pro} dataset used in Barozi *et al.* (2024). However, not all features in the M^{pro} dataset are normally distributed, meaning this technique should not have been applied. Therefore, a

new normalisation technique, MinMaxScaler, was utilised instead of StandardScaler. This technique rescales the data of each feature to a fixed range, preserving the distribution and shape of each feature. Improvements to the model performances were observed in Barozi *et al.* (2024) when applying feature scaling. Therefore, an alternative scaling technique was chosen rather than removing feature scaling altogether.

2.3.2.2 Class Imbalance

The M^{pro} dataset contains uneven classes, with 212 residues labelled as not mutating and only 92 mutating. This produces a class imbalance within the dataset, which must be addressed. Stratified sampling was implemented during the splitting of samples to preserve the original class distribution, maintaining approximately 70% non-mutating and 30% mutating residues in each subset. This means that the splitting of the residues no longer remains completely random. However, without stratified sampling, the training and evaluation of the model would no longer be accurate.

2.3.2.3 Performance Metrics for the Chosen ML Models

As both regression and classification tasks must be predicted (mutation frequency and whether a given residue will mutate or not), one or more metrics for each task must be chosen. In this case, the R-value and accuracy metrics were selected for the regression and classification of mutations, respectively.

The M^{pro} dataset has an uneven class distribution, where a higher proportion of residues are considered not to mutate. As a result, accuracy is not necessarily the most optimal metric for analysing the performance of the classification models. However, it is necessary to determine whether the fine-tuned models in this study are improved compared to the performance of the models in Barozi *et al.* (2024). Despite this class imbalance, accuracy was chosen as the metric for comparing the models in this study. This decision allowed for a simpler comparison between the performance of these models and the ANN and RF

models in Barozi *et al.* (2024), without the additional complexity of producing confusion matrices for each run.

2.4 Experiment 1: Hyperparameter Tuning

The first experiment aimed to improve the performance and reduce the variability of ML models by fine-tuning their hyperparameters. Building on the models from Barozi *et al.* (2024), where limited tuning was performed, this experiment introduces further fine-tuning of the ANN and RF models for regression and classification tasks. Additionally, SVM classification and regression models were included to evaluate the performance of alternate ML models.

2.4.1 Methodology

Batch normalisation was included between some of the hidden layers of the regression model. This technique is less aggressive than dropout and can be applied to mitigate overfitting (Ioffe, 2015). In addition, a dropout layer was included between the classification ANN model hidden layers to reduce overfitting and allow the model to be trained for longer (Srivastava *et al.*, 2014). This meant the number of epochs could be increased from 60 in Barozi *et al.* (2024) to 1000. In addition, the batch size for both models was increased. As the computing power of the device increases, the batch size should similarly increase. The original batch size of 2 was insufficient for the computing power of the device, so the batch size was increased to 16. A set of optimal parameters was chosen for the classification and regression ANNs, and they were applied to all random data splits. These configurations were selected as they achieved the highest model performance across the training, validation, and test sets while minimizing overfitting.

The final ANN regression model architecture consisted of 2 hidden layers containing 16 and 8 nodes, respectively. Batch normalisation was included between the last hidden layer and the output layer. The sigmoid function was chosen as the activation function

for all hidden layers, while the linear activation function was applied in the output layer (consisting of a single node). This model was compiled using the Adam optimiser, with mean squared error as the loss function and evaluation metric for the model, as performed in Barozi *et al.* (2024). The ANN model was then trained on the scaled M^{pro} dataset, split into 75%-15%-15% train-validation-test subsets. The model was fitted for 400 epochs with a batch size of 16. Subsequently, the predicted values were refined through fitting the predicted values again, this time over 200 epochs with a batch size of 8.

The ANN classification model architecture consisted of three hidden layers with 64, 32, and 16 nodes, respectively. Each hidden layer utilised the sigmoid activation function, while the output layer used a sigmoid activation function for binary classification. Dropout layers were applied after each hidden layer with a 10% drop rate to help prevent overfitting. The model was compiled using the Adam optimiser, with binary cross-entropy as the loss function and accuracy as the evaluation metric. The ANN model was trained on the scaled M^{pro} dataset, split into 75%-15%-15% train-validation-test subsets. The model was fitted on the dataset for 1000 epochs with a batch size of 16.

A set of commonly used hyperparameters were chosen for tuning the RF model. This was performed with GridSearchCV and a `cv` of 5. The best model was selected using the lowest negative mean squared error for regression and the highest accuracy for classification. Instead of choosing one set of optimal parameters for all random data splits, the best model was re-defined for each random split tested. Table 2.1 lists all the parameters of the model that were tuned and the possible values for tuning. These parameters were the same for the regression (RandomForestRegressor) and classification (RandomForestClassifier) models.

Table 2.1: Parameter grid used for RF in GridSearchCV.

Hyperparameter	**Values Tested**
`n_estimators`	50, 100, 200
`max_depth`	None, 10, 20, 30
`min_samples_split`	2, 5, 10
`min_samples_leaf`	1, 2, 4
`max_features`	`sqrt`, `log2`

Similarly, the best hyperparameters for the SVM classification (SVC) and regression (SVR) models were identified for each new train-validation-test split. This was performed using GridSearchCV with a `cv` of 10. The best model was chosen based on the lowest negative mean squared error for regression and the highest accuracy for classification. Table 2.2 shows the potential values for each tuned hyperparameter.

Table 2.2: Parameter grid used for SVM in GridSearchCV.

Hyperparameter	Values Tested
`Kernel`	linear, poly, rbf, sigmoid
`C`	0.1, 0.5, 1, 5, 10
`Degree`	1, 3, 5
`Coef0`	0.001, 0.01, 0.1, 10, 0.5
`Gamma`	auto, scale

2.4.2 Results and Discussion

2.4.2.1 Classification for M^{pro} Dataset

The average accuracies for all three ML models, shown in Table 2.3, are comparable to the 67% to 78% accuracy range for the validation and testing subsets reported by Barozi *et al.* (2024). To assess performance consistency, the standard deviation of the models' accuracy across five runs was calculated. The models' overall standard deviation (and therefore variance) across random data splits appeared to be lower compared to the initial average results reported in Barozi *et al.* (2024). However, the standard deviation of the accuracy between these five runs is often over 5%. This indicates a high variance between average accuracies for a model with different splits and different hyperparameters for the RF and SVM models. Finally, no models exhibited high performance, as the average accuracies for the validation and test splits were never above 77%.

2.4.2.2 Regression for M^{pro} Dataset

Previously, we had been considering the performance of classification models, predicting whether a given M^{pro} residue would mutate. Changes to the original regression model

Table 2.3: Average accuracies of the ML models after hyperparameter tuning.

Model	Training Set Accuracy (Mean ± SD)	Validation Set Accuracy (Mean ± SD)	Testing Set Accuracy (Mean ± SD)
ANN	81.8868 ± 1.7548	74.3478 ± 3.4783	75.2174 ± 2.9488
	82.8302 ± 1.5673	71.7391 ± 6.8745	72.1739 ± 5.7352
	83.8679 ± 1.5616	71.3043 ± 3.7401	77.3913 ± 3.5322
RF	77.9203 ± 1.4004	76.9565 ± 6.5362	75.6522 ± 3.4783
	76.7929 ± 2.0036	76.5217 ± 8.4083	76.9565 ± 3.2536
	76.7796 ± 1.1839	74.7826 ± 4.8804	73.9130 ± 5.3250
SVM	82.8302 ± 7.7943	66.5217 ± 4.2600	73.0435 ± 7.0911
	83.4906 ± 4.9292	72.6087 ± 3.5322	63.4783 ± 5.0330
	77.6415 ± 2.9016	72.1739 ± 7.7044	67.8261 ± 9.6635

hyperparameters were introduced, as provided in Section 2.4.1. These new regression models were used to determine the extent to which these M^{pro} mutation frequencies could be predicted using the given M^{pro} dataset as input. The performance of these models is measured by the R-values shown in Table 2.4.

The RF regression models had the highest R-values for the training set but performed comparably to the ANN and SVM models for the validation and test sets. The ANN model showed the lowest R-values and highest variability of all the regression models, indicating potential underfitting or instability. However, the SVM model outperformed the ANN and RF models in the testing subset, with average R-values of around 0.5. This suggests that these models generalised better to the M^{pro} dataset in this scenario. In addition, the SVM models had the lowest training average R-values but were most similar to the validation and testing set average R-values. This indicates that these models may not be overfitting, unlike the RF models. However, the majority of the standard deviation values for all models appear to be over 0.1 for the validation set and 0.05 for the test set, indicating a large variance between runs of the regression models.

The R-values for the original RF regression models in Barozi *et al.* (2024) were recorded as 0.489 for the validation subset and 0.581 for the test subset. These values are slightly higher when compared to the average R-values produced by the RF models in this experiment, which range from 0.4643 and 0.5305 for the validation set and 0.4893 and 0.5612 for the test subset. However, there was only a single example Python regression score in

Table 2.4: Average R-values of the ML models after hyperparameter tuning.

Model	Training Set R-value (Mean ± SD)	Validation Set R-value (Mean ± SD)	Testing Set R-value (Mean ± SD)
ANN	0.7249 ± 0.0198	0.3867 ± 0.0932	0.3975 ± 0.0843
	0.7191 ± 0.0169	0.4088 ± 0.1317	0.4681 ± 0.0255
	0.7037 ± 0.0259	0.4589 ± 0.0600	0.4877 ± 0.0831
RF	0.9176 ± 0.0304	0.5305 ± 0.1003	0.4893 ± 0.1214
	0.9038 ± 0.0498	0.4643 ± 0.1148	0.5612 ± 0.0990
	0.8680 ± 0.0494	0.5039 ± 0.1022	0.4984 ± 0.0827
SVM	0.6800 ± 0.0238	0.4910 ± 0.1164	0.5280 ± 0.0644
	0.6326 ± 0.0688	0.5256 ± 0.1093	0.4898 ± 0.0613
	0.5874 ± 0.0592	0.4994 ± 0.0762	0.5290 ± 0.1018

Barozi *et al.* (2024) to compare against. This means it is harder to determine whether the slight reduction in model performance seen by the decrease in R-value is a true representation.

However, the average training R-values were much higher for the RF models in this study, at around 0.9, compared to 0.6 for the original regression models in Barozi *et al.* (2024). This indicates that the RF models in this study were overfitting using this set of hyperparameters, compared to the hyperparameters chosen in Barozi *et al.* (2024). This is likely due to the low minimum number of samples (1, 2 or 4) per leaf as the only possible option for the RF models in this study.

The original ANN models' R-values varied, ranging from 0.382 to 0.602 for the validation subset and 0.480 to 0.696 for the testing subset. Comparatively, the average R-values for the tuned ANN models in this experiment were much lower, ranging between 0.3867 and 0.4589 for the validation subset and between 0.3975 and 0.4877 for the testing subset. However, the variance between these average R-values is much lower, particularly for the testing subset. This suggests that while the overall R-values decreased, the overall performance of the new model was more consistent.

2.4.3 Conclusion

It appears that the overall performance of all the ML models did not increase after fine-tuning the hyperparameters. This suggests that hyperparameter optimisation could not address the issues affecting the performance of these models.

2.5 Experiment 2: PCA

Instead of continuing to fine-tune the models, a decision was made to focus on preprocessing the input data of the models. Therefore, PCA was applied to the input features of the M^{pro} dataset to determine whether preprocessing of the input data increases the overall performance of the models.

2.5.1 Methodology

PCA was applied to the input features of the M^{pro} dataset using the Scikit-learn PCA function, resulting in numerous principal components, which are linear combinations of the original input features. The top 3, 5 and 7 components which explained the most variance within the data were then identified. They were used as the input data for the ML models described in Experiment 1 instead of the full 31 features of the M^{pro} dataset. Besides the number of features as inputs for these models, no other changes were made to either the model architectures or the methodology for identifying the best models and their subsequent training.

2.5.2 Results and Discussion

Tables 2.5, 2.6 and 2.7 contain the average accuracies of the ML models which were trained, validated and tested with the 3, 5, and 7 components explaining the most variance within the data, respectively.

Table 2.5: Average accuracies of the ML models with features from PCA with 3 components.

Model	Training Set Accuracy (Mean ± SD)	Validation Set Accuracy (Mean ± SD)	Testing Set Accuracy (Mean ± SD)
ANN	76.6981 ± 1.4182	76.9565 ± 3.5322	77.8261 ± 4.2154
	79.3396 ± 0.6933	75.2174 ± 3.2536	73.9130 ± 6.5938
	76.9811 ± 2.9139	74.7826 ± 8.6521	76.0870 ± 1.9444
RF	76.4939 ± 1.1788	76.9565 ± 2.9488	76.5217 ± 4.4339
	76.5094 ± 1.4970	77.8261 ± 3.4783	71.7391 ± 1.9444
	74.4385 ± 1.4066	77.3913 ± 7.4803	79.1304 ± 6.8193
SVM	74.3396 ± 2.2444	71.7391 ± 10.4710	64.7826 ± 3.4783
	70.6604 ± 1.2797	70.4348 ± 4.2600	67.3913 ± 2.7498
	73.4906 ± 3.5096	71.3043 ± 5.8977	70.0000 ± 4.4339

Table 2.6: Average accuracies of the ML models with features from PCA with 5 components.

Model	Training Set Accuracy (Mean ± SD)	Validation Set Accuracy (Mean ± SD)	Testing Set Accuracy (Mean ± SD)
ANN	79.3396 ± 1.7497	77.3913 ± 3.7903	74.3478 ± 5.0330
	79.2453 ± 2.4419	77.8261 ± 4.4339	78.6957 ± 7.1969
	79.6226 ± 1.3474	74.7826 ± 1.0650	76.5217 ± 8.1804
RF	77.2558 ± 1.4827	72.6087 ± 3.5322	73.4783 ± 2.1300
	76.3189 ± 1.4967	76.5217 ± 5.2174	74.3478 ± 7.1969
	74.6534 ± 2.6643	75.6522 ± 7.4550	76.0870 ± 4.3478
SVM	73.7736 ± 4.2463	71.7391 ± 6.8745	69.1304 ± 7.5807
	79.5283 ± 7.0002	70.4348 ± 10.3438	67.3913 ± 4.5600
	75.3774 ± 2.6916	63.9130 ± 5.2535	72.1739 ± 8.9527

Table 2.7: Average accuracies of the ML models with features from PCA with 7 components.

Model	Training Set Accuracy (Mean ± SD)	Validation Set Accuracy (Mean ± SD)	Testing Set Accuracy (Mean ± SD)
ANN	79.7170 ± 1.4307	78.6957 ± 1.6268	74.3478 ± 4.2154
	81.6038 ± 1.7900	78.2609 ± 4.9573	71.7391 ± 4.5600
	79.9057 ± 1.0590	75.2174 ± 3.2536	76.9565 ± 2.9488
RF	76.1417 ± 2.8816	70.8696 ± 6.9565	78.2609 ± 3.6377
	78.0221 ± 1.3935	71.3043 ± 5.3955	76.5217 ± 7.1969
	77.4707 ± 1.6881	74.3478 ± 3.9848	74.3478 ± 4.8415
SVM	77.7358 ± 4.6351	73.4783 ± 2.1300	70.4348 ± 4.4764
	75.0000 ± 2.0883	73.9130 ± 4.1247	63.9130 ± 4.8804
	76.9811 ± 5.2813	66.0870 ± 4.4764	73.4783 ± 9.0577

The performance of the SVM models decreased compared to the ANN and RF models when PCA was applied. This may suggest that the SVM may need different hyperparameter tuning options when using fewer features during training. The average accuracies for the ANN and RF models were similar to the baseline average accuracies seen in Experiment 1, with some slight fluctuations, but not noticeable enough to see a trend between the change in the number of components and the average accuracy. There also appears to be no relationship between the standard deviation of the accuracy of the models and the number of components from PCA chosen as input features for any of the ML models. It was observed that the average accuracies and standard deviations were not significantly improved compared to the baseline models when using the original features.

2.5.3 Conclusion

When tested on the 3, 5 and 7 PCA components, the average accuracy and standard deviations of the ML models did not improve compared to the results of the models in Experiment 1, tested on the original features. This suggests that preprocessing of the input dataset was also insufficient for improving either the average accuracy or reducing the variance in performance for these models. Therefore, the decision was made to revert to using the original features.

2.6 Experiment 3: Fixing of Training-Validation-Test Splits

Instead of further tuning the model parameters or input features in hopes of improving performance, identifying the root cause of the variance in model performance was prioritised. This is because targeted improvements for enhancing model performance consistently are easiest to implement when the cause of the variance in performance is identified. The variance between the models is most likely caused either by improper hyperparameter tuning (where the most optimal parameters for the models have not been

provided as options) or the dataset splits (where the presence/absence of specific residues within subsets cause the model to perform better or worse). The aim of this experiment is to remove the effects of splitting so that the given variance is only attributable to hyperparameter tuning.

2.6.1 Methodology

The initial models and dataset from Experiment 1 were used for this experiment. This included the continuation of hyperparameter tuning per random split chosen for the SVM and RF models. However, random data splits were not implemented for each model run. Instead, one fixed training-validation-test dataset split was selected for all five runs of all three ML models before being reset. This was repeated three times.

2.6.2 Results and Discussion

The results for the ANN, SVM and RF models when introducing the fixed training-validation-test split are shown in Table 2.8. By fixing the split, the same residues will always be found in the same subsets, leaving only the initial weights and the training of the model as the determining factor for the variance in performance. A low standard deviation indicates that the splits caused the variance in the previous experiments, while a high standard deviation (above 5%) indicates that hyperparameter tuning produced this variance.

The RF and SVM models are not stochastic, meaning that given the same data and hyperparameters, the model will be trained so that the same final model weights and predictions are achieved. Due to their non-stochastic nature, running them with the same training-validation-test split produces the same accuracy for all five runs, resulting in a standard deviation of 0 (as seen in Table 2.8). However, for the ANN model, the initial random weights would result in slight changes in accuracy for each of the five runs. However, the standard deviation of the accuracy between these runs was below 5%. In addition, the

Table 2.8: Average accuracies of the ML models with fixed 70-15-15 splits

Model	Split	Training Set Accuracy (Mean ± SD)	Validation Set Accuracy (Mean ± SD)	Testing Set Accuracy (Mean ± SD)
ANN	Split 1	77.9245 ± 0.6258	82.1739 ± 1.6268	72.1739 ± 2.8840
		78.5849 ± 0.7060	81.3043 ± 2.9488	72.1739 ± 0.8696
		78.6792 ± 0.5501	80.8696 ± 2.1300	73.0435 ± 1.0650
	Split 2	84.5283 ± 0.6933	73.4783 ± 0.8696	65.2174 ± 1.3749
		84.2453 ± 0.4810	72.1739 ± 0.8696	65.2174 ± 2.3814
		84.2453 ± 0.4810	73.0435 ± 1.0650	67.3913 ± 1.3749
	Split 3	81.8868 ± 0.7060	80.8696 ± 0.8696	73.9130 ± 0.0000
		82.3585 ± 0.6398	81.7391 ± 1.0650	73.4783 ± 0.8696
		81.6981 ± 0.5501	81.7391 ± 1.0650	73.9130 ± 0.0000
RF	Split 1	74.5404 ± 0.0000	86.9565 ± 0.0000	76.0870 ± 0.0000
	Split 2	75.4928 ± 0.0000	80.4348 ± 0.0000	76.0870 ± 0.0000
	Split 3	75.4817 ± 0.0000	80.2609 ± 0.0000	71.7391 ± 0.0000
SVM	Split 1	80.6604 ± 0.0000	80.4348 ± 0.0000	69.5652 ± 0.0000
	Split 2	100.000 ± 0.0000	60.8696 ± 0.0000	58.6957 ± 0.0000
	Split 3	75.9434 ± 0.0000	69.5652 ± 0.0000	63.0435 ± 0.0000

average accuracies between the different splits varied significantly. This indicates that the dataset split is the main source of the variance and not the hyperparameter tuning of the models. As a result, the current standard deviation cannot be improved with the current data containing only 304 residues as samples, and potentially, the data would need to be increased to reduce the variance.

2.6.3 Conclusion

Due to the observed reduction in variance after introducing the fixed training-validation-test splits, the variance in ML model performance is attributed to the dataset splits and not hyperparameter tuning. Therefore, further improvements to the models are likely to be obtained by increasing the number of inputs provided rather than continuing to tune the hyperparameters of the models at this stage.

2.7 Experiment 4: Train-Test Only Splits

While Experiment 3 demonstrated that dataset splitting contributed to model variance, allocating samples across three subsets (training, validation, and test) may have exacerbated the observed performance issues due to the limited dataset size. To address this, Experiment 4 investigates whether performance can be improved by simplifying the splitting strategy only to include training and testing subsets. This reduction could allow more data points in each subset, potentially enhancing model training and evaluation accuracy.

2.7.1 Methodology

The validation set was removed for this experiment, leaving only the training and test sets. Train-test splits of 85%-15% and 70%-30% were implemented on the initial dataset, using the same design and tuning of the models in Experiment 1. Similarly, the average performance for each model was calculated over five runs, repeated 3 times.

2.7.2 Results and Discussion

Hyperparameter tuning was performed using cross-validation for both the SVM and RF models, while no additional tuning was performed for the ANN models. This means the subsets of the training set provided were used to fit and determine the optimal model parameters, which meant the validation set was unnecessary. In addition, the number of residues in either the training or testing subset would increase due to the absence of a third subset. Any potential effects of increasing the number of data points used to either train or test the model can now be observed.

The results of using an 85%-15% split for the ML models are shown in Table 2.9, where an increase in the number of training values was introduced. The training accuracy was seen to have increased slightly from the values in Experiment 1, particularly for the RF model. However, there was no reduction in standard deviation values for any of the

Table 2.9: Average accuracies of the ML models with an 85%-15% train-test split

Model	Training Set Accuracy (Mean ± SD)	Testing Set Accuracy (Mean ± SD)
ANN	83.0233 ± 1.2109	73.4783 ± 0.8696
	83.6434 ± 1.9519	73.4783 ± 7.0644
	81.5504 ± 1.1654	76.5217 ± 4.2154
RF	78.2262 ± 0.8432	69.1304 ± 2.5352
	77.5988 ± 1.3627	78.2609 ± 4.7628
	78.2127 ± 1.0464	72.1739 ± 3.7401
SVM	79.8450 ± 3.4144	72.6087 ± 2.2170
	81.6279 ± 8.3152	64.7826 ± 7.0644
	78.6822 ± 3.2613	75.2174 ± 3.2536

models. This indicates that with an increase in training points, the models continue to struggle to accurately predict whether residues mutate irrespective of the dataset split, suggesting that there may be variance within the dataset itself.

The number of residues within a 15% subset may be too small to gauge the variance between the models accurately. Therefore, a 70%-30% train-test split was tested, and the results of the averaged runs are seen in Table 2.10. From these results, it appears that the current limit of the dataset is around 76% accuracy. Additional samples following the same distribution would be required to potentially improve the average accuracy of the models. Interestingly, the standard deviations of the RF and SVM model accuracy appear to have decreased compared to Experiment 1, in which the validation subset was present. This may be due to the increased number of points in the testing subset, which reduced the impact of any false predictions for this subset.

2.7.3 Conclusion

This experiment showed that splitting the dataset into training and testing subsets only did not significantly improve model performance. While increasing the number of data points in the training and testing sets showed slight improvements in training accuracy, the models continued to struggle with accurately predicting residue mutations in the test subset, where a limit of around 76% accuracy was reached. The standard deviations of the

Table 2.10: Average accuracies of the ML models with a 70%-30% train-test split

Model	Training Set Accuracy (Mean ± SD)	Testing Set Accuracy (Mean ± SD)
ANN	85.1887 ± 1.4493	71.5217 ± 2.7840
	84.9057 ± 1.5212	73.2609 ± 5.3515
	82.4528 ± 1.8487	71.9565 ± 5.7680
RF	76.9767 ± 2.2522	77.1739 ± 3.3678
	77.9048 ± 1.3654	74.3478 ± 1.8952
	78.3123 ± 2.0058	76.0870 ± 3.7653
SVM	85.7547 ± 4.9777	69.1304 ± 3.4096
	79.2453 ± 2.4419	71.9565 ± 4.0320
	82.2642 ± 3.1657	69.7826 ± 2.8676

model accuracies decreased slightly, suggesting an increase in the dataset size is advisable.

2.8 Summary

In this chapter, classification and regression ANN, SVM and RF models were designed and implemented, with the goal of improving the overall performance of these models to achieve higher accuracy and R-values than the ML models in Barozi *et al.* (2024). However, these average values were not improved for any of the models, even when applying techniques such as hyperparameter tuning and PCA. The variance in accuracy for the random models may have been reduced when compared to those in Barozi *et al.* (2024), but these variances remain large.

It was observed that none of the model types appeared to outperform the others, suggesting that there was not a large difference between using different ML models for residue mutation predictions. All the current ML models require either further tuning, more input data samples, or different input features to potentially improve their performance. Another option would be to change the type of input provided to the ML models instead of features derived from MD trajectories.

Chapter 3

Prediction of Mutation Sites in SARS-CoV-2 M^{pro} using CNNs with Raw Trajectory Data

3.1 Introduction

The previous chapter focused on residue mutation predictions using input features derived from the SARS-CoV-2 M^{pro} MD simulation trajectories. However, the SVM, ANN, and RF models did not achieve high accuracies or R-values on these datasets, and only low or moderate performance was achieved for both the classification and regression tasks. Instead of further refining these models, this chapter will use the raw M^{pro} MD trajectories as ML inputs (i.e. coordinates of each protein atom over time). This input is tested to determine whether crucial patterns were lost due to the processing methods implemented to produce the features in the original M^{pro} dataset. However, this change in input data necessitates applying a different type of ML model, namely a convolutional neural network (CNN). This new dataset with the new ML model is investigated in this chapter to determine whether an improved performance is achieved compared to the M^{pro} dataset and models previously used in Chapter 2.

3.2 Literature Review

3.2.1 CNN Models

A CNN is a specialised artificial neural network designed to process structured data with spatial or temporal components (Wu, 2017). It is particularly suited for handling grid-like data such as images (Wu, 2017), videos, and spatiotemporal data (Chen *et al.*, 2021). Unlike traditional neural networks, CNNs can automatically detect and learn features directly from raw input data, making them especially useful for prediction tasks with image and sequence inputs.

CNN models are commonly applied to problems solved using pattern recognition (Wu, 2017). CNNs are mainly used for image processing, such as classifying images into different groups and detecting objects within images. Similarly, CNNs are used for video analysis, which introduces a temporal element compared to image processing, allowing for movement detection and tracking (Chen *et al.*, 2021). Natural language processing and audio or signal processing are also possible with the CNN models for text and sound classification. Finally, these models have also been applied to medical and scientific outputs, such as CT scans, MRI medical images and biological data.

3.2.2 CNN Key Components

CNNs are composed of several key layers, each with specific functions that contribute to the network's ability to automatically learn and extract meaningful features from raw input data (Yamashita *et al.*, 2018). The primary components of a CNN are the convolutional blocks (composed of a convolutional layer, activation function and pooling layer), fully connected layer, normalisation layers, dropout layers and output layers. These layers are detailed below.

3.2.2.1 Convolutional Layers

The convolutional layers are the core layers of CNN models, as they are responsible for feature extraction (Vakalopoulou *et al.*, 2023). Each of these layers applies a series of convolutional filters (also called kernels) to the input data (if it is the first convolutional layer in the model) or feature map (for subsequent convolutional layers). Each filter is a small matrix that slides over the input data, performing element-wise multiplication and summation at each position (O'Shea, 2015). This operation captures spatial relationships and local features (Yamashita *et al.*, 2018). This could be edges, shapes or textures for inputs such as images and videos.

The output of this layer is a feature map, which represents the specific features extracted from the previous input for certain regions of the input data (Yamashita *et al.*, 2018). The initial convolutional layer allows CNNs to detect low-level features, such as edges. However, higher-level, more abstract features like full objects or overall shapes are detected in the subsequent convolutional layers when provided with image input data.

3.2.2.2 Activation Function

Following the application of the convolutional filters, the resulting feature map passes through an activation function. An activation function is applied to introduce non-linearity into the network (Vakalopoulou *et al.*, 2023). This is critical as it allows for the model to identify complex, non-linear relationships that may be present in the data. There are many activation functions, the most common of which is the Rectified Linear Unit (ReLU), which is the default in tools such as Tensorflow due to the simplicity and effectiveness of the function. Other activation functions such as sigmoid and tanh may also be used (Vakalopoulou *et al.*, 2023).

3.2.2.3 Pooling Layer

The pooling layer is used to reduce the spatial dimensions of the feature maps after using the activation function (Wu, 2017). The reason for including this reduction layer is to

decrease the number of parameters and computations for the network, improving efficiency (Vakalopoulou *et al.*, 2023). Additionally, pooling helps make the network more robust to distortions in the input data.

This layer is implemented by sliding a window over the feature map and selecting the most important value within the window (Vakalopoulou *et al.*, 2023). Two common pooling operators are max pooling and average pooling (Zafar *et al.*, 2022). When using max pooling, the maximum value from each window is chosen as the output value. However, in average pooling, the average value of each feature map value within the window is computed and selected.

3.2.2.4 Fully Connected Layers

The fully connected layer (also called a dense layer) comes at the end of the CNN, where it connects every neuron from the previous layer to each neuron in the current layer (Wu, 2017). These layers combine the features learned by the convolutional and pooling layers to make final predictions for classification or regression tasks. These layers are similar to the original ANN layers (Wu, 2017).

The output of the last convolutional or pooling layer is flattened into a one-dimensional vector before being passed into the fully connected layer (Yamashita *et al.*, 2018). This is performed by flattening, which combines these features into a single vector, which contains all the information learned by the CNN within the convolutional blocks, which is necessary for making predictions.

3.2.2.5 Output Layer

The output layer is the final dense layer, which produces the predictions of the CNN model (Yamashita *et al.*, 2018). This layer is crucial because it converts the high-level features extracted by the previous convolutional and fully-connected layers into meaningful outputs for specific tasks, such as class probabilities for classification tasks or continuous values

for regression tasks. This is achieved by applying an activation function matching the task and implementing a single neuron in this layer to allow for the correct number of outputs provided as a prediction.

3.2.2.6 Normalisation Layers

Normalisation layers are used to improve training efficiency and convergence. The most common normalisation technique in CNNs is Batch Normalization. This technique normalises the activations within a mini-batch by adjusting and scaling them (Ioffe, 2015). This helps the network to train faster, reduces overfitting, and often improves the final performance. It works by subtracting the mean and dividing by the standard deviation of each activation in the batch, followed by scaling and shifting with learnable parameters.

3.2.2.7 Dropout Layers

The dropout layer is a regularisation technique that helps to prevent overfitting, particularly in large networks (Hinton, 2012). During training, dropout randomly deactivates a fraction of neurons in the network by setting their output to zero. This prevents the model from relying too heavily on any one feature or neuron and forces it to learn more robust features. Dropout is typically applied to fully connected layers and has been shown to significantly improve the generalisation ability of CNNs, especially when working with large and complex datasets (Park and Kwak, 2017).

3.2.3 CNN Model Application on Spatio-temporal Data

CNNs are increasingly utilised for analysing spatio-temporal data, encompassing spatial and temporal dimensions (Chen *et al.*, 2021). In previous studies, MD simulation data have been used successfully as input data for CNNs. For example, Berishvili *et al.* (2019) used MD simulation data with CNNs to predict ligand-protein binding affinities. In addition, Fukuya and Shibuta (2020) employed CNNs to classify atoms as belonging to

solid or liquid phases within a solid-liquid biphasic system. Building on this foundation, the current study aims to apply a CNN model to the raw trajectory data from MD simulations of the SARS-CoV-2 M^{pro}. This approach seeks to evaluate whether a CNN trained directly on raw trajectory data can outperform models that rely on predefined features extracted from the same data. By using the CNN's ability to learn hierarchical features automatically, the study aims to enhance the accuracy and robustness of mutation predictions for M^{pro}.

3.3 Methodology

3.3.1 Conversion of Trajectories to Valid CNN Input

The trajectories produced from simulating the M^{pro} protein (as a homodimer) using the GROMACS software resulted in .xtc coordinate and .pdb topology files. Both of these file formats are not valid inputs for any Python CNN architecture. Therefore, it was necessary to convert the information in these files to a format compatible with CNN models produced using Tensorflow 2.18.0 (Abadi *et al.*, 2015). An array was chosen as the input format required for these models using the NumPy package (version 2.0.2) (Harris *et al.*, 2020). The MDTraj package (version 1.10.2) (McGibbon *et al.*, 2015) was employed to read each trajectory's coordinate and topology files.

The shape of the array must be determined from the trajectory data to create the input array. One consideration is that the target data (mutation frequency) is defined for each residue, while the coordinates in the trajectory files are per atom for each residue. In addition, there are 20 unique residues, each with uniquely named atoms based on the naming conventions of GROMACS (e.g. for the element carbon, you have CA, CB, CG, CZ, CG, CD, CD1, CD2, etc). In total, there were 82 possible names for each atom of all 304 residues in Chain A of the M^{pro} protein. It was necessary to ensure that for each residue, the coordinates of each atom within the residue were included. Still, distinctions between atom names were kept (e.g. the coordinates of CG were distinct from other

carbon atoms, such as CA for each residue). Therefore, five dimensions were identified - the number of trajectories (6), frames per trajectory (50 000), residues per frame (304), maximum distinct atoms for all residues (82) and number of coordinates per atom (3).

An in-house Python script was coded and run, reading each coordinate file and topology file using MdTraj. From this, a new Numpy array was created for each trajectory, initialised with 0 values with size (50 000, 304, 82, 3). The coordinate values were added for each valid atom in the trajectory for all frames. All remaining elements in the arrays remained 0 (representing no possible coordinates for atoms that did not exist in the trajectory for a given residue). These arrays were saved using the Numpy array binary format (.npy files). Subsequently, these six arrays were concatenated and saved to form one NumPy array with size (6, 50 000, 304, 82, 3).

However, the order of the dimensions was not compatible with the architecture of the CNN, as it was required that the first dimension (currently six trajectories) was required to match the sample size for the target data (304 residues). Therefore, the array shape needed to be converted from (6, 50 000, 304, 82, 3) to (304, 82, 3, 50 000, 6). Due to the large size of the NumPy array, the array was converted to a NumPy memory map during this step to allow for better memory usage when loaded.

3.3.2 CNN Model Design

The CNN models were developed to analyse the multidimensional data of the simulated M^{pro} protein trajectories. As the predictions will be performed per residue, the input shape required for the model is four dimensions, representing the maximum possible atom names for any residue (82), the coordinates of the atom (3), the number of time frames of each trajectory and the number of trajectories run (6). The number of time frames varied, ranging between 1000 and 10 000 out of the total 50 000 frames available from the trajectories. This was done due to computational and time constraints that prohibited using all time frames. The model architecture was produced to extract the spatial and temporal features from the trajectories, using 3D convolutions followed by pooling and normalisation for multiple layers.

The initial CNN regression model consists of an input layer, three convolutional blocks, a flattening and fully connected layer, and an output layer. The input layer was designed to accept the residue samples using the abovementioned shape. The first convolutional block applied a 3D convolutional layer with 32 filters, a 3×3×3 kernel size and ReLU activation. This layer extracted low-level spatial and temporal features from the input data. A max-pooling layer with a pool size of 2×1×2 was used to reduce the dimensionality along the respective axes, enabling computational efficiency while preserving critical features. Finally, batch normalisation was applied to stabilise and accelerate training by normalising the activations at the end of the block.

Two additional convolutional blocks were added to continue the feature extraction process. The second convolution layer was designed with 64 filters, while the third used 128 filters with the same kernel size and activation function as the first layer. After the convolutional layers, the max pooling and batch normalisation layers were also employed for both blocks.

The output of the final convolutional block was flattened into a one-dimensional vector, representing the extracted features from the input data. This vector was then passed through two fully connected layers. The first fully connected layer contained 128 neurons with ReLU activation, followed by a dropout layer with a rate of 0.5 to reduce overfitting. The second fully connected layer included 64 neurons with ReLU activation. These layers enabled the model to learn non-linear relationships within the extracted features.

The output layer consisted of a single neuron with a linear activation function, producing a continuous regression output for each sample. The model was compiled using the Adam optimiser with a learning rate of 0.001, the mean squared error was selected as the loss function, and the mean absolute error was the chosen metric for evaluating the model's performance.

Other regression CNN model architectures were designed and tested. For example, the pool size of 2x1x5 for each max-pooling layer and an additional convolutional block, including the convolutional layer with an additional 128 kernels, were tested. For the classification CNN models, the output layer was redesigned to include the sigmoid activation function, producing a probability range of (0,1], beneficial for binary classification

in which the classes are either 0 or 1. Similarly, the pool size in each layer was also tested for these models, increasing the size to 2x1x5.

3.3.3 CNN Model Implementation and Analysis

The M^{pro} trajectory input dataset was split using the train_test_split function from Scikit-learn using stratified sampling into a 70-15-15 train-validation-test split. Each CNN model (both for regression and classification tasks) was trained using the fit method from Tensorflow using the training subset. Training was conducted using an epoch size of 1000 and a batch size of 4. An early stopping mechanism was included to monitor model performance during training and prevent potential overfitting of the model with the large epoch size. The EarlyStopping Keras API class was chosen to track the validation loss of the validation set and stop the model's training after 10 epochs of validation loss failing to improve (patience 10). The weights found at the point of the lowest validation loss were then reverted. A custom memory usage callback was also employed to monitor the GPU memory at each epoch and ensure the computational resources were managed correctly.

Training and testing of these models were performed on the Alvis HPC cluster, provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS). For this study, when training on the input dataset with the first and spread 1000 trajectory frames, 1 NVIDIA T40 with 16 GB VRAM and 1 x 16 core Intel(R) Xeon(R) Gold 6226R CPU and 72 GB system memory were used. With 2000 frames, this increased to 2 NVIDIA T40s and 8 CPUs, with the specifications listed above. For the 5000 and 10 000 frame trajectories, 4 NVIDIA A40 GPUs (each with 48 GB VRAM) and 2 x 32 core Intel(R) Xeon(R) Gold 6226R CPUs with 256 GB system memory were chosen for training the model. In addition, the GPU acceleration was employed using CUDA version 12.1.1.

3.4 Results and Discussion

3.4.1 Regression

Various architectures were tested for the regression CNN to determine the best model, resulting in the highest average R-values. This tuning was performed using only the first 1000 frames of the M^{pro} trajectories. The resulting R-values for each subset, when run using different architectures, are shown in Table 3.1. Low average R-values under 0.3 were achieved when testing the different CNN architectures. The most improved performance, when compared to that of the initial architecture, was obtained by increasing the `patience` parameter to 10, increasing the batch size to 8, and increasing the kernel size from (2,1,2) to (2,1,5). However, the subsets' average R-values between 0.35 and 0.45 are still very low. Consideration must be taken that only the first 1000 frames of each 50 000-frame trajectory file for the simulated M^{pro} protein were used.

Table 3.1: Average R-values of the CNN models with different parameters on features from averaged trajectories over the first 1000 frames

Model	Training Set R-value (Mean ± SD)	Validation Set R-value (Mean ± SD)	Testing Set R-value (Mean ± SD)
100 epochs, batch size 8, dropout 0.5, patience 5, kernel (2,1,2), 3 layers	0.1147 ± 0.0372	0.1318 ± 0.1148	0.0178 ± 0.1994
100 epochs, batch size 8, dropout 0.5, **patience 10**, kernel (2,1,2), 3 layers	0.2650 ± 0.0330	0.2487 ± 0.1149	0.1872 ± 0.1436
100 epochs, **batch size 4**, dropout 0.5, **patience 10**, kernel (2,1,2), 3 layers	0.1441 ± 0.1044	0.2319 ± 0.0815	0.0987 ± 0.2007
1000 epochs, batch size 8, dropout 0.5, **patience 10**, kernel (2,1,2), 3 layers	0.1066 ± 0.1053	0.1937 ± 0.0781	0.1098 ± 0.1220
100 epochs, batch size 8, dropout 0.5, **patience 10**, **kernel (2,1,5)**, 3 layers	0.3983 ± 0.0720	0.4653 ± 0.1674	0.3488 ± 0.1307
100 epochs, batch size 8, dropout 0.5, **patience 10**, kernel (2,1,2), **4 layers**	NaN	NaN	NaN

To determine whether the low average R-values were potentially caused by the small subset

of frames used for training the model, the best-performing CNN architecture identified previously was then trained, validated and tested on input data produced from double the frames (2000). In addition, it was unclear whether a difference would be seen if the frame sections were chosen from different parts of the trajectories. Therefore, the results from using datasets containing the trajectories from the first 1000 and 2000 frames were compared to those using every 50th frame (spread 1000) and 25th frame (spread 2000) from each trajectory. The results when using different numbers and distributions of frames, using one randomly generated dataset split, are shown in Table 3.2.

The average R-values for all of these combinations of frames were very similar, with R-values of around 0.4 for the training and validation subsets and 0.54 for the testing subset. This suggests no clear improvement when using a particular spread or number of frames from the trajectory tested here. While these R-values appear to be higher (particularly for the testing set) than the average R-values found for the first 1000 frames in Table 3.1, this could be attributed to the singular random split chosen rather than the models generalising well to the test data.

Table 3.2: Average R-values of the CNN model on features from averaged trajectories over varying numbers of frames

Model	Training Set Accuracy (Mean ± SD)	Validation Set Accuracy (Mean ± SD)	Testing Set Accuracy (Mean ± SD)
First 1000 time steps	0.3797 ± 0.0822	0.4017 ± 0.0664	0.5331 ± 0.0940
Spread 1000 time steps	0.3987 ± 0.0469	0.3972 ± 0.0272	0.5929 ± 0.1228
First 2000 time steps	0.4128 ± 0.0222	0.3783 ± 0.0208	0.5425 ± 0.0299
Spread 2000 time steps	0.4091 ± 0.0241	0.3871 ± 0.0711	0.5473 ± 0.0347

The number of frames used as input increased to 5 000 and 10 000, as the previous highest number of 2000 used was still a small percentage of the 50 000 overall frames saved per trajectory. However, due to the change in inputs, the architecture of the CNN models needed to be tuned again. Due to computational constraints, the new CNN models with the updated inputs were run only once with the single random split chosen before. The R-value results for each subset of this random split are shown in Table 3.3.

Model	Training Set R-value	Validation Set R-value	Testing Set R-value
Spread 5000 time frames: 100 epochs, batch size 8, dropout 0.5, patience 10, kernel (2,1,5), 3 layers	0.3076	0.4155	0.5081
Spread 5000 time frames: 100 epochs, **batch size 4**, dropout 0.5, patience 10, kernel (2,1,5), 3 layers	0.3783	0.2472	0.4606
Spread 5000 time frames: 100 epochs, batch size 8, dropout 0.5, patience 10, **kernel (2,1,10)**, 3 layers	0.3959	0.3589	0.5297
Spread 10 000 time frames: 100 epochs, batch size 8, dropout 0.5, patience 10, kernel (2,1,5), 3 layers	0.2884	0.5620	0.5328
Spread 10 000 time frames: 100 epochs, **batch size 4**, dropout 0.5, patience 10, kernel (2,1,5), 3 layers	0.0774	0.4865	-0.0577
Spread 10 000 time frames: 100 epochs, batch size 8, dropout 0.5, patience 10, **kernel (2,1,10)**, 3 layers	0.3923	0.3617	0.4799
Spread 10 000 time frames: 100 epochs, batch size 8, dropout 0.5, patience 10, **first layer kernel (2,1,20)**, rest kernel (2,1,10), 3 layers	0.3848	0.4720	0.4806

Table 3.3: Average R-values of the ML models after hyperparameter tuning

The R-values for the 5 000 frames dataset were similar to the results found in Table 3.2 for the 1000 and 2000 frames datasets. In particular, there was no visible, sustained increase in the R-values compared to the previous results. The R-values are sometimes lower than the average R-values for some of the architectures. However, this may be due to the constraint of only being able to run the model once and not taking an average of 5 runs for each architecture. Interestingly, the R-values resulting from the random split used on the dataset with 10 000 frames with all architectures were slightly lower, especially for the training or testing subset. The average R-value for the testing set was below 0.5, with one architecture resulting in a negative R-value. These reduced R-values suggest that increasing the number of time frames from the trajectories does not necessarily improve the regression models' performance and may actually result in reduced performance.

3.4.2 Classification

Following the implementation and analysis of the regression CNN models, classification models were designed based on the regression model architectures with the best performance and the first CNN architecture tested. The average accuracies and standard deviations for these classification models are provided in Table 3.4 using the first 1000 time frames of each trajectory as input.

The average accuracies of each subset were found to be around 70% for the training subset, 71% for the validation subset and 69% for the testing subset. There was a slightly improved average accuracy for all subsets when using the original architecture (pooling filter (2,1,2) and patience 5). Still, it is unclear whether this was due to differences in the random splits of the dataset or whether the architecture itself caused this improvement. The similar accuracy values for all subsets indicate that the models are not overfitting, compared to the regression models. However, when further studying the predictions of individual model runs, it was evident that only one class (class 0 for non-mutating residues) was consistently predicted for most of the tests. This was causing the resultant 70% accuracy for each subset, as around 70% of the residues were determined not to mutate from the GISAID data.

Table 3.4: Average accuracies of the CNN models with different parameters on features from averaged trajectories over the first 1000 frames

Model	Training Set Accuracy (Mean ± SD)	Validation Set Accuracy (Mean ± SD)	Testing Set Accuracy (Mean ± SD)
100 epochs, batch size 8, dropout 0.5, patience 5, kernel (2,1,2), 3 layers	70.1887 ± 0.3530	71.3043 ± 1.6268	69.5652 ± 2.3814
100 epochs, batch size 8, dropout 0.5, **patience 10**, **kernel (2,1,5)**, 3 layers	69.9057 ± 0.4622	70.8696 ± 2.6087	68.6957 ± 1.7391

Similar to the regression models, the CNN model with pooling size (2,1,5) and patience 10 was chosen for further analysis, where the model was run using input data from the spread 1000 and 2000 frame datasets. The resultant average accuracies and standard deviations from these inputs, keeping the same random initial split of the data, are shown in Table 3.5. The training, validation and testing set accuracies remained the same for the random split chosen when using the 1000 and first 2000 time frame inputs. The predictions for all these runs were given as class 0, indicating that the model consistently predicted that the residues in all the subsets would not mutate. The accuracies for the spread 2000 dataset improved slightly, where a few residues were correctly predicted to be mutating.

These results suggest that the low number of frames provided as input does not provide enough discriminatory patterns to determine whether residues are mutating. Improved accuracies may be achievable by providing larger numbers of time frames as input. However, a significant compute cost is associated with testing these models with larger input datasets, and the extent of the improvement may be minimal. The average accuracies produced from the RF model in Experiment 4 of the previous section is around 5% higher than the current values found for these CNN models. Considering this difference, it may be preferable to continue to improve the RF models by investigating further features and increasing the number of samples for testing.

The overall low performance of all tested regression and classification CNN models trained on the raw trajectory M^{pro} dataset raises questions about the reason for this low performance. In particular, it is interesting that it had overall lower R-value and accuracy

Table 3.5: Average accuracies of the classification CNN models on features from averaged trajectories over varying numbers of frames

Model	Training Set Accuracy	Validation Set Accuracy	Testing Set Accuracy
First 1000 time steps	69.8113	69.5652	69.5652
	69.8113	69.5652	69.5652
	69.8113	69.5652	69.5652
Spread 1000 time steps	69.8113	69.5652	69.5652
	69.8113	69.5652	69.5652
	69.8113	69.5652	69.5652
First 2000 time steps	69.8113	69.5652	69.5652
	69.8113	69.5652	69.5652
	69.8113	69.5652	69.5652
Spread 2000 time steps	71.2264	69.5652	69.5652
	71.2264	73.9130	69.5652
	69.8113	69.5652	69.5652
	71.6981	76.0870	71.7391
	69.8113	69.5652	69.5652

scores compared to the previous models trained on the processed M^{pro} trajectory data. While this may indicate the incorrect choice of ML model for this study, there is also the possibility that certain assumptions may have been incorrect for the CNN input data.

3.4.3 Potential Issues with the CNN Input Data

One of these potential issues is that the coordinate data within the 5D array was never normalised using any feature scaling. Without normalisation, the patterns within the data, particularly the change in the position of the atoms, may not be as prevalent. Instead, the atoms with higher coordinate values may be erroneously thought to be of higher importance when it is only the residue atoms' unit position within the simulation. There is also a large class imbalance within the dataset, where only 30% of the residues mutate. Even with the stratified sampling included to try and mitigate this imbalance, there may have been too few examples of this class from which the CNN could identify any potential patterns shared by these residues.

A decision was made to encode any atoms that do not exist for certain residues with

(0,0,0) values for the (x,y,z) coordinates. However, this may have been an error, and another technique, such as masking, could have been employed, as the models may not have interpreted these zero values as atoms that did not exist. Also, this made the dataset very sparse, which may have affected the model's overall performance. Further study would need to be conducted to determine whether this had any effect, where the coordinates of only the carbon alpha atom of each residue were included in the dataset.

The raw coordinates for each atom were added to the 5D array based only on the order in which the atom types were identified. However, both Berishvili *et al.* (2019) and Fukuya and Shibuta (2020) that implemented CNN models with MD simulation trajectories as input data processed these trajectories in a different manner. Instead of providing each atom's x,y and z positions as values, three dimensions of the input data were created to form a 3D grid. Each value within the input represented a section of the 3D space of the simulation, where a value of 0 indicated the absence of an atom in that area and 1 for the presence of an atom. It has been suggested that CNN models perform better with spatial data in a grid-like configuration like this, such as the pixels of images. However, this may not work for our prediction, in which it is important to group the specific atoms for each residue. Other ML models, such as graph neural networks (GNNs), which are based on empirical distance, may result in improved performance.

3.5 Summary

A new input dataset was created using MDTraj, which contained the coordinates of each atom per frame in each M^{pro} trajectory. To evaluate this new dataset, various CNN regression and classification models were applied to a subset of the frames, as the number of frames was limited by the computational power of the hardware. The overall performance of all tested CNN models resulted in lower average R-values and accuracies compared to the ML models tested in the previous chapter. However, this subpar performance may not result from the choice of ML model but rather the format in which the input data was provided.

Chapter 4

Simulations and Analysis of the NSP10-NSP16 Complex to Investigate Generalised SARS-CoV-2 Residue Mutation Predictions

4.1 Introduction

The SARS-CoV-2 M^{pro} protein has previously been simulated using MD simulations, and the resultant trajectories have been processed using techniques such as DRN analysis as detailed in Barozi *et al.* (2024). These processed trajectories formed the basis of the M^{pro} dataset used to train ANN, RF, and SVM models in Chapter 2. While these models achieved moderate performance when tested on M^{pro} residues within the test set, their ability to generalise to other SARS-CoV-2 proteins remains unexplored. Therefore, this chapter extends this analysis, dataset creation and prediction to the NSP10-NSP16 complex, evaluating whether the ML models trained on M^{pro} are robust mutation predictors for other SARS-CoV-2 proteins.

This chapter is split into three main parts. Part I involves running MD simulations of

the NSP10 and NSP16 proteins in their complex form. To achieve this, a suitable Protein Data Bank (PDB) file for the complex is identified, processed and used in initial, short MD simulations to determine the best set of MD parameters. Then, multiple replicates of MD simulations are run to ensure stable and equilibrated trajectories for the protein chains are generated.

In Part II, the MD trajectories undergo further processing, including DRN analysis, to extract dynamic network centrality values. This ensures that the datasets for NSP10 and NSP16 are constructed using the same feature set as the M^{pro} dataset, allowing for meaningful comparisons. Additionally, the classification of mutated and non-mutated residues within NSP10 and NSP16 is examined, focusing on their placement within specified DRN metric bins, as performed for M^{pro} in Barozi *et al.* (2024).

Part III evaluates the performance of the ANN, RF, and SVM models trained on the M^{pro} dataset when tested on the newly generated NSP10 and NSP16 datasets. This is performed to assess the generalisability and robustness of these M^{pro}-trained models when applied to other SARS-CoV-2 proteins.

4.2 Part I: Processing and MD Simulations of the SARS-CoV-2 NSP10-NSP16 Complex

4.2.1 Literature Review

4.2.1.1 MD Simulations

MD simulations have revolutionised computational biophysics by providing new insights into the dynamic behaviour of biological molecules, particularly proteins, at the atomic level (Rapaport, 2004). Proteins are dynamic structures crucial for biological processes, including enzymatic reactions, signal transduction, and molecular recognition. MD simulations offer an approach to studying their dynamics by numerically solving Newton's

equations of motion for a system of interacting atoms. The trajectory generated by MD simulations provides a detailed timeline of atomic positions and velocities within the protein structure (Lemkul, 2018). Analysis of this trajectory involves computing metrics such as root mean square deviation (RMSD), root mean square fluctuation (RMSF), solvent accessible surface area (SASA), radius of gyration (Rg) and hydrogen bonding patterns. These metrics can be used to determine protein stability, flexibility, binding affinities, and allosteric mechanisms, potentially resulting in insights into biological function and aiding in drug discovery efforts. Changes in these protein properties can also be described by comparing the simulations of wild-type and mutant proteins.

4.2.1.1.1 Basic Principles of MD Simulations

At the core of MD simulations lies Newton's second law of motion, in which the equations of motions for the system of interacting atoms are solved. To solve these equations, the initial positions, velocities, and a description of the total potential energy as a function of the atomic coordinates must be provided. The formula for Newton's Second Law can be written as $F = ma$, where F denotes the force applied to a particle, m its mass, and a its acceleration. The force F of this formula can be calculated by determining the negative derivative of the total potential energy of the system.

Quantum and molecular mechanics are two main physical theories that can be used to model interactions between atoms (Braun *et al.*, 2019). The quantum mechanics theory explicitly represents electrons in the system and solves the electronic structure of the system with few to no empirical measures to determine the overall interaction energy of the system. However, the molecular mechanics theory represents the atoms in the system as particles, with no electrons represented. Instead, each particle is given an electric charge and electrostatic potential, with many empirical parameters used to determine the interaction energy between the atoms in the system. A molecular mechanics model is preferred in most MD simulations because it is significantly simpler and faster to solve than quantum mechanics. However, this comes at the cost of reduced accuracy due to the approximations inherent in using empirical parameters.

4.2.1.1.2 Molecular Interactions and Force Fields

The interactions between atoms in a molecular mechanics system can be categorised into two main types, which are bonded and non-bonded potentials (Vanommeslaeghe *et al.*, 2014). Bonded potentials describe the interactions between atoms that are covalently bonded to each other. These bonded potentials include bond stretching, angle bending, and torsional rotations of proper and improper dihedrals. Parameters for bonded potentials are derived from experimental data and quantum mechanical calculations. Non-bonded potentials describe the interactions between atoms that are not covalently bonded but interact through van der Waals forces (Lennard-Jones potential) and electrostatic interactions (Coulombic potential).

Force fields parameterise the potential energy function that describes the interactions between atoms in MD simulations (Lemkul, 2018). Each force field is different, as they include different interaction terms, mathematical forms and empirical parameter values (Braun *et al.*, 2019). The force field's potential energy function determines the simulated system's stability and dynamics, influencing key thermodynamic properties such as enthalpy, entropy, and free energy (Allen and Tildesley, 2017). Continuous refinement of force fields enhances the predictive power of MD simulations, enabling detailed investigations into biomolecular structures and dynamics (Lemkul, 2018). The optimal force field for a simulation is protein-dependent, so multiple force fields are tested to determine which model is most applicable and results in the most stable trajectory for the simulated system.

4.2.1.1.3 Thermostats and Barostats

Thermostats and barostats are essential tools used during the equilibration step of MD simulations to control the temperature and pressure of the system, respectively (Lemkul, 2018). They ensure the system reaches a stable temperature and density state required for an accurate simulation before the production phase. Thermostats adjust the randomly generated initial velocities of the particles to ensure that the system's kinetic energy corresponds to the desired temperature. On the other hand, barostats adjust the volume of

the simulation box to maintain the desired pressure. Several types of thermostats and barostats are used in MD simulations, each with its own characteristics and suitability for different studies. These include the Berendsen (Berendsen *et al.*, 1984), Anderson (Andersen, 1980) and Nosé-Hoover (Nosé, 1984, Hoover, 1985) thermostats and the Berendsen (Berendsen *et al.*, 1984) and Parrinello-Rahman (Parrinello and Rahman, 1981) barostats. Effective use of thermostats and barostats ensures accurate and stable simulation conditions, which is critical for generating reliable and reproducible results in MD studies (Lemkul, 2018).

4.2.1.2 Metal Parameterisation

As detailed in the literature review, the NSP10 protein of SARS-CoV-2 contains two zinc fingers that play a critical role in its structural integrity and function. The standard force fields commonly used in MD simulations, such as AMBER, CHARMM, OPLS, and GROMOS, do not include the parameters necessary to describe metal ion interactions such as zinc cofactors (Peters *et al.*, 2010). This limitation is caused by the complexity and specificity of metal sites within proteins, where coordination modes (e.g., tetrahedral, trigonal bipyramidal) and binding residues (e.g., cysteine, histidine) vary significantly between proteins and even between sites within the same protein (Hay, 1993).

To address this, three commonly used models have been developed to extend the AMBER force field for metal ion interactions:

1. **Bonded Model:** In this approach, covalent bonds are defined between the metal ion and its coordinating residues (Peters *et al.*, 2010). It models the interactions using bond, angle, dihedral, electrostatic and van der Waals terms. This model assumes a fixed geometry for the coordination site, making it suitable for systems with well-defined and rigid coordination geometry.

2. **Non-Bonded Model:** This approach models the interactions between the metal ion and the coordinating residues as non-covalent, typically using Lennard-Jones and electrostatic potentials (Li *et al.*, 2013). While this approach allows for greater

flexibility, it can lead to inaccuracies in cases where the coordination geometry is critical.

3. **Cationic Dummy Atom Model:** This method uses dummy atoms to represent the electrostatic field around the metal ion, allowing for a more accurate description of coordination without explicitly defining bonds (Li *et al.*, 2013). However, this model can be computationally demanding and complex to implement.

4.2.2 Methodology

4.2.2.1 Choice of PDB Structure for SARS-CoV-2 NSP10-NSP16 Complex MD Simulations

The PDB database (Berman *et al.*, 2000) was searched to determine the PDB IDs of all the NSP10-NSP16 complex structures for SARS-CoV-2. A suitable structure was selected from those identified based on their corresponding experimental and validation data from sources such as the wwPDB validation service (Berman *et al.*, 2003). Other criteria were also considered, such as the time the record was provided and whether the PDB structure had previously been used in simulations.

4.2.2.2 Preparation of NSP10-NSP16 Complex PDB File

First, the original PDB file with ID 6W4H (Minasov *et al.*, 2020) was downloaded from the PDB database. Using PyMOL (Schrödinger, LLC, 2015), non-protein molecules, such as solvent (H_2O) and ligands (SAM, BDF, ACT, and SO_3), were removed. The SAM cofactor was excluded from the file, as it required separate parameterization for inclusion in the MD simulations. In contrast, the BDF, ACT, and SO_3 ligands were only necessary for the crystallization process and were therefore removed for MD simulations. Any alternate locations of sidechains (rotamers) were also removed, resulting in the single conformation of the protein necessary for MD simulations. Furthermore, the first residue of NSP16 was removed from the file, as this residue was an artefact of protein expression

and was not present in the UniProt sequence (UniProt ID P0DTD1) for SARS-CoV-2 NSP16. Although the presence of this residue may have influenced the local conformation of neighbouring residues, its removal ensures a more biologically accurate representation of the protein. Any potential steric clashes or conformational changes introduced by this residue are expected to be resolved after removal of the artefact during the energy minimization step before the MD simulation. The two zinc ions were selected and saved to separate PDB files, as they are required in further steps, but are removed from the file during protonation by the protonation server H++ (Anandakrishnan *et al.*, 2012). Finally, the anisotropic temperature factor records (ANISOU) present in the original PDB file were removed.

The cleaned PDB file was then renumbered using the `pdb4amber` command from AmberTools23 (Case *et al.*, 2023). However, this results in a PDB file in which atomic charges are removed, e.g. N instead of N+. These charges were re-introduced to the PDB file, and the resultant file was uploaded to the H++ server for protonation. Protonation can be performed on the protein complex or the separated proteins. However, this leads to differences in the overall pKa values and total charge, as the presence of both proteins induces environmental changes that are not accounted for when the proteins are treated separately. As the NSP10-NSP16 complex will be simulated using MD, the file containing the complex was chosen for protonation.

The pH chosen for protonation was 7.5, which has been experimentally determined to result in high percentage activity of the complex (Khalili Yazdi *et al.*, 2021) and is very close to the physiological pH of host cells. The remaining parameters were the default values, with a salt concentration of 0.15 M (physiological molarity), an internal dielectric of 10 and an external dielectric of 80. The resultant pKa for the complex was determined to be 7.81, with a total charge of 2. However, this total charge does not accurately reflect the actual total charge of the complex. This is because the H++ server cannot detect the zinc ions, which have a +2 charge each, and the residues coordinating the zinc atoms were incorrectly protonated.

The .top and .crd files produced by the H++ server were combined using the `ambpdb`

command from AmberTools23 to create a new PDB file containing the protonated complex. The MCPB.py parameter builder tool (Li and Merz, 2016) was chosen to create the bonded model. The manual recommends using this tool on PDB files with a single chain. Therefore, the two protein chains, NSP10 and NSP16, were split into two PDB files. The two zinc atoms were re-introduced to the NSP10 PDB file from the individual zinc PDB files using PyMOL. The HG atoms found on the CYS residues that coordinate the two zincs (positions 57, 60, 73, 100, 103, 111 and 113 for the renumbered NSP10 chain) were removed using PyMOL. This was performed to allow for the interactions between these residues and the zinc atoms. These CYS residues were renamed as CYM according to Amber naming conventions due to the removal of the HG atoms. The HID residue, which coordinates with the first zinc, was not changed as it was in the correct protonation state. These residue name changes allow the correct total charge to be calculated in further steps.

4.2.2.3 Formation of the Bonded Model

The two zincs are found in two different zinc-coordinating sites in the NSP10 protein. For the NSP10 protein, a geometrically accurate structure is essential to understand its functional dynamics, particularly in its role within the NSP10-NSP16 complex. Given the importance of maintaining the structural integrity of the zinc finger domains, the bonded model was chosen to create the required force field parameters for zinc coordination in NSP10. This approach ensures that the coordination geometry is preserved throughout the simulation, which is critical for capturing biologically relevant behaviour during simulations. Therefore, each zinc ion needed to be parameterised separately and a separate PDB file was produced for each zinc ion. Each zinc PDB file was converted to mol2 format in preparation for parameterisation using the *metalpdb2mol2.py* script from AmberTools23.

The *MCPB.py* software developed by Li and Merz (2016) was used to produce small and large models, and these were used as input files for the Gaussian09 program. Gaussian calculations for the small model for each zinc and the large model for the second zinc

were processed using the default basis set B3LYP/6-31G*, with 6GB of memory and eight shared processors on the Rhodes University Chemistry Department Carbon server. However, the first zinc large model would not reach an optimisation endpoint during the calculation of Merz-Kollman RESP charges when using the default Gaussian09 optimisation. Therefore, it was necessary to specify the calculation of the Hessian matrix for the initial geometry within the optimisation file header.

Following Gaussian calculations, the *MCPB.py* software was used to generate force field parameters for each zinc ion, employing the default Seminario method. RESP charge fitting followed using the default ChgModB option, resulting in mol2 files for each zinc-binding site's residues and metal ion. The last step of *MCPB.py* was performed, generating a template LEaP input file and PDB file for each zinc-binding site and the NSP10 chain.

Manual adjustments were made to generate a single LEaP input file, combining zinc binding sites and protein chains into one system. Firstly, this included uniquely renaming the mol2 files generated during RESP charge fitting. In addition, the renamed zinc coordinating residues for both binding sites were included in the single mcpbpy PDB file. Lastly, the force field parameters for each residue in the frcmod files for each zinc-binding site were added to the LEaP input file. The TIP3P water model, ff14SB (Maier *et al.*, 2015) and ff19SB (Tian *et al.*, 2019) force fields for proteins and GAFF2 force field (Wang *et al.*, 2004) for ligands were chosen. In addition, a salt concentration of 0.15 M was included in the system by manually determining the number of Na+ and Cl- ions to include using the methods provided by Amber Tutorial 8 (Held and Nagan, 2021). This is determined by converting the system's volume from Å to litres and using the number of chloride ions present per litre at a 0.15 M concentration. These values are multiplied and rounded to get the required number of chloride ions. Running the LEaP tool resulted in AMBER topology and coordinate files for this system. ACPYPE (Sousa da Silva and Vranken, 2012) was applied to these files to convert them to GROMACS format for MD simulations.

4.2.2.4 Parameterisation of the SAM Ligand

The H++ server could not recognise the SAM ligand, and an error occurred. Therefore, the SAM ligand structure from the PDB file with ID 6W4H was isolated using Maestro (Schrödinger, 2022b). The Protein Preparation Wizard (Madhavi Sastry *et al.*, 2013) was used to preprocess the SAM ligand by assigning bond orders, adding hydrogens, creating any disulphide bonds and predicting the correct protonation and tautomeric states for the ligand using Epik (Schrödinger, 2022a) at pH 7.5 ± 2. No problems were identified for the ligand when reviewed in the Wizard. Finally, the hydrogen bond assignment optimisations were performed using PROPKA with a pH of 7.5. The resultant structure for the SAM ligand was saved in PDB format.

Antechamber (Wang *et al.*, 2001) within UCSF Chimera v1.17.3 (Pettersen *et al.*, 2004) was employed to generate the topology files for the SAM ligand, by choosing to add charges to the PDB file. This resulted in a ligand with a total positive charge of 1, saved to mol2 format. ParmChk2 from AmberTools23 was subsequently utilised to create the force field parameters from the mol2 file. Both the topology file and force field parameters for SAM were essential for parameterising the ligand within the system. These files were combined with the protein chains and zinc binding sites in the input LEaP file. All the parameterised SAM ligand atoms (excluding the newly added hydrogen atoms) had identical coordinates when compared to the original structure. Therefore, the SAM molecule was reintroduced to the final protein complex using the original SAM molecule position rather than redocking the molecule.

4.2.2.5 MD Simulations of NSP10-NSP16 Complex

Energy minimisation of the NSP10-NSP16 complex was performed on a local Rhodes University Linux cluster using GROMACS 2021.4 (Páll *et al.*, 2020). The steepest descent algorithm was applied until the maximum force of the system fell below 1000 kJ/mol/nm or 50 000 steps were reached. Temperature equilibration was performed using the NVT ensemble at 310 K with V-rescale coupling for 100 ps. Subsequently, pressure equilibration

using the NPT ensemble was conducted at 1 atm and 310 K for 100 ps, coupled with either the Parrinello-Rahman or C-rescale barostats. Initial production runs of 50 ns for each system were used to determine the best combination of force fields and barostats. Final runs of 1 µs, both with and without the SAM ligand, were performed with 2 fs integration time steps on the CHPC cluster in Cape Town. The long-range electrostatics were tread using PME, with a Fourier spacing of 0.16 nm and 1 nm cutoffs for short-range electrostatics and van der Waals interactions. The LINCS algorithm (Hess *et al.*, 1997) was applied for periodic boundary conditions and bond constraints.

Following the simulation, each resultant trajectory was processed using the GROMACS `gmx` commands. First, the NSP10-NSP16 complex was centred, and the PBC conditions used during the simulation were removed. Each trajectory was visually inspected using Visual Molecular Dynamics (VMD) (Humphrey *et al.*, 1996) to determine whether each zinc ion remained within the binding site throughout the simulation. For systems containing the SAM ligand, VMD was also used to confirm that the ligand remained within the SAM binding site for the entire production run. Measures such as RMSD of the alpha carbons and RMSF were calculated using GROMACS.

4.2.3 Results and Discussion

4.2.3.1 Choice of PDB Structure for SARS-CoV-2 NSP10-NSP16 Complex MD Simulations

A high-quality 3D structure of the NSP10-NSP16 complex is required for MD simulations to be reliable and display accurate behaviour of this protein in its simulated environment.

The PDB database was searched and all 3D structure files for the SARS-CoV-2 NSP10-NSP16 complex were identified. The experimental data (Table 4.1) and wwPDB validation scores (Table 4.2) of these structures were compared for all the structures with resolutions under 2 Å.

Table 4.1: Experimental data for SARS-CoV-2 NSP10-NSP16 complex PDB entries.

PDB ID	Resolution (Å)	R-value Free	R-value Work	R-value Observed
8OV1	1.67	0.188	0.174	0.174
8RV8	1.70	0.196	0.187	0.187
8OTR	1.77	0.209	0.188	0.188
7L6T	1.78	0.162	0.141	0.142
9EMJ	1.79	0.207	0.181	0.181
6W4H	1.80	0.163	0.149	0.150
8BZV	1.80	0.200	0.178	0.178
8OTO	1.80	0.186	0.175	0.175
8RVA	1.80	0.204	0.196	0.197
6WKS	1.80	0.188	0.152	-
8OV3	1.82	0.211	0.182	0.182
8OSX	1.83	0.223	0.197	0.198
8OV2	1.86	0.198	0.177	0.178
7ULT	1.90	0.188	0.164	0.166
8C5M	1.90	0.211	0.184	0.184
8RV7	1.90	0.197	0.173	0.174
8RV9	1.90	0.208	0.186	0.188
8TYJ	1.90	0.201	0.184	0.185
8OV4	1.93	0.207	0.179	0.179
8BSD	1.95	0.207	0.180	0.181
8RVB	1.95	0.203	0.187	0.188
6W75	1.95	0.174	0.157	-

The PDB structure with PDB ID 80V1 had the highest resolution of 1.67 Å. Resolution is one of the important metrics for structures, as this is an indication that atom positions are more detailed and accurate. Compared to 8OV1, 8RV8, 8OTR, and 9EMJ have lower resolutions, worse R-value Free, Work and Observed values, as well as worse Clashscores, more Sidechain outliers and RSRZ outliers, meaning that these structures were not considered further. However, two other structures (IDs 7L6T and 6W4H) had better R-value Free, R-value Work and R-value observed values when compared to the 80V1 structure. They were, therefore, considered preferable to 80V1 despite having lower resolutions.

While 7L6T has slightly better metric values for RSRZ-outliers and clashscore than 6W4H, this structure includes a capped RNA substrate and the SAH ligand. On the other hand, the 6W4H structure contains the SAM ligand, and the RNA substrate is not present. This indicates that the 7L6T structure is in the post-reaction state, where the methyltransferase activity has occurred, while the 6W4H structure is in the pre-reaction state. Since the

Table 4.2: wwPDB Validation data for SARS-CoV-2 NSP10-NSP16 complex PDB entries.

PDB ID	Rfree	Clashscore	Ramachandram outliers (%)	Sidechain outliers (%)	RSRZ outliers (%)	ANISOU
8OV1	0.183	4	0.0	0.0	3.6	No
8RV8	0.196	6	0.0	0.6	5.1	No
8OTR	0.208	9	0.0	0.8	6.0	No
7L6T	0.168	2	0.0	0.0	10.6	Yes
9EMJ	0.199	2	0.0	0.6	4.8	No
6W4H	0.171	3	0.0	0.0	11.3	Yes
8BZV	0.195	3	0.0	0.6	4.8	No
8OTO	0.180	3	0.2	0.8	2.6	No
8RVA	0.206	2	0.0	0.9	4.4	No
6WKS	0.188	4	0.0	2.0	13.6	Yes
8OV3	0.207	3	0.0	0.8	8.2	No
8OSX	0.224	12	0.0	1.7	6.0	No
8OV2	0.197	4	0.0	1.4	9.9	No
7ULT	0.194	2	0.0	0.0	12.5	Yes
8C5M	0.208	6	0.0	0.3	1.9	No
8RV7	0.197	5	0.0	0.9	2.9	No
8RV9	0.205	4	0.0	0.9	3.6	No
8TYJ	0.201	3	0.0	1.1	4.1	No
8OV4	0.207	4	0.0	1.7	8.4	No
8BSD	0.205	3	0.2	1.1	1.7	No
8RVB	0.203	3	0.0	0.9	2.7	No
6W75	0.179	3	0.0	0.0	13.8	Yes

focus of this simulation is on the behaviour of the complex before the methyltransferase activity has occurred, the 6W4H structure is preferred to 7L6T.

PDB structures with IDs 9EMJ, 8OTO, 8TYJ, 8BSD and 8RVB had some of the lowest wwPDB validation scores, and it needed to be determined if any of these would be more suitable than 6W4H. However, the percentage of RSRZ outliers for 9EMJ and 8TYJ is the highest (both over 4%) compared to these other structures, so they were not considered further. Also, 8OTO and 8BSD had Ramachandran outliers, which were absent for any of the other identified structures. This left the structure with PDB ID 8RVB, which had reasonably low validation scores for all the wwPDB validation criteria, even though it has a lower resolution of 1.95 Å and slightly higher R-values than some of the other listed structures.

Therefore, the two remaining structures considered were 6W4H and 8RVB. 6W4H has a better resolution (1.67 Å), R-values, and sidechain outliers (0%) compared to 8RVB (resolution 1.95 Å, sidechain outliers 0.9%), and both had clash scores of 3 and no Ramachandran outliers. However, the RSRZ outliers percentage was much higher for 6W4H, at 11.3%, compared to the much lower 2.7% for PDB structure 8RVB. Further investigation was required to determine the reason for the high RSRZ value and whether 6W4H was viable.

Gore *et al.* (2017) defines RSRZ outliers as the "fraction of polypeptide and/or polynucleotide residues that do not fit the electron density well when compared with other instances of the same residues in structures at similar resolution". wwPDB uses a REFMAC program to create the electron density maps based on the PDB file and then determine the fit between the model and the 2Fo-Fc electron density map by calculating real-space R-value (RSR) (Worldwide Protein Data Bank, 2024). Next, they normalise the RSR score based on the residue type by comparing it to the RSR values of the same residues from other PDB files with similar resolutions, which results in the RSR Z-score (RSRZ) for each residue.

The locations of the RSRZ outlier residues visualised for both PDB files are shown in subfigures A and B of Figure 4.1 for 8RVB and 6W4H, respectively. This was performed in order to determine if these RSRZ outlier residues were located in functionally important sites or the less important flexible loops.

Subfigure A of Figure 4.1 suggests that most of the residues which are RSRZ outliers for the 8RVB structure are in the loop regions. However, subfigure B of Figure 4.1 indicates that many of the RSRZ outliers for the 6W4H structure are found in the loop regions and some of the beta sheets, which form part of the Rossmann-like fold. However, no RSRZ outliers were found in the Rossmann-like fold in 8RVB. As these beta sheets are integral to the functionality and structure of NSP16, it is imperative that the residues within these sheets are in the correct position.

After the alignment of the 6W4H and 8RVB structures seen in Figure 4.2, the positions of the residues with high RSRZ values in 6W4H were very similar to those in 8RVB (red

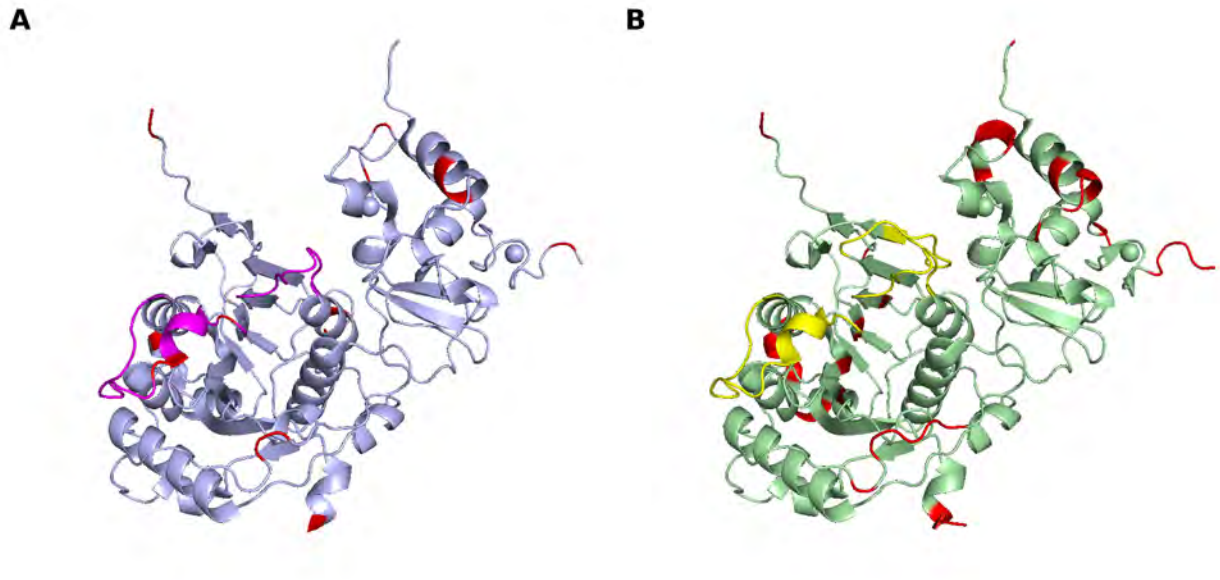

Figure 4.1: (A) 3D structure with PDB ID 8RVB. The red areas indicate residues which are RSRZ outliers, while the SAM binding pocket is shown in pink. (B) 3D structure with PDB ID 6W4H. The red areas indicate RSRZ outlier residues, while the SAM binding pocket residues are shown in yellow.

for 6W4H, cyan for 8RVB). However, these residues were classified as outliers for 6W4H and not 8RVB. This indicates that it was most likely not a change in residue position causing the outliers but potentially problems with the generated electron density map.

The 6W4H PDB file has both ATOM and ANISOU records (the records which hold the anisotropic temperature factors), while 8RVB only has ATOM records. According to the FAQ on the PDB site (Worldwide Protein Data Bank, 2024), there are three times that there may be issues when calculating the electron density maps using REFMAC:

- Lowish resolution structures that have been refined using Phenix where hydrogen atoms are involved.
- Low resolution anisotropic structures.
- Structures refined with Phenix twin option.

6W4H was refined using REFMAC, meaning that the first and third options do not apply to this structure. According to the REFMAC documentation (Collaborative Computa-

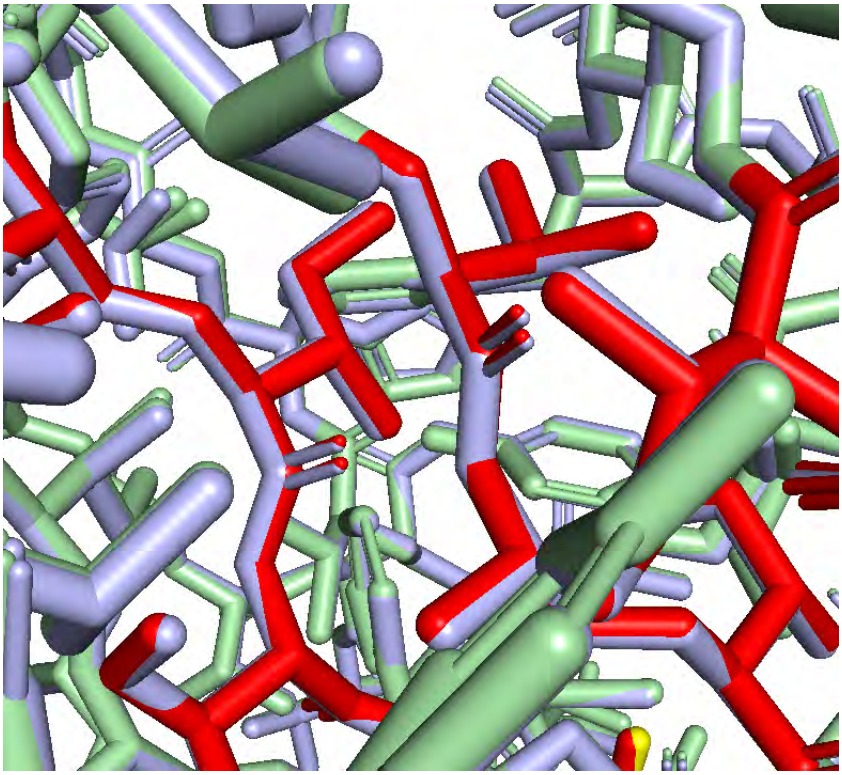

Figure 4.2: The superimposed 3D structures with PDB IDs 8RVB (blue) and 6W4H (green). The red areas indicate residues which are RSRZ outliers for 6W4H.

tional Project Number 4, 2024), anisotropic is defined as "Individual anisotropic B-factor refined for all atoms". 6W4H can, therefore, be considered to be an anisotropic structure. The second option may explain the high percentage of RSRZ outliers for 6W4H, as it has a resolution of 1.8, which is relatively low for calculating anisotropic temperature values. This may be causing the discrepancy between 6W4H and 8RVB for that metric. Interestingly, all five structures with resolutions under 1.95 Åusing anisotropic temperature factors had the highest RSRZ outlier percentages of all the identified NSP10-NSP16 complex structures.

Other validation tools were used to determine whether the 6W4H or 8RVB files contained a higher quality structure than the other. The score returned from the ERRAT program was 93.473 for the 6W4H structure, which was slightly higher than the 92.857 ERRAT score for the 8RVB structure. However, these scores are over 90.0, meaning this validation tool suggests both structures are of high quality. The ProSA tool was used next, where Z-scores of -7.72 and -3.06 were returned for the NSP16 and NSP10 chains of the 6W4H

structure, while -7.56 and -3.15 Z-scores were given for the 8RVB structure. These Z-scores fall within the range of typical proteins, indicating that both structures are of a reliable quality.

The 6W4H structure contains the SAM ligand, whereas an inhibitor is bound to the SAM-binding site of the NSP16 chain in the 8RVB file. The presence of an inhibitor can result in the sidechain positions of the binding site residues being affected by the inhibitor. Therefore, choosing a structure with a naturally occurring ligand that prevents this potential change in structure is preferable.

One important factor in this comparison is the evaluation of the zinc-binding sites in the NSP10 chain, as these sites are critical for the structural and functional integrity of the protein. Proper coordination geometries and consistent distances between the zinc ions and their coordinating residues are essential to ensure the accuracy and reliability of the structure. In the 6W4H structure, these distances were approximately 2.3 Å for cysteine (CYS) and 2.0 Å for histidine (HIS), closely matching the expected values. In contrast, the 8RVB structure showed slightly shorter zinc-CYS distances, ranging between 2.1 and 2.2 Å.

Lastly, the PDB structure with ID 6W4H has previously been used in MD simulations by other groups, such as Sk *et al.* (2020) and Eissa *et al.* (2022), demonstrating its suitability for such studies. In contrast, no MD simulations have been performed using the PDB structure with ID 8RVB. This lack of prior usage makes 6W4H the preferred choice for this research, as there is proof of successful MD simulations conducted using this structure, reducing potential uncertainties in the initial stages of the study.

Overall, the decision was made to use the PDB structure with ID 6W4H for MD simulations, as it was assumed that the high percentage of RSRZ outliers was caused by errors in the electron density map that was produced rather than incorrect residue positions. This, coupled with the better values for experimental data and other wwPDB validation data values and the fact that it contained the correct ligand, made the 6W4H structure more desirable than 8RVB.

4.2.3.2 MD Simulations

4.2.3.2.1 Determining optimal parameters for MD simulations of the complex

Several considerations guided the initial selection of parameters for simulating the NSP10-NSP16 complex. First, parameters from previous MD simulations of this complex were gathered and analysed to identify viable options. The choice of parameters was further refined based on the requirement to use LEaP for metal parameterisation and the GROMACS software for running the simulations. To evaluate the impact of these parameter combinations, two runs for each potential configuration were conducted, each lasting 50 ns. These test simulations were assessed to identify parameter sets that produced optimal trajectories, such as containing minimal fluctuations and convergence to a single conformation.

The main combinations tested were the ff14SB and ff19SB force fields with either the C-rescale or Parrinello-Rahman barostats. The resulting RMSD distributions of these MD simulation runs are shown in Figure 4.3. For the systems using V-rescale, C-rescale and ff14SB, both Run 1 and 2 resulted in similar ranges of RMSD values, although Run 2 showed two distinct peaks compared to Run 1. However, neither of these runs resulted in an RMSD distribution with one prominent peak but rather multiple smaller peaks, indicating the presence of multiple conformations during the simulation. Similarly, two main conformations characterised by two main peaks were seen in the violin plots for the two runs with ff19SB, V-rescale and Parrinello-Rahman systems.

In contrast, the Parrinello-Rahman barostat, in combination with ff14SB runs, displayed a higher variability in RMSD values across both runs. A bimodal distribution was seen for Run 1, suggesting the presence of multiple conformational states during the run. However, for Run 2, a unimodal distribution for the RMSD values was seen, indicating a single conformation for the complex was seen across the 50 ns trajectory. Similarly, Run 1 of the system using V-rescale, C-rescale and ff19SB was also unimodally distributed, while Run 2 was slightly less compact and unimodal. However, the violin plots for Run 1

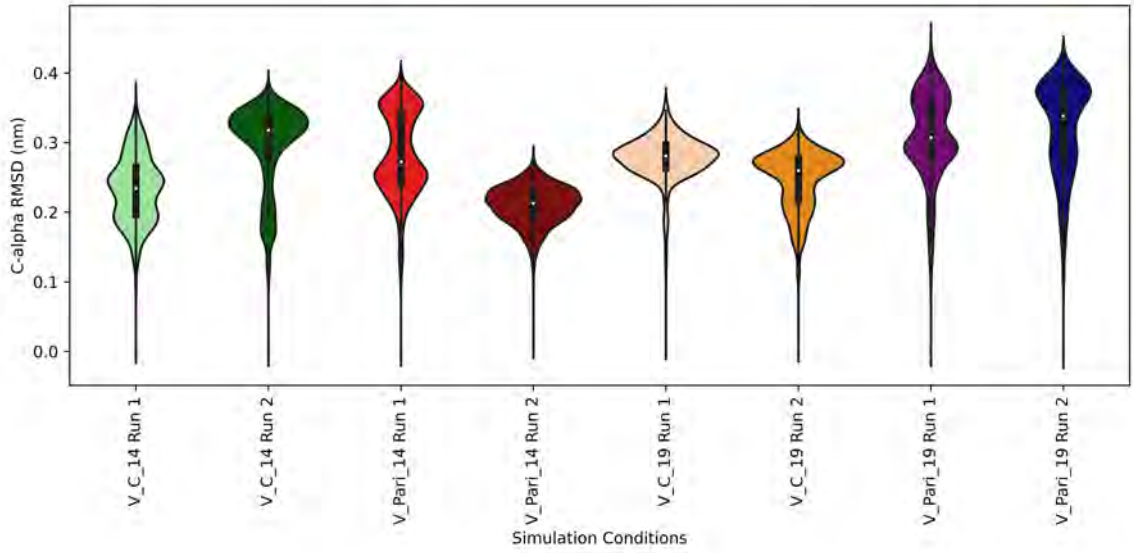

Figure 4.3: Violin plots of the C-alpha RMSD values over 50 ns for eight MD simulation runs of the SARS-CoV-2 NSP10-NSP16 complex. The simulations were conducted using combinations of two force fields (ff14SB and ff19SB), the V-rescale thermostat and two barostats (C-rescale and Parrinello-Rahman). The y-axis labels indicate the simulation conditions, where V denotes the V-rescale thermostat, C the C-rescale barostat, Pari the Parrinello-Rahman barostat, 14 the ff14SB force field, and 19 the ff19SB force field. Each combination includes two independent runs.

and 2 with V-rescale, C-rescale and ff19SB were the most similar in shape and suggested that the RMSD values were similar.

Based on the violin plots of the RMSD distributions of these systems, it was thought that the systems using ff19SB, V-rescale, and C-rescale resulted in the most stable trajectories over the 50 ns period. Therefore, it was decided that four runs of this combination would be run for at least 300 ns to determine if the runs remained stable throughout an extended simulation.

Additionally, it was discovered that the Parrinello-Rahman barostat was used erroneously both in the equilibration step and production run. This barostat has been described in the GROMACS manual as unstable during the equilibration step and should only be used during the production run (Abraham *et al.*, 2024). Other barostats, such as the Berendsen or C-rescale barostats, should be used instead during equilibration. Therefore, it was necessary to rerun the affected MD simulations where the C-rescale barostat was used in the equilibration step instead of the Parrinello-Rahman barostat.

A decision was made to run each new combination in triplicate for 50 ns. This means that there were three MD simulation runs for the combinations of V-rescale, C-rescale and ff14SB, V-rescale, C-rescale followed by Parrinello-Rahman and ff14SB, and finally V-rescale, C-rescale followed by Parrinello-Rahman and ff19SB. The final combination of V-rescale, C-rescale and ff19SB was run in quadruplicate. Figure 4.4 shows the line plot (A) and violin plot (B) of RMSD values for the combination runs using the ff14SB force field. Similarly, Figure 4.5 displays the corresponding line (A) and violin (B) plots for the ff19SB force field.

As seen in the line plots of Figure 4.4, RMSD values for Runs 1 and 2 of the system produced using V-rescale, C-rescale, and ff14SB continued to fluctuate, resulting in lines which did not plateau for the full 50 ns runs. Run 3 of the system using V-rescale, C-rescale, and Parrinello-Rahman with the ff14SB force field also did not plateau, with the RMSD value continuing to increase for the last 10 ns. This indicates that these systems had not yet equilibrated and continually moved between different conformations. Similarly, there is no single peak for the violin plot of Run 1 with only C-rescale and V-rescale in Figure 4.4, indicating that there was no single conformation for most of the run. It appeared that the RMSD value of the second run peaked at around 0.35 nm, indicating a conformation at around that RMSD value, but this was actually fluctuating between 0.3 and 0.4 nm, as seen in the line plot.

However, Run 3 using V-rescale and C-rescale only, and Runs 1 and 2 using Parrinello-Rahman for the production run with the ff14SB force field did appear to plateau at around 20 ns, 10 ns and 40 ns, respectively. Additionally, these runs appeared to settle into a conformation with peak RMSD values of 0.25, 0.24 and 0.28 nm, seen in the violin plot of Figure 4.4. However, none of these combinations resulted in equilibrated systems within 50 ns for all triplicate runs. In addition, none of these triplicates resulted in similar average RMSD values, as seen by the violin plots.

As seen in Figure 4.5, the RMSD values of Runs 1 and 3 did not appear to plateau when Parrinello-Rahman was used as the production step barostat in combination with the ff19SB force field. Large fluctuations were seen over the entire 50 ns runs, and the

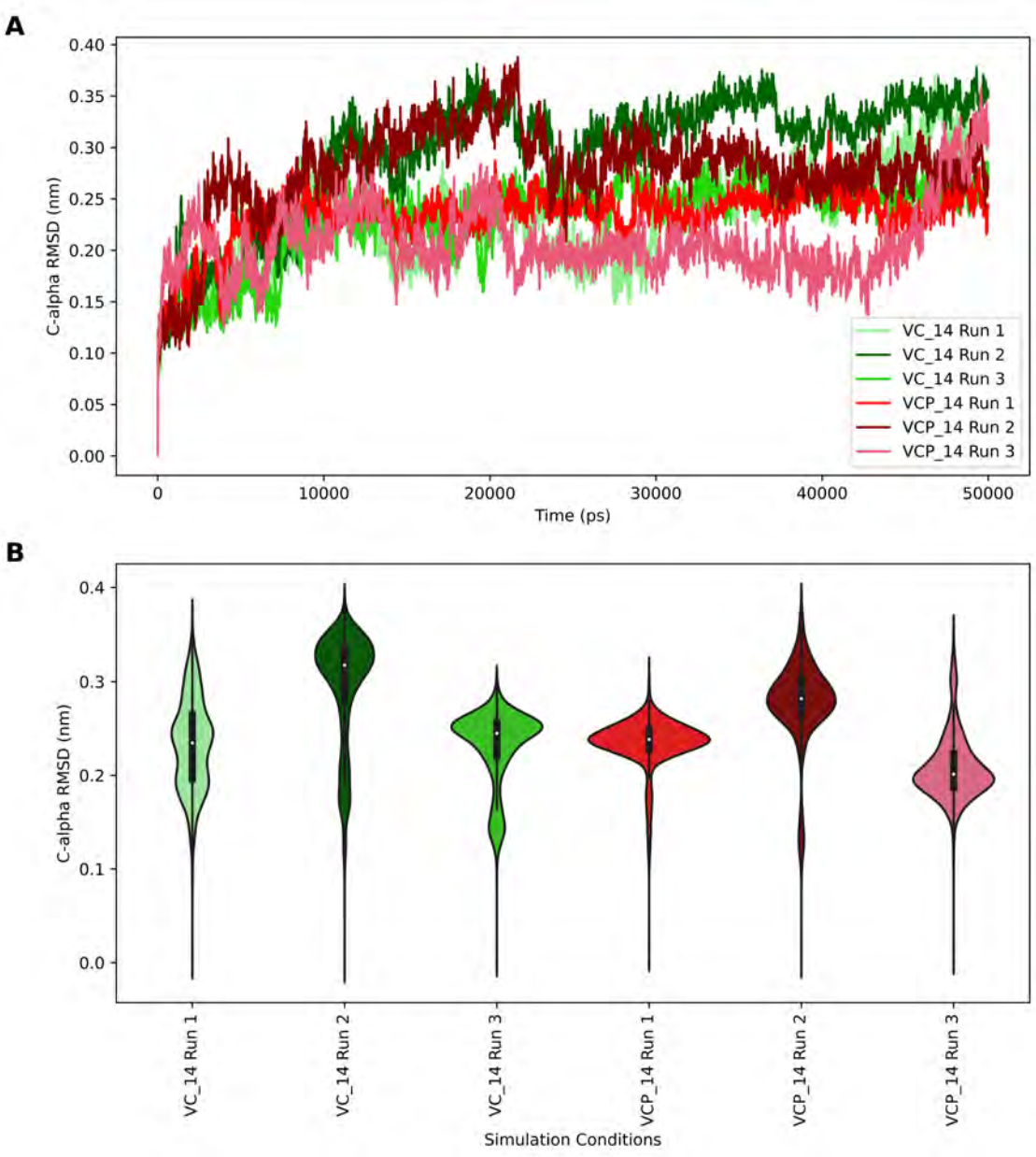

Figure 4.4: (A) Line plot and (B) violin plot of C-alpha RMSD values for six 50 ns simulation runs of the NSP10-NSP16 complex using the ff14SB force field. In three runs, the system was coupled with the V-rescale thermostat and C-rescale barostat (VC_14), while the remaining three runs had systems coupled with the V-rescale thermostat, the C-rescale barostat during equilibration, followed by the Parrinello-Rahman barostat in the production phase (VCP_14).

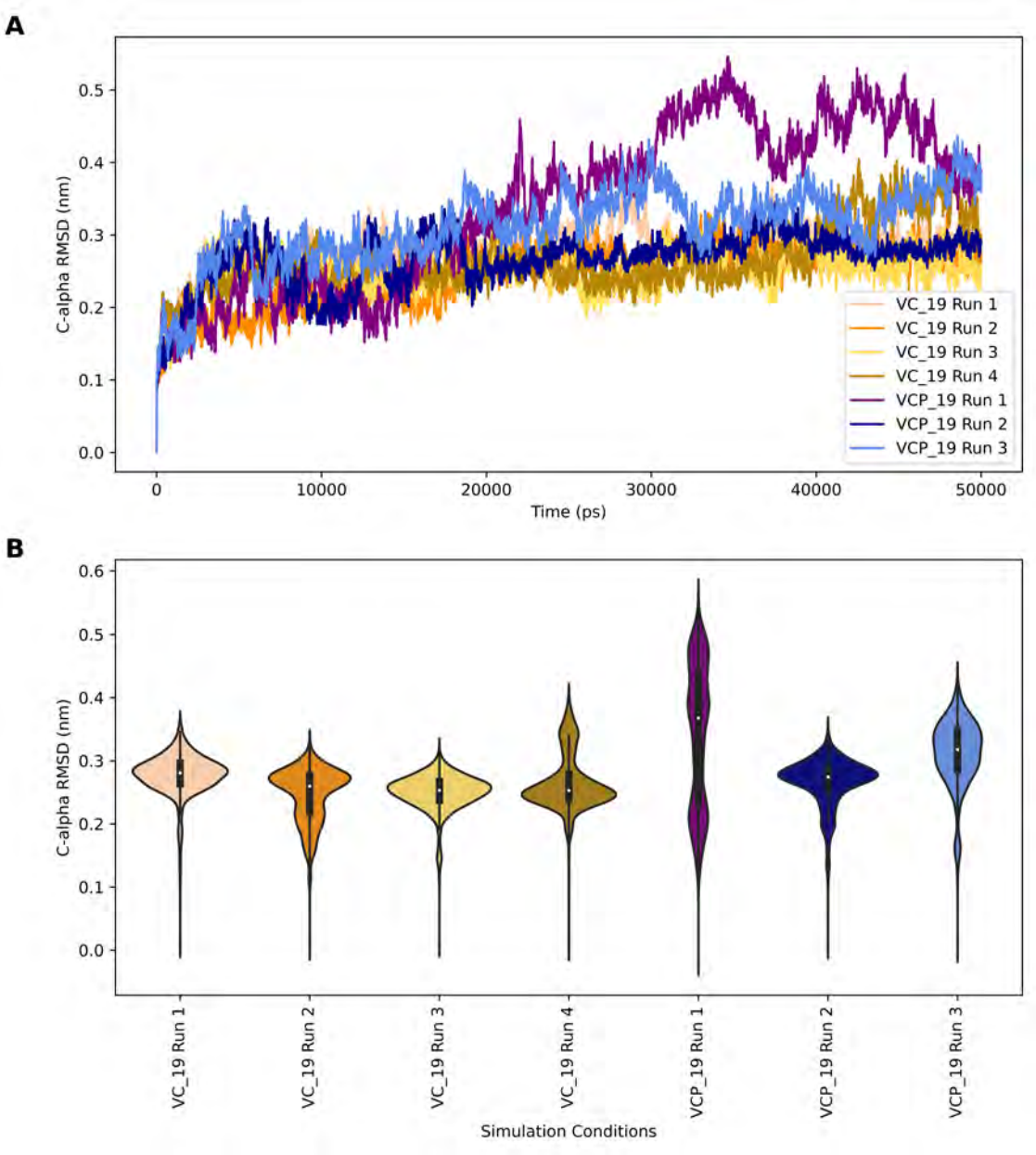

Figure 4.5: (A) Line plot and (B) violin plot of C-alpha RMSD values for seven 50 ns simulation runs of the NSP10-NSP16 complex using the ff19SB force field. In four runs, the system was coupled with the V-rescale thermostat and C-rescale barostat (VC_19), while the remaining three runs had systems coupled with the V-rescale thermostat, the C-rescale barostat during equilibration, followed by the Parrinello-Rahman barostat in the production phase (VCP_19).

violin plots in Figure 4.5 suggest that no stable conformation was reached, particularly for Run 1. However, Run 3 of this combination and Runs 1 to 3 using only V-rescale and C-rescale all plateaued within 20 ns with RMSD values between 0.25 and 0.3 nm. Run 4 with V-rescale and C-rescale appeared to plateau at an RMSD of around 0.2 nm between 20 ns and 40 ns, but the RMSD values increased to around 0.35 nm within the last 10 ns before beginning to reduce just before the end of the 50 ns run.

As seen in the violin plot of Figure 4.5, all four runs using only the C-rescale barostat had similar average RMSD values between 0.25 and 0.3. Each run also had a prominent peak within that range. Only Runs 2 and 4 showed a smaller peak around 0.2 and 0.35, respectively. The spread and ranges for the three runs using Parrinello-Rahman as the production barostat were much more variable, indicating that this combination was unsuccessful in producing stable and equilibrated systems within 50 ns.

As the combination of the ff19SB force field, V-rescale thermostat and C-rescale barostat appeared to create the most stable and equilibrated systems, these four runs were extended further to 500 ns. This was done to determine whether the fourth run would equilibrate again between 0.25 and 0.3 like the other runs and if the other runs would result in stable and equilibrated systems. The RMSD values for these extended runs are shown as line plots, as well as violin plots for the full 500 ns and last 300 ns in Figure 4.6.

The RMSD values fluctuated the least for Runs 1 and 3 using this combination, as seen in the line plot (A) of Figure 4.6. The RMSD values for these runs did not plateau but slowly increased from 0.3 to 0.35 nm between 100 and 500 ns. In addition, the fluctuations of the RMSD values were over 1 Å in some sections, suggesting that the system did not reach an equilibrated state. A single peak at around 0.35 was found for both these runs, with similar average RMSDs over the 500 ns runs, as seen in violin plot (B) of Figure 4.6. However, within the last 300 ns, Run 3 appeared to increase towards RMSDs of 0.4 nm, while Run 1 suddenly increased to over 0.5 nm at around 500 ns.

The remaining two runs of the system using the V-rescale thermostat and C-rescale barostat with the ff19SB force field also did not remain stable for the full 500 ns. Run 2 appeared to plateau with an RMSD value just below 0.4 between 150 ns and 450 ns, but

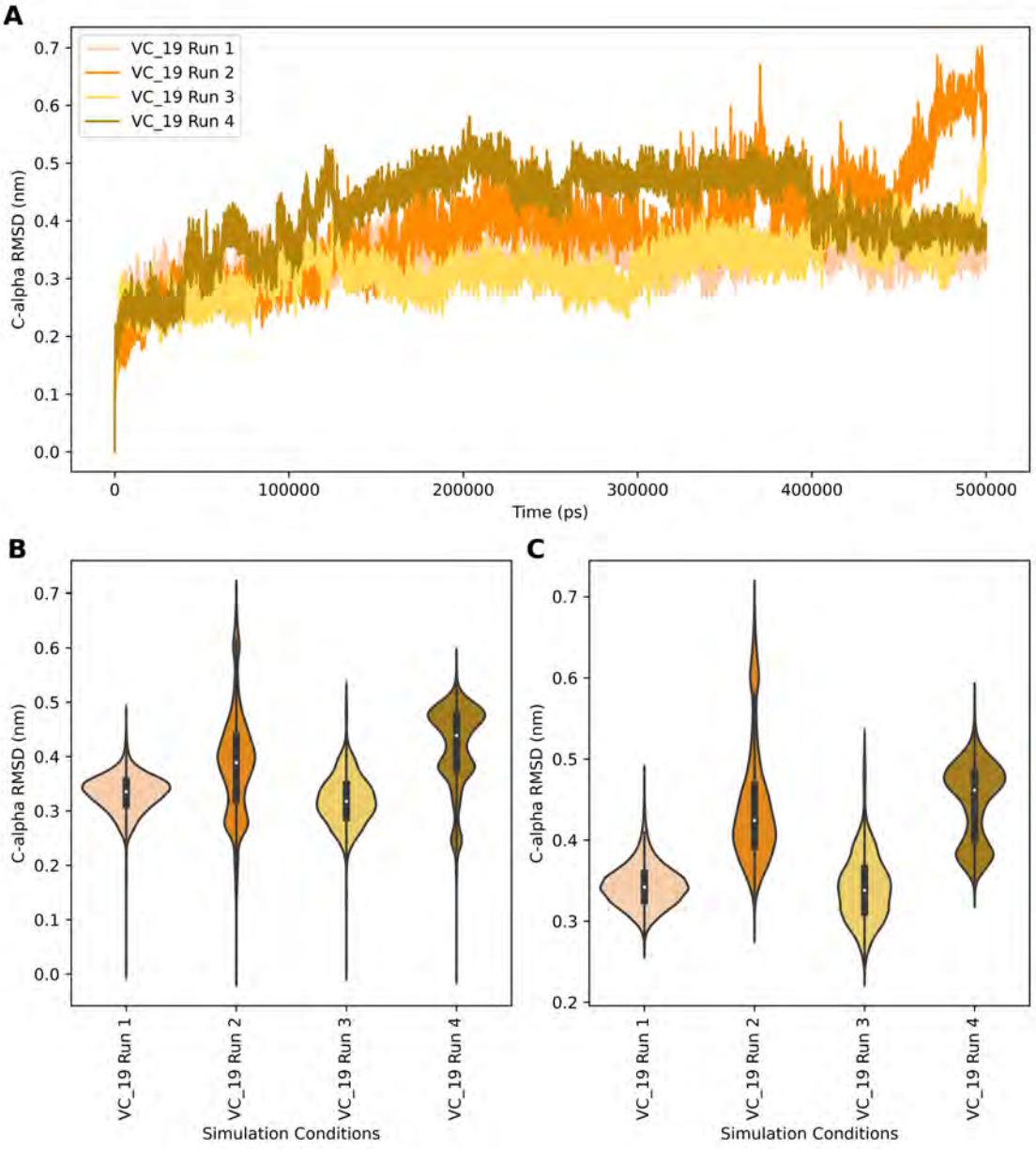

Figure 4.6: (A) Line plot of C-alpha RMSD values over time for four 500 ns simulations using the ff19SB force field with the V-rescale thermostat and C-rescale barostat (VC_-19). (B) Violin plot of RMSD distributions over the full 500 ns for the same simulations. (C) Violin plot of RMSD distributions for the last 300 ns of the same simulations.

the RMSD value suddenly increased dramatically to over 0.6 within the last 50 ns (seen in line plot (A) of Figure 4.6. The RMSD values continued fluctuating by more than 1 Å even during the plateau phase, indicating that the complex did not achieve a stable conformation and, therefore, the system was not equilibrated.

Run 4 appeared to alternate between two main structural conformations after 250 ns, with RMSD values centred around 0.38 and 0.48. This is evident from the two distinct peaks observed for Run 4 at these RMSD values in violin plot (C) of Figure 4.6, as well as from the line plot (A), which highlights the transitions between these two conformations. However, again, the system did not appear to equilibrate or stabilise.

This indicates that systems produced using this combination of thermostat, barostat and force field are unstable and cannot be used for further DRN analysis and mutation prediction. Further analysis was conducted to determine which parts of the system were unstable. To do this, NSP16 and NSP10 were analysed as separate entities. Line plots of the RMSF values per residue and the C-alpha RMSD line plots are shown in Figure 4.7 for NSP16 and Figure 4.8 for NSP10.

The NSP16 residues with the highest RMSF values were the last five residues on the C-terminal end of the chain. These were the only residues with RMSF values over 1 nm in the RMSF line plot (A) of Figure 4.7. These residues are found in a loop region of the protein, resulting in a very flexible tail of residues at the periphery of the protein. The residues on the N-terminal end had the next-highest RMSF values for all runs, with a maximum RMSF of 0.8 nm. Other residues of the NSP16 chain had RMSF values of around 0.7 nm for some runs, which indicates that some regions of this protein are also structurally fluctuating.

The line plot (B) in Figure 4.7 showing the RMSD values for only NSP16 residues are very similar in shape to those seen for the protein complex. Runs 1 and 3 are reasonably stable, as they plateau by 50 ns at an RMSD of just over 0.2 nm and rarely fluctuate over 1 Å. Run 2 slowly increased in RMSD value from around 0.2 to 0.3 until around 400 ns, then rapidly increased to over 0.5 within 100 ns. Run 4 plateaued at around 150 ns with

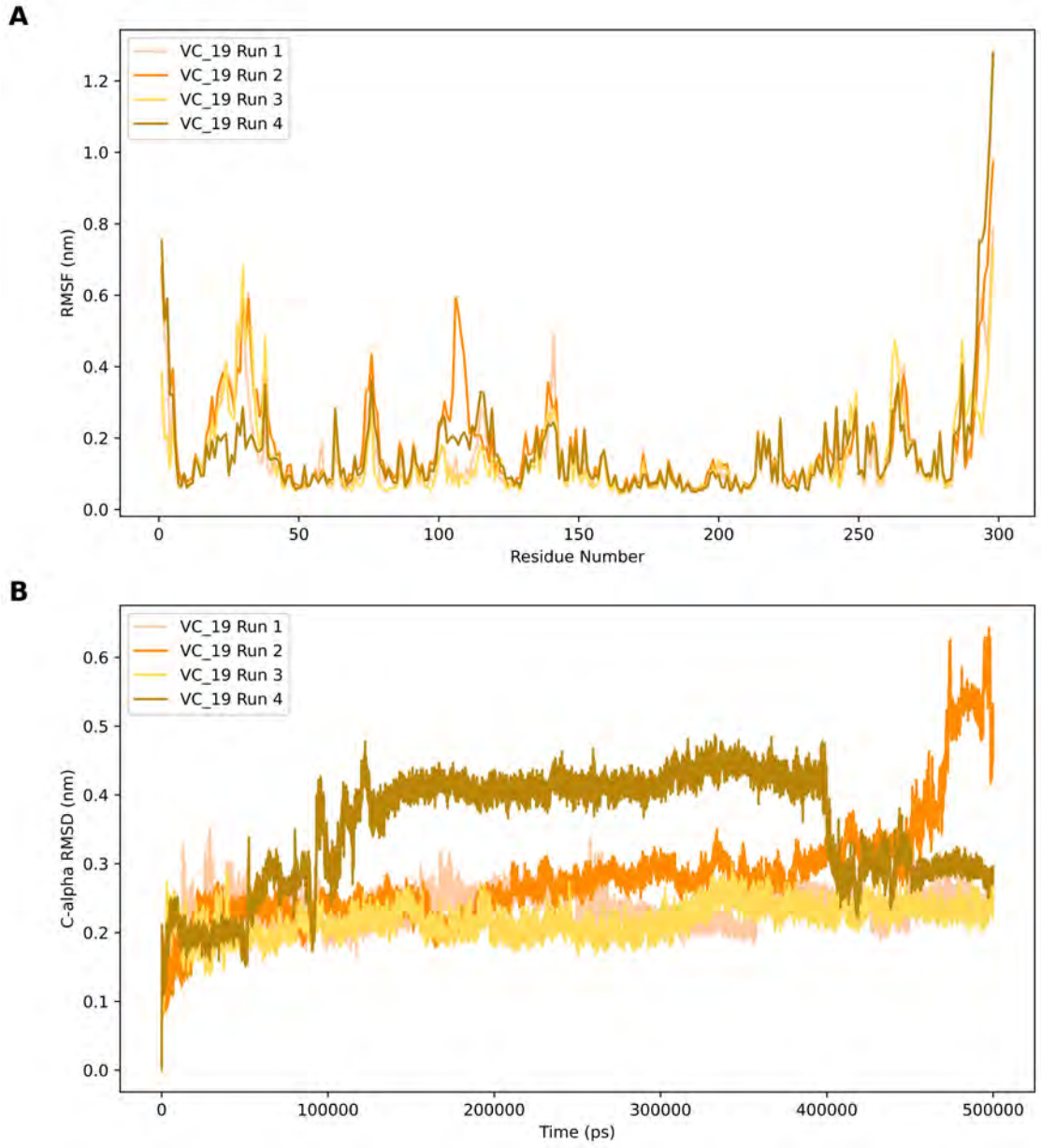

Figure 4.7: (A) Line plot of RMSF per NSP16 residue (1-298) for four 500 ns runs of the NSP10-NSP16 complex, using the ff19SB force field with the V-rescale thermostat and C-rescale barostat (VC_19). (B) Line plot of C-alpha RMSD values over the full 500 ns for the NSP16 residues of the same simulations.

an RMSD of 0.4 nm and appeared to reach a second conformation in the last 100 ns with an RMSD of around 0.3 nm.

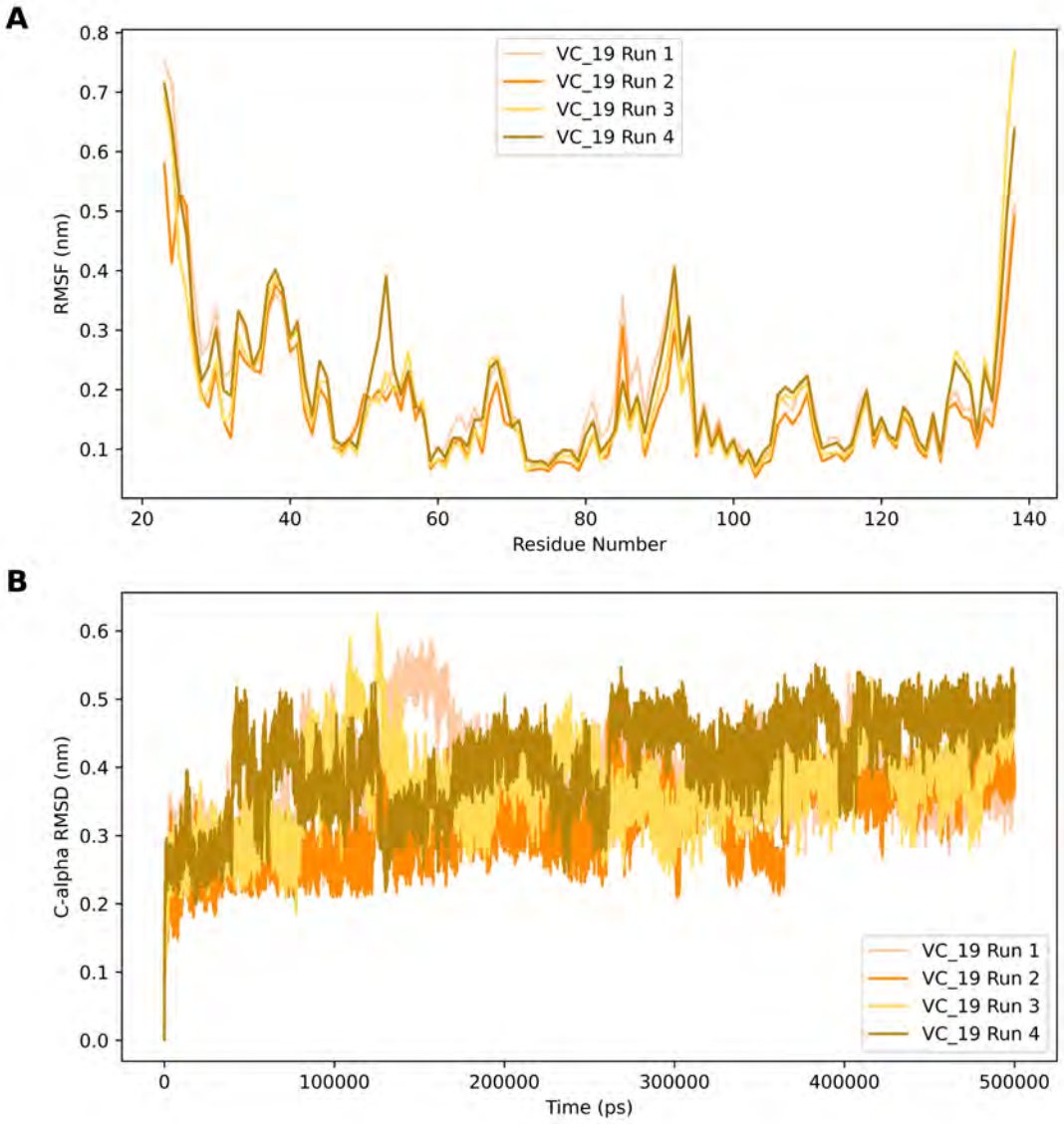

Figure 4.8: (A) Line plot of RMSF per NSP10 residue (18-133) for four 500 ns runs of the NSP10-NSP16 complex, using the ff19SB force field with the V-rescale thermostat and C-rescale barostat (VC_19). (B) Line plot of C-alpha RMSD values over the full 500 ns for the NSP16 residues of the same simulations.

None of the residues within the NSP10 chain reached an RMSF value over 1 nm, as seen in

the RMSF line plot (A) of Figure 4.8. The residues with the highest RMSF values for this chain were the first five simulated residues (18-22) of the N-terminal end of the protein and the last two residues of the C-terminal end of the protein. Again, these residues form part of the loop regions of NSP10, meaning that they are naturally more flexible and fluctuate compared to the rest of the protein.

However, the RMSD values for these runs are interesting. While all these runs appear to plateau before 150 ns, the fluctuations in RMSD values are very high throughout the 500 ns simulations, sometimes exceeding a difference of 2 Å.

Further consultation of the literature was performed at this stage due to the inability of any of the chosen combinations to result in stable, equilibrated systems for all quadruplicate runs. A discovery was made that within the AmberTools24 manual, the recommended water model for the ff19SB force field was not TIP3P as first thought, but the OPC water model. This model cannot be simulated in any version of GROMACS at this time, meaning it was no longer feasible to continue using the ff19SB force field without its recommended water model. Therefore, the decision was made to continue only using the ff14SB force field with the recommended water model of TIP3P.

From the 50 ns runs using the ff14SB force field, it appeared that the combination using C-rescale as the equilibration step barostat and Parrinello-Rahman as the production step barostat resulted in the more stable systems, with fewer conformations reached within the time frame. However, this combination did not result in similar RMSD values for the different runs, and they did not equilibrate quickly. A decision was made to proceed with this combination on a system containing the NSP10-NSP16 complex only, as well as a system containing the complex with a parameterised SAM ligand. This would provide the ability to compare the results and determine if the SAM ligand was necessary for stability.

4.2.3.3 Full Trajectory Analysis

Each system, both with and without SAM, was simulated in quadruplicate for 1 µs using the ff14SB force field, V-rescale thermostat, and C-rescale barostat during equilibration, followed by the Parrinello-Rahman barostat for the production run.

4.2.3.3.1 Zinc Parameterisation

The simulations were analysed to assess whether the zinc-binding site residues remained coordinated with the zinc ions throughout the trajectory. This coordination is essential for confirming the accuracy of the zinc-binding site parameterisation, ensuring that the binding sites were parameterised correctly. The distance between each coordinating residue and the first zinc ion is presented for the first production run of both systems, with and without SAM. In Figure 4.9, line plots (A) and (B) depict these distances for the system with and without SAM, respectively, while density plots (C) and (D) provide the corresponding distributions. Figure 4.10 was produced based on the distances between the coordinating residues and the second zinc ion within the NSP10 chain for the systems with SAM (line plot (A) and density plot (C)) and without SAM (line plot (B) and density plot (D)).

For Run 1 of the system containing the SAM ligand, the average distances between the sulfur atoms of residues C74, C77, H83, and C90 and the first zinc ion were found to be 2.296 Å, 2.305 Å, 2.107 Å, and 2.250 Å, respectively. These values correspond to the peaks observed in the density plots shown in line plot (C) of Figure 4.9. The average distances between these residue-ion bonds were almost identical for the first run of the system without the SAM ligand, which were 2.301 Å, 2.309 Å, 2.105 Åand 2.252 Å. Notably, the average distances for both runs are in close agreement with the typical zinc-sulfur bond distance of approximately 2.3 Å and zinc-nitrogen bond distance of approximately 2.1 Å found in zinc-coordinating sites. This supports the accuracy of the parameterisation for this first zinc binding site.

The average distance between the first zinc ions and the coordinating residues fluctuated

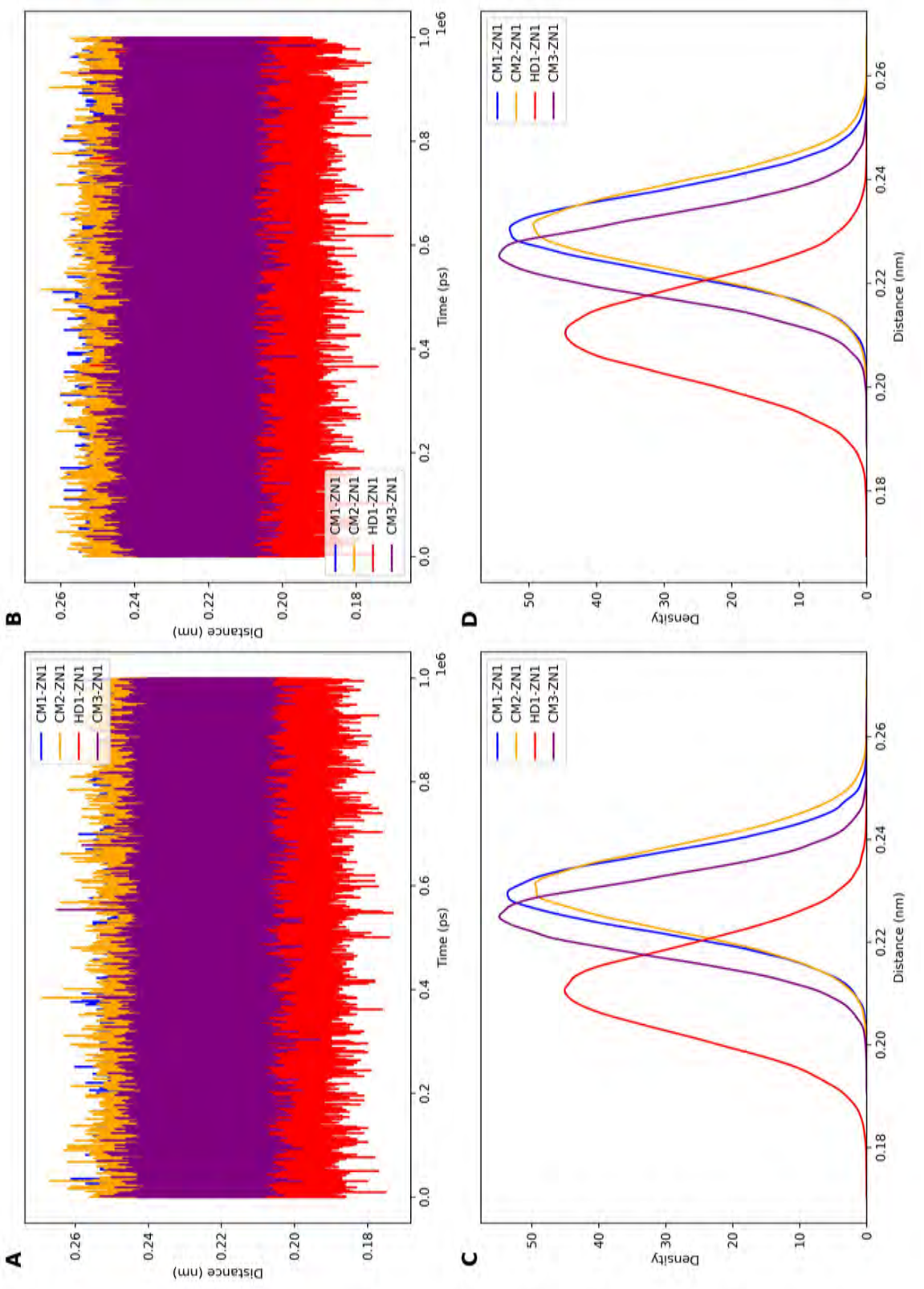

Figure 4.9: (A) Line plot and (C) density plot of distances between the first Zn atom and the S atoms of CM1–CM3 and the N atom of HD1 in the NSP10-NSP16 complex with SAM. Distances are taken from the first 1000 ns MD run using ff14SB with the V-rescale thermostat, C-rescale barostat during equilibration, and Parrinello-Rahman barostat during the production phase. (B) Line plot and (D) density plot of the same distances for the NSP10-NSP16 complex without SAM under identical conditions.

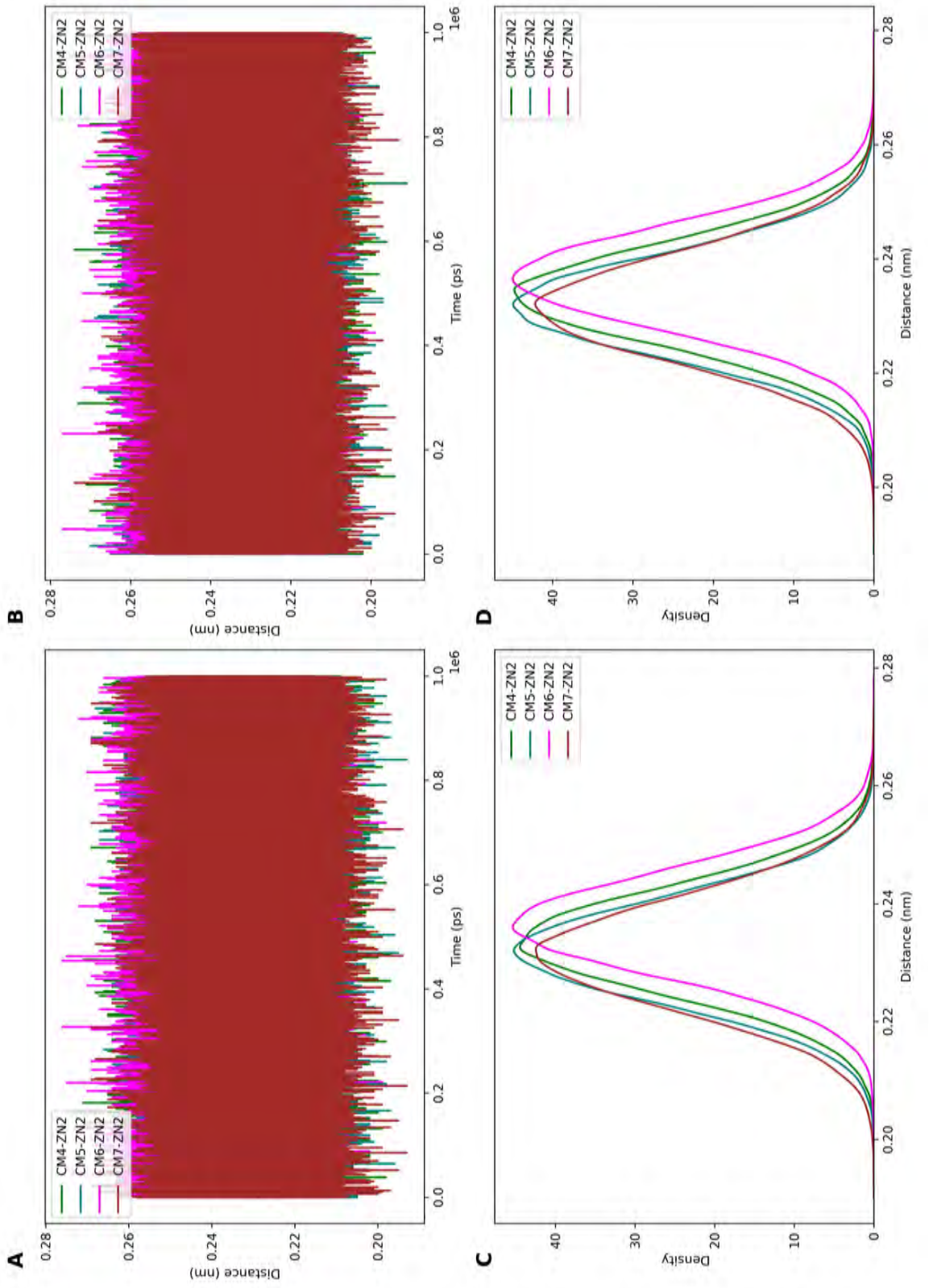

Figure 4.10: (A) Line plot and (C) density plot of the distances between the second Zn atom and the S atoms of CM4-CM7 in the NSP10-NSP16 complex with SAM. Distances are taken from the first 1000 ns MD run using ff14SB with the V-rescale thermostat, C-rescale barostat during equilibration, and Parrinello-Rahman barostat during the production phase. (B) Line plot and (D) density plot of the same distances for the NSP10-NSP16 complex without SAM under identical conditions.

within a range of 0.6 Å throughout the entire 1 µs simulation for both systems, as seen in line plots (A) and (B) of Figure 4.9. Importantly, these fluctuations were consistent over time and did not show a trend of decreasing or increasing, indicating stable coordination between the zinc ion and the residues. This suggests that the system reached an equilibrium state with no significant drift in the zinc coordination geometry during the simulation period.

The second zinc-binding site was also examined. For the first simulation run of the system which included the SAM ligand (line plot (A) and density plot (C) of Figure 4.10), the average distances between the sulfur atoms of the C117, C120, C128, and C130 residues and the second zinc ion were 2.337 Å, 2.321 Å, 2.366 Å, and 2.315 Å, respectively. Similarly, for the first run of the system without the SAM ligand (represented by the line plot (B) and density plot (D) in Figure 4.10), the average distances were nearly identical, with distances of 2.338 Å, 2.319 Å, 2.367 Å, and 2.315 Å, respectively.

As observed for the first zinc-binding site, these average distances align closely with the expected zinc-sulfur bond distances. Furthermore, consistent fluctuations of approximately 0.6 Å were observed throughout the simulations, demonstrating stable and reliable parameterisation of the zinc coordination sites in both systems.

4.2.3.3.2 Trajectory Analysis of the Full NSP10-NSP16 Complex

An analysis of the RMSD values of the quadruplicate runs of each system was performed after the zinc-binding site parameterisation checks. The analysis was performed, and the results are shown as line plots and violin plots for the full 1000 ns run, and violin plots for the last 300 ns in Figure 4.11 for the system with SAM and Figure 4.12 for the system without SAM.

The line plot (A) of the simulation runs shown in Figure 4.11 indicate that the systems with SAM have largely equilibrated, as each RMSD curve for all four runs plateaus after 300 ns. During this stable phase, the RMSD values fluctuate within a range of 0.25 nm to 0.4 nm. Specifically, for Runs 1 and 4, the fluctuations are limited to within 1 Å, particularly in the final 300 ns (as seen in the last 300ns violin plot of Figure 4.11), suggesting these runs are reasonably equilibrated and stable. However, in Runs 2 and 3, the fluctuations remain between 1 and 2 Å, even during the last 300 ns of the simulations. This indicates that these runs are not fully equilibrated and somewhat less stable for the NSP10-NSP16 complex.

All four simulation runs of the system without the SAM ligand showed a rapid increase in RMSD, reaching values between 0.25 and 0.35 nm within the first 50 ns (line plot (A) in Figure 4.12). However, the RMSD continued to rise gradually throughout the microsecond simulation instead of plateauing, indicating that these runs did not fully equilibrate. In Run 4, a sudden drop in RMSD from 0.3 to 0.175 nm was observed just before the 800 ns mark, followed by a return to over 0.3 nm by 800 ns, suggesting a brief structural rearrangement before stabilisation. While RMSD fluctuations were generally within 1 Å throughout the simulation, the overall change in RMSD exceeded 1 Å due to the gradual increase across all runs over the entire microsecond duration. This trend is also visible in the violin plot (B) of the full simulations in Figure 4.12, where all runs except Run 3 show a range exceeding 1 Å. Notably, all runs except Run 4 exhibited fluctuations within 1 Å and a single peak, indicating a stable conformation, during the last 300 ns of the simulation, as seen by the violin plot (C) in Figure 4.12.

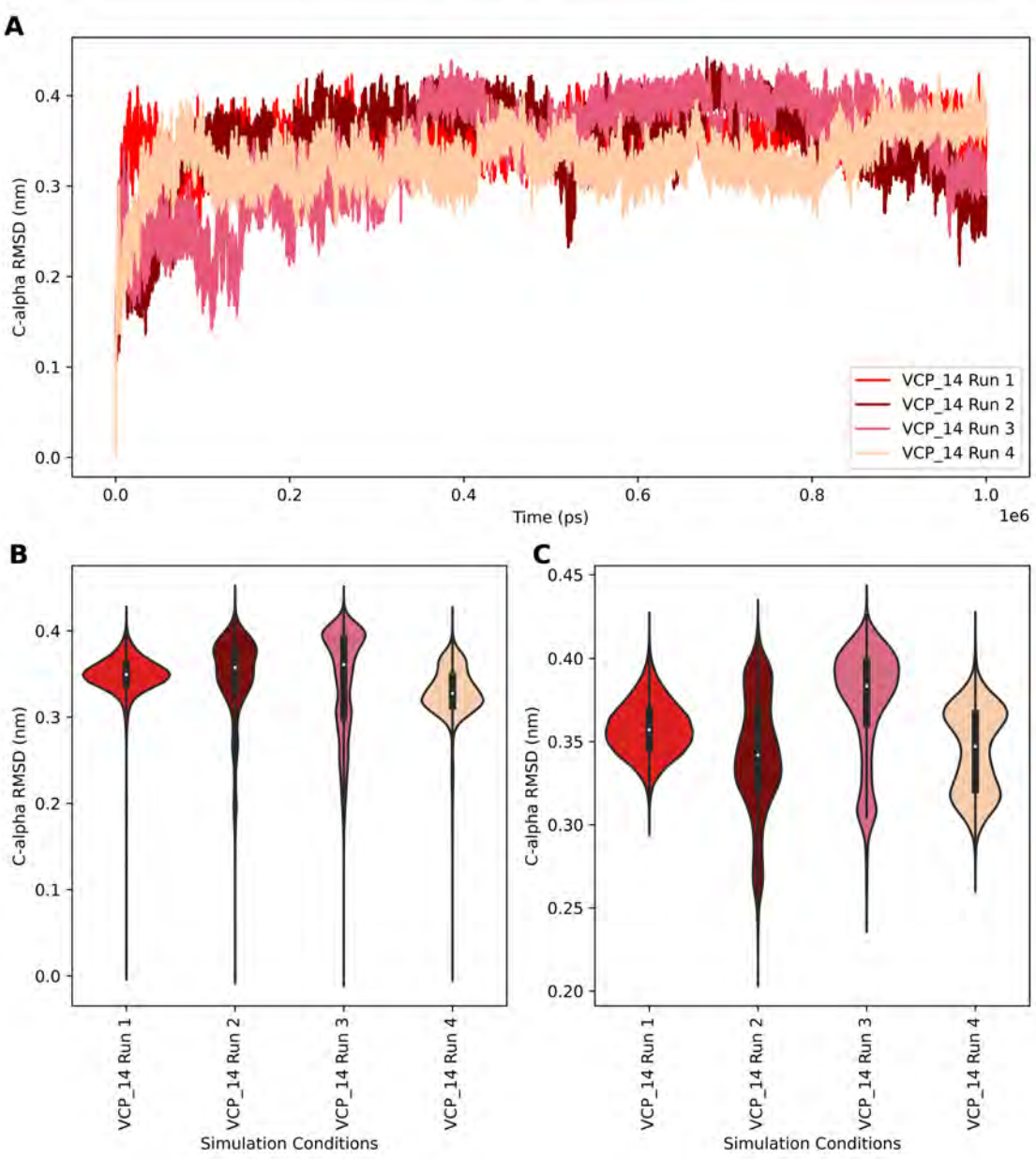

Figure 4.11: (A) Line plot of C-alpha RMSD values over time for four 1000 ns runs of the NSP10-NSP16 complex with SAM using the ff14SB force field. Systems were coupled with the V-rescale thermostat, the C-rescale barostat during equilibration, and the Parrinello-Rahman barostat in the production phase (VCP_14). (B) Violin plot of RMSD distributions over the full 1000 ns for the same simulations. (C) Violin plot of RMSD distributions for the last 300 ns of the same simulations.

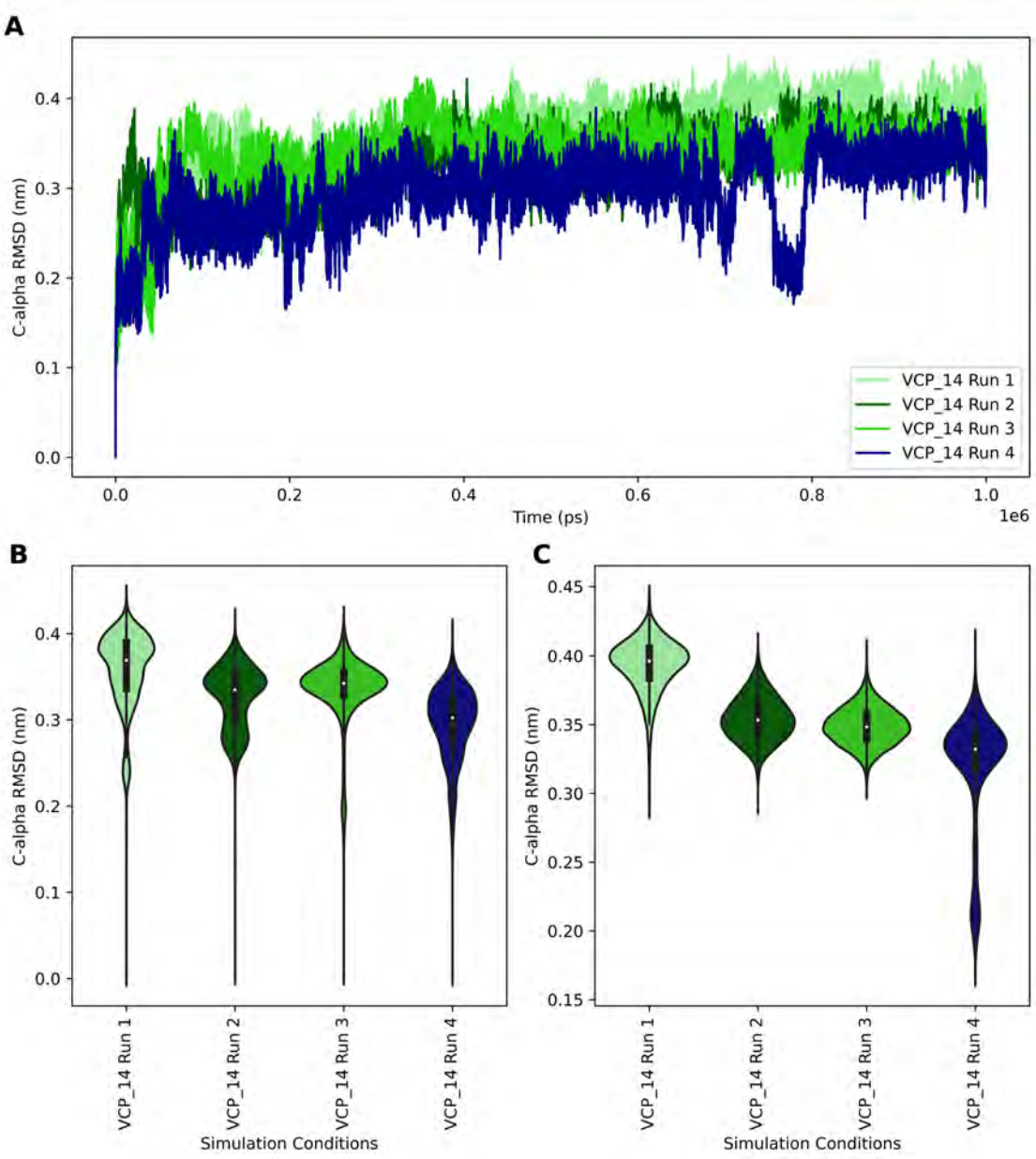

Figure 4.12: (A) Line plot of C-alpha RMSD values over time for four 1000 ns runs of the NSP10-NSP16 complex without SAM using the ff14SB force field. Systems were coupled with the V-rescale thermostat, the C-rescale barostat during equilibration, and the Parrinello-Rahman barostat in the production phase (VCP_14). (B) Violin plot of RMSD distributions over the full 1000 ns for the same simulations. (C) Violin plot of RMSD distributions for the last 300 ns of the same simulations.

4.2.3.3.3 Trajectory Analysis of the Separated NSP10 and NSP16 Chains

The RMSF values for each residue simulated for the NSP10 and NSP16 chains were calculated and plotted for further analysis. This allowed for analysis into the flexibility of each chain to be conducted, such as identifying the more rigid and flexible regions of the protein and comparing the variability of the structure between the different runs. The RMSF values for each residue in NSP16 are plotted in Figure 4.13 for the systems simulated with and without SAM. Similarly, Figure 4.14 represents the line plots of the RMSF values for each NSP10 residue in the systems simulated with SAM and without SAM.

When comparing the RMSF values of the NSP16 residues for both the systems with and without SAM in Figure 4.13, it was observed that the residues with the highest RMSF values were consistent across both systems. These residues, located at the C-terminal end of the NSP16 protein, show RMSF values ranging from 0.4 to 1.2 nm. The high RMSF values for these residues are attributed to their role in forming a flexible tail. As a result, these five residues were excluded from the RMSD calculations for the individual NSP16 protein. Additionally, it was noted that most RMSF values were similar, not only between the runs of the same system but also across the two systems.

Similarly to NSP16, the RMSF values for each of the NSP10 residues simulated were very similar over all the runs of both systems with and without SAM, as seen in Figure 4.14. In particular, the residues with the highest RMSF values are the first five residues of this chain on the N-terminal end. These residues, again, are not part of a helix or beta-sheet, meaning that they are part of the loop region and form a flexible tail. It was decided to exclude these residues from the RMSD calculations for the NSP10 protein.

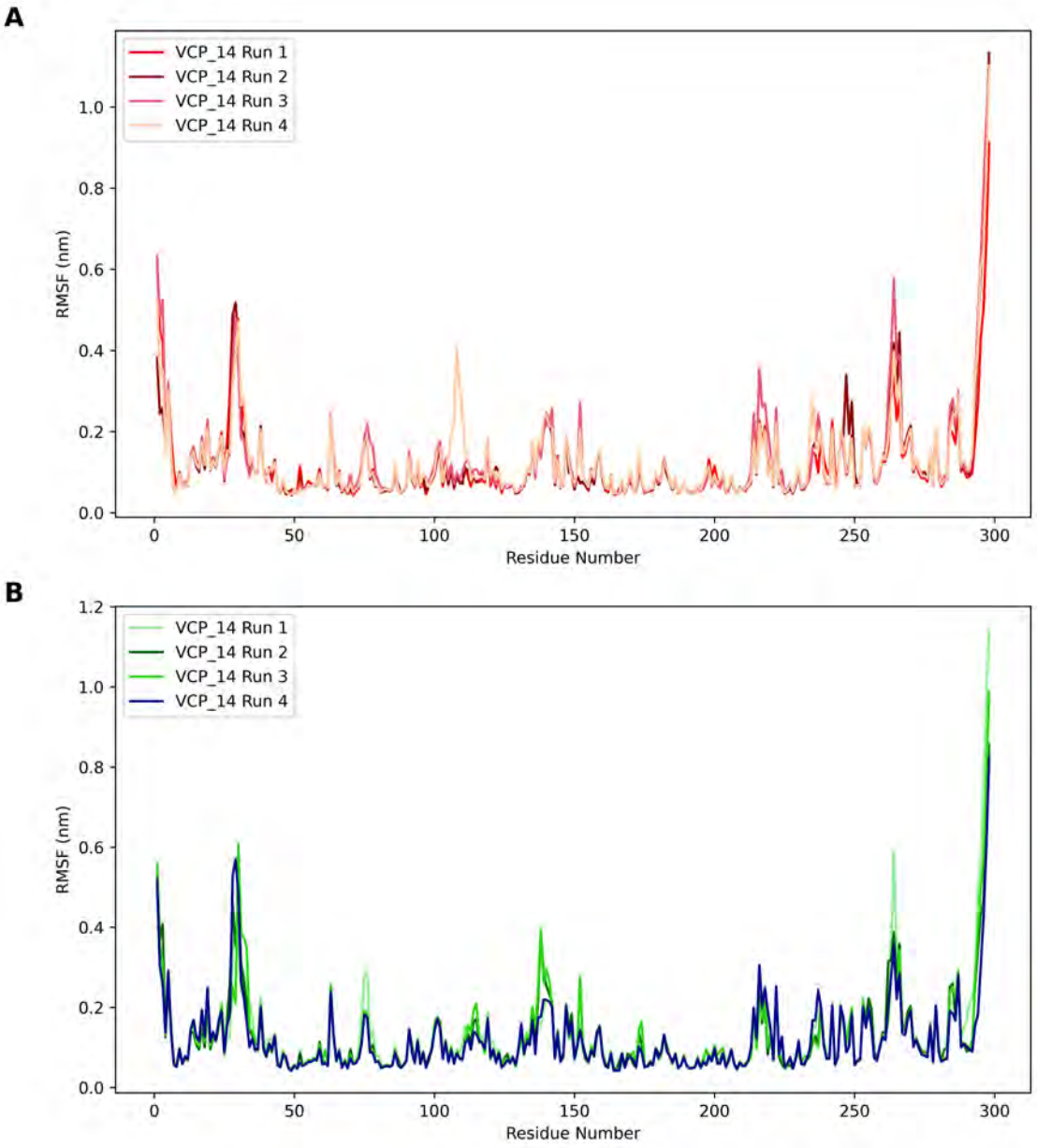

Figure 4.13: (A, B) Line plots of RMSF per NSP16 residue (1–298) for four 1000 ns runs of the NSP10-NSP16 complex, simulated (A) with SAM and (B) without SAM using the ff14SB force field. The systems were coupled with the V-rescale thermostat, C-rescale barostat during equilibration, and Parrinello-Rahman barostat during the production phase (VCP_19).

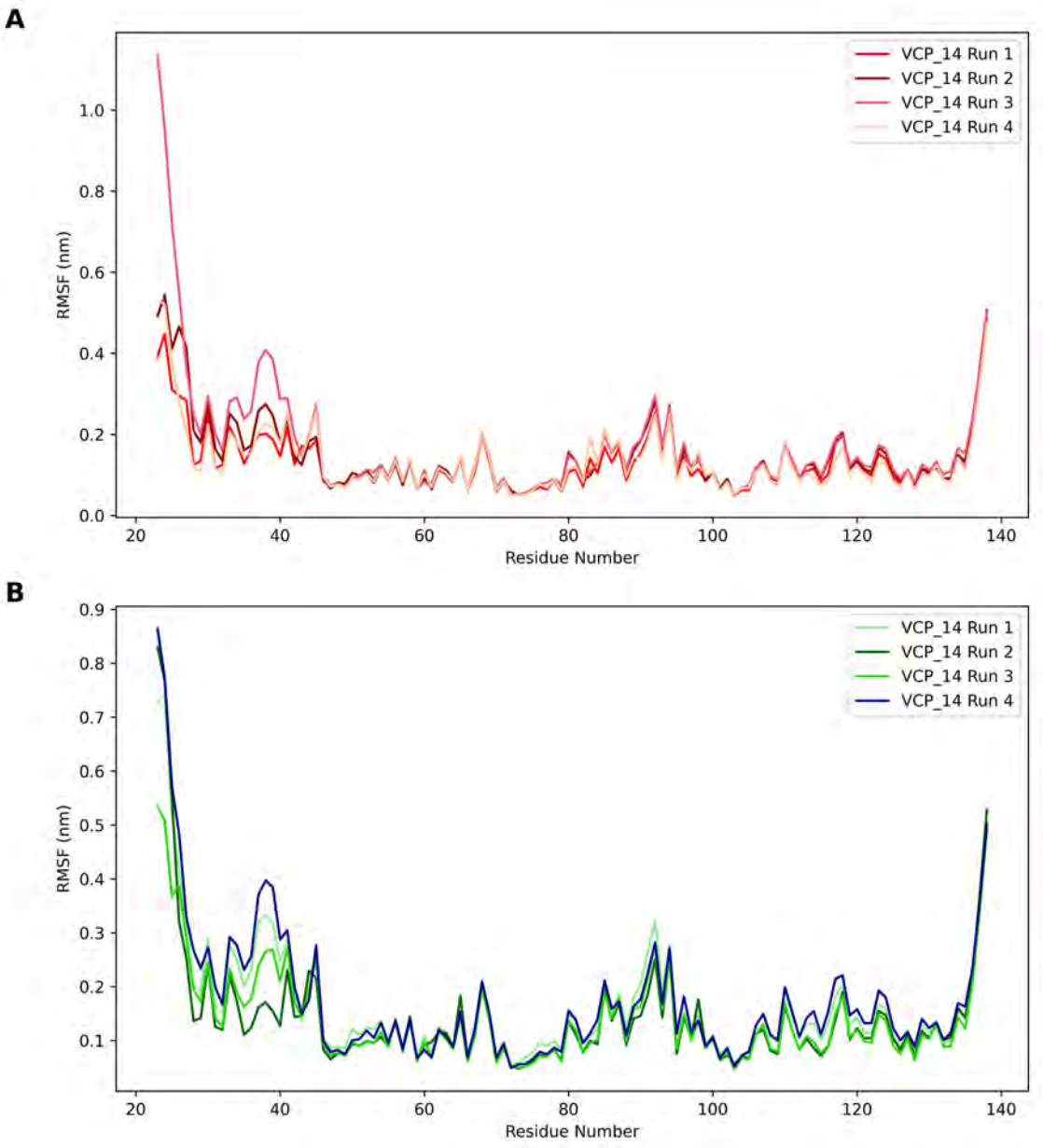

Figure 4.14: (A, B) Line plots of RMSF per NSP10 residue (18–133) for four 1000 ns runs of the NSP10-NSP16 complex, simulated (A) with SAM and (B) without SAM using the ff14SB force field. The systems were coupled with the V-rescale thermostat, C-rescale barostat during equilibration, and Parrinello-Rahman barostat during the production phase (VCP_19).

4.2.3.3.4 Trajectory Analysis of the NSP10 and NSP16 Subsections

The RMSD values for the separated NSP10 and NSP16 chains were calculated based on a specific selection of residues. This decision allows for the analysis of the structural dynamics of NSP10 and NSP16 while minimising the noise from highly flexible termini. For NSP16, residues 1–293 were included, excluding the last five tail residues, while for NSP10, residues 23–133 were selected, leaving out the first five simulated residues. The RMSD values are visualized using line, density, and violin plots for NSP16 (Figure 4.15) and NSP10 (Figure 4.16).

For both systems with and without SAM, the RMSD values for the selected NSP16 residues are much lower for all runs than for the full complex, as seen in Figure 4.15. For both systems, none of the runs reach RMSD values over 0.2 Å for the full microsecond simulations. In addition, the runs with SAM appear to plateau within 200 ns of the microsecond run, and the fluctuations of the RMSD values are all within 1 Å. However, the average RMSD values for each of these runs are slightly different, ranging from 0.13 nm to 0.17 nm, as seen in violin plot (B) of Figure 4.15. The production runs for the system without SAM show RMSD values that plateau around 300 ns, with overall fluctuations remaining within 1 Å. However, these fluctuations are slightly larger than those observed in the SAM-bound runs. The average RMSD values for each of these runs are between 0.14 and 0.16 nm, which is a smaller difference than between the runs with SAM.

The appearance of the line plots (A) and (B) for the NSP10 residues of both systems indicates that the RMSD values plateau very quickly, usually within 50 ns (Figure 4.16). However, the RMSD fluctuations over the full runs are both very high, with most of the runs experiencing rapid changes in RMSD values of over 2 Å. This indicates that the NSP10 chain may be quite flexible and that any small changes may cause significant deviations from the original structure. It appears that a single conformation of the NSP10 protein is reached for each run, characterised by the single, prominent RMSD peak in both violin plots (C) and (D) of Figure 4.16. The average RMSD values for the runs without SAM were very similar, at around 0.35 nm, while the runs for the system with SAM diverged slightly, ranging between 0.35 and 0.4 nm.

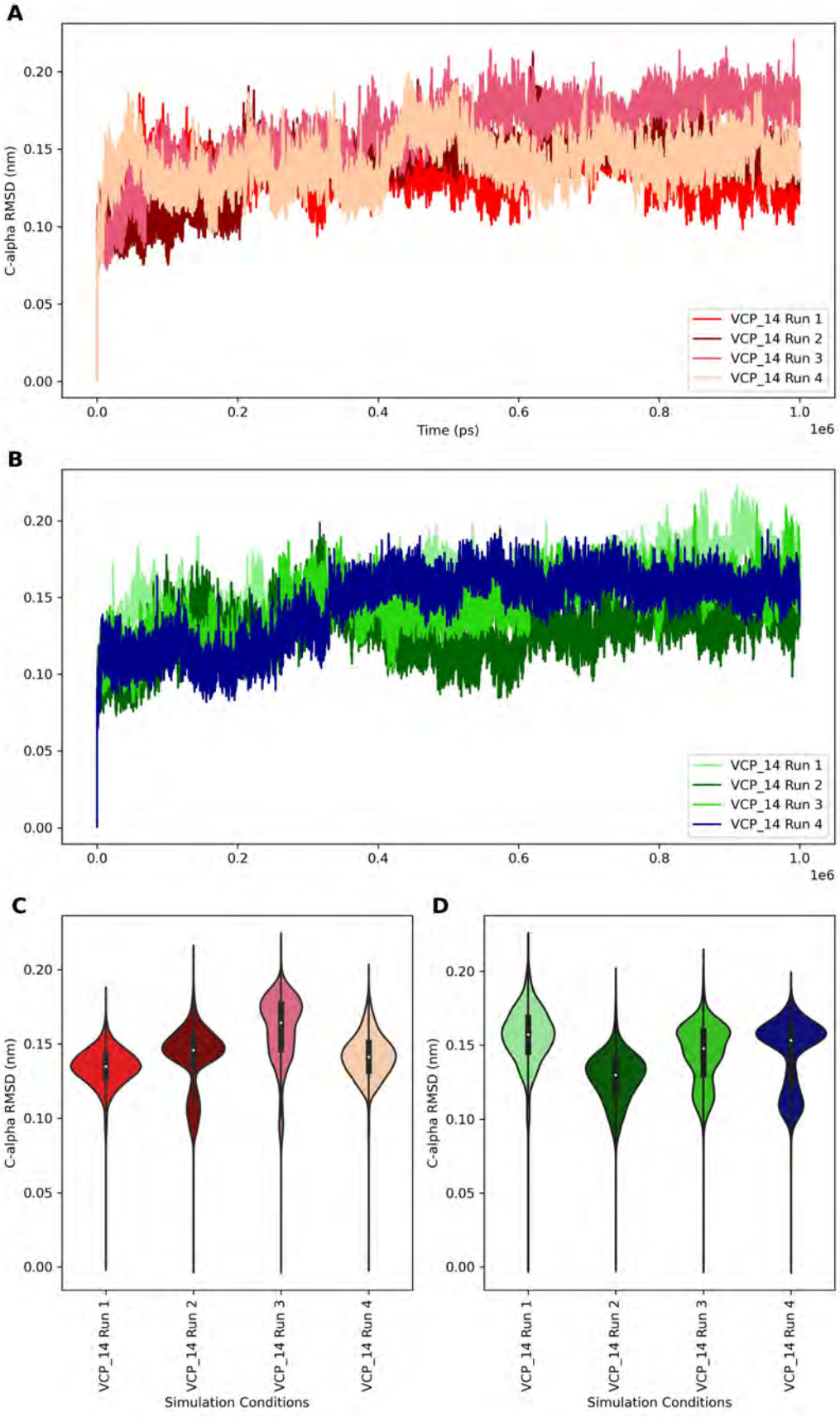

Figure 4.15: (A, B) Line plots of RMSD values for four 1000ns runs of the NSP16 protein residues 1-293 using ff14SB with V-rescale, C-rescale and Parrinello-Rahman (VCP_14), simulated (A) with SAM and (B) without SAM. (C, D) Violin plots of RMSD values for the four 1000 ns ff14SB MD simulation runs of NSP16 residues 1-293, simulated (C) with SAM and (D) without SAM.

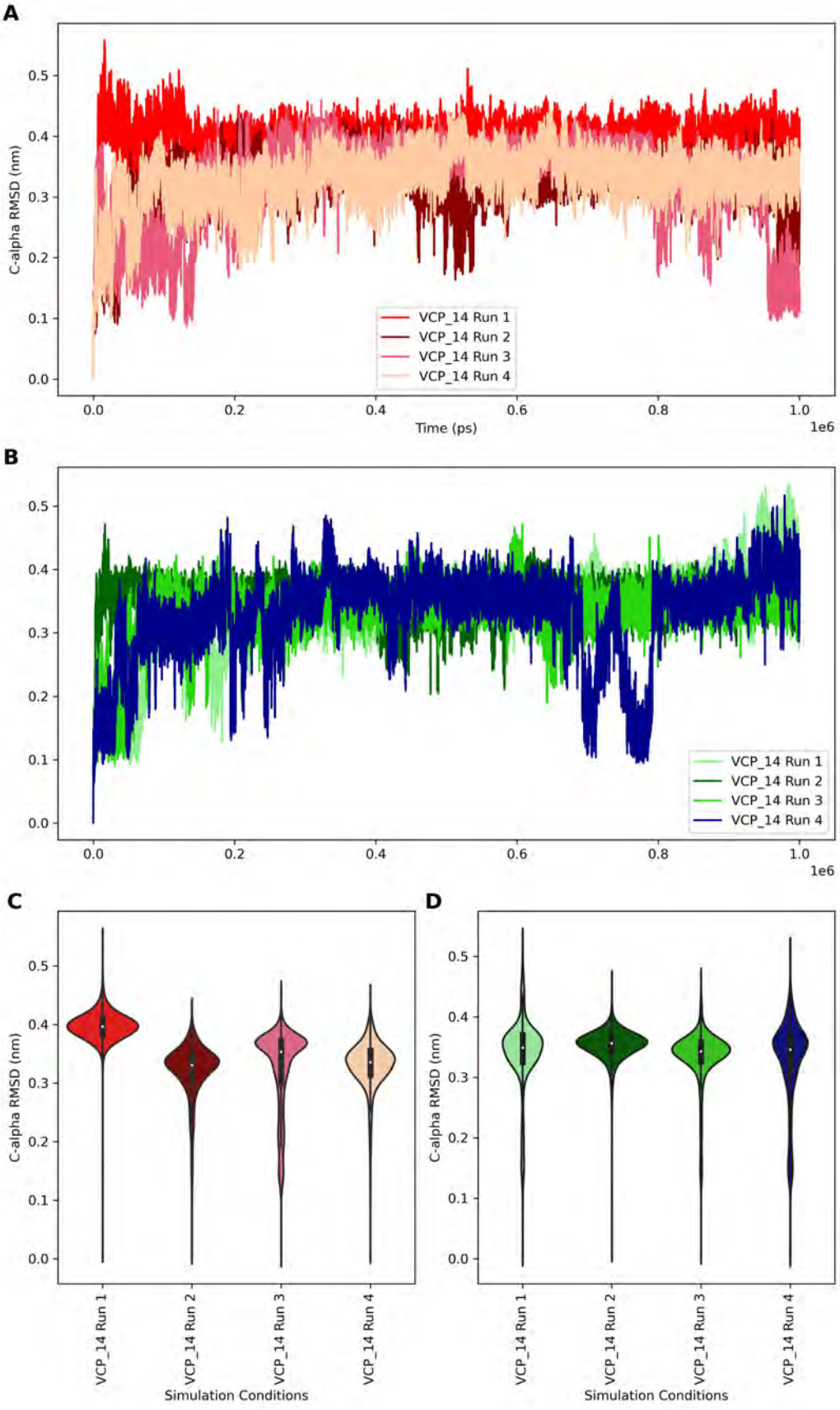

Figure 4.16: (A, B) Line plots of RMSD values for four 1000ns runs of the NSP10 protein residues 23-133 using ff14SB with V-rescale, C-rescale and Parrinello-Rahman (VCP_14), simulated (A) with SAM and (B) without SAM. (C, D) Violin plots of RMSD values for the four 1000 ns ff14SB MD simulation runs of NSP10 residues 23-133, simulated (C) with SAM and (D) without SAM.

4.2.4 Selection of MD Trajectory Data for Further Analysis

Four independent MD simulations were performed for the NSP10-NSP16 complex with the SAM ligand, along with four additional runs of the complex without SAM, each spanning 1 microsecond. Based on results from the initial short MD simulations, the ff14SB force field was chosen, with V-rescale used as the thermostat, C-rescale as the barostat during the equilibration step, and the Parrinello-Rahman barostat applied during the production runs. To assess system equilibration and protein stability, RMSD and RMSF values were analyzed for the full complex as well as for the individual NSP10 and NSP16 proteins.

Highly fluctuating tail residues were excluded to focus on the most structurally stable and functionally relevant regions, resulting in the selection of residues 1–293 for NSP16 and residues 23–133 for NSP10. This refined selection ensured that subsequent analyses, including DRN calculations, captured representative MD trends while minimizing noise from highly flexible termini. As neither system (with or without SAM) exhibited greater protein stability, both were chosen for further analysis and comparison of their structural and dynamic properties.

4.3 Part II: DRN Analysis of the NSP10-NSP16 Complex

4.3.1 Literature Review

4.3.1.1 Residue Interactions and DRN Analysis

A protein can be represented as a residue interaction network (RIN). Each node in the network represents a specific residue in the protein, specifically a $C\alpha$ atom if it is a glycine residue and a $C\beta$ atom for the remaining residues. Edges between nodes are produced when the distance between the two residue atoms is less than a cutoff value, the default of which is 6.7 Angstroms (Penkler *et al.*, 2018). However, RINs can only be used to describe static proteins, as the network is created from the distances between atoms at a single point in time. To describe dynamic proteins over a time period, a RIN is produced from each selected trajectory frame of an MD simulation based on a chosen time step (Brown et al., 2017). Network centrality metrics are then calculated for the nodes of each RIN and subsequently averaged over all selected MD trajectory frames, producing averaged centrality values for each residue. By averaging over all the frames, the natural fluctuations of the distance between the residues are reflected in the metrics. If the distance between two residues is not always within the cutoff, the centrality metrics of those residues will vary accordingly. The calculation of these averaged centrality metrics is called DRN analysis.

4.3.1.1.1 Average Centrality Metrics

The main DRN analysis average centrality measures calculated are *averaged betweenness centrality* (BC), *closeness centrality* (CC), *degree centrality* (DC), *eigencentrality* (EC), *farness* (L), *Katz centrality* (KC), *eccentricity* (ECC) and *Page Rank* (PR) (Sheik Amamuddy *et al.*, 2021b). The mathematical formulae used to calculate these metrics are found in Sheik Amamuddy *et al.* (2021b).

BC is a metric that describes the number of times a protein residue is found within the shortest path between other residue pairs (Sheik Amamuddy *et al.*, 2021a). A high *averaged BC* value indicates the residue is most likely one of the important communication hubs. These hubs connect various portions of the protein structure and control information flow between the protein residues (Okeke *et al.*, 2021).

CC measures the reachability of a node, describing the potential influence the node has on the other residues of the protein (Sheik Amamuddy *et al.*, 2021a). A high *CC* value indicates the residue has high reachability, which is often associated with protein domains of high functional importance, such as catalytic sites (Okeke *et al.*, 2021).

DC is a measure of the number of contacts a residue has (Sheik Amamuddy *et al.*, 2021a). A high *DC* value indicates that the residue is surrounded by many other residues and is buried within the protein. In contrast, a low *DC* value suggests that the residue is found near the periphery of the protein (Okeke *et al.*, 2021).

The *EC* value of a residue indicates the importance of that residue based on the number of residues it neighbours and the importance of those neighbouring residues (Sheik Amamuddy *et al.*, 2021a).

The *KC* metric is a generalised version of *EC*, which describes the relative degree of influence of a residue within the connected residues of a network. The main distinction between *KC* and *EC* is the addition of an adjacency dampening matrix in the formula of *KC* (Sheik Amamuddy *et al.*, 2021b). This matrix adds a weighting factor to the edges in the RIN. It adjusts how much influence the direct and indirect neighbours of a residue contribute to its centrality.

ECC represents the longest among the shortest paths from a given node to any other node within a network (Sheik Amamuddy *et al.*, 2021b). In the context of an RIN, *ECC* quantifies the relative distance of a residue to the most distant residue it can reach through the shortest possible connections.

4.3.1.1.2 Rationale of DRN Analysis for Residue Mutability

DRN analysis has been shown to highlight functionally important residues within a protein, as these tend to act as communication hubs in the structural network (Okeke *et al.*, 2021). Specifically, it has been found that residues with high averaged BC are closely associated with functional sites, suggesting a critical role in mediating allosteric communication or structural integrity. Similarly, high DC values indicate residues with numerous local interactions, often marking local structural or functional significance. As functionally important residues have previously been found to be less tolerant to mutation (McLaughlin Jr *et al.*, 2012), DRN metrics have been identified as a potential means of identifying mutation-resistant regions within proteins. Therefore, the eight DRN metrics were used as features when attempting to predict mutation frequency.

4.3.2 Methodology

4.3.2.1 DRN Analysis

DRN analysis was performed using the MDM-TASK toolkit (Brown *et al.*, 2017, Sheik Amamuddy *et al.*, 2021b) to study the interactions between residues within the separate NSP10 and NSP16 proteins. Each trajectory (with and without SAM) was course-grained using the GROMACS `gmx trjconv` command with a step size of 20, resulting in 5000 frames chosen from the 100 000 frames produced over 1 microsecond. Next, the trajectories of the complexes were split into the individual chains of NSP10 (residues 23-133) and NSP16 (residues 1-293), where the tail residues with the highest fluctuations (residues 18-22 for NSP10 and 294-298 for NSP16) were removed. The alpha-carbons of the glycine residues and the beta-carbons for the remaining residues were then extracted to create the final trajectories from which the RINs would be produced.

The calc_network.py tool from MDM-TASK (Sheik Amamuddy *et al.*, 2021b) was used to perform network calculations using the 5000 frame trajectories (.xtc) and corresponding topology files for each protein (.pdb) as inputs. A step size of 5 and Euclidean threshold

of 6.7 Å were applied during the calculations. The 8 centrality metrics (*BC*, *CC*, *DC*, *EC*, *ECC*, *Katz*, *L* and *PR*) for each RIN were calculated and subsequently averaged over all selected frames in the trajectory for each residue using MD-TASK. These values were then averaged over the four replicate trajectories of the complex simulated with or without SAM, resulting in two output .csv files.

To analyze the DRN metric distributions, residues were grouped into three bins, which are top, middle, and bottom. The top bin contained the residues with the highest third of DRN metric values, while the bottom bin contained the lowest third. For the *ECC* and *L* metrics, the binning was reversed, with the residues with the lowest values assigned to the top bin and the highest values to the bottom bin, due to the nature of the metric function.

This three-bin approach was chosen over alternative methods, such as percentile-based thresholds (e.g., top and bottom 10%), to ensure a balanced distribution of residues across categories while maintaining sufficient statistical power for analysis. It also prevents small sample sizes in extreme groups, reduces sensitivity to outliers, and allows for more robust comparisons. Additionally, this binning strategy was used in Barozi *et al.* (2024), enabling direct comparison with their results.

4.3.2.2 NSP10 and NSP16 Residue Mutation Frequencies

The mutation frequencies for each residue of the NSP10 and NSP16 proteins were calculated by retrieving all SARS-CoV-2 sequences for the Alpha, Beta, Gamma, Delta and Omicron variants from the GISAID database. These sequences were filtered, leaving only those with high coverage and patient status uploaded between 1st December 2019 and 24th February 2024, resulting in a total of 191 878 sequences. These remaining sequences were provided in batches to the GISAID CoVsurver tool to identify the single nucleotide variants (SNVs) and output to a .tsv file per batch. The existing variant SNVs for the NSP10 and NSP16 proteins were identified from the .tsv file using an in-house Python Jupyter notebook, where the mutation frequency for each residue was computed overall.

4.3.3 Results and Discussion

4.3.3.1 DRN Analysis

4.3.3.1.1 DRN Metrics Difference with and without SAM

DRN analysis was conducted on the selected residues of the NSP10 (23-133) and NSP16 (1-293) proteins isolated from the MD trajectories during their analysis. The DRN analysis was performed separately for the two proteins of the trajectories of both systems (simulated with and without SAM). This resulted in eight average DRN metric values per residue for each protein, which were further averaged over the four runs per system. Four datasets were formed, each containing the eight average DRN metric values per residue for each system of either the NSP10-NSP16 complex with SAM or without SAM.

The residues of each dataset were grouped into top, middle, and bottom bins based on their DRN metric values, as described in Section 4.3.2.1. The overlap of residues in the top and bottom bins between the SAM-bound and SAM-free datasets was statistically evaluated using the hypergeometric test. This was performed to determine whether a higher number of residues were shared between the top and bottom bins of these datasets compared to a random selection of residues. The null hypothesis is the proportion of residues in the top third of DRN metric values that overlap between the SAM-free and SAM-bound systems is equal to the proportion expected by random chance. In other words, this would suggest the observed overlap is consistent with the distribution of residues in the whole proteins. The alternate hypothesis is that the proportion of residues in the top third of DRN metric values that overlap between the SAM-free and SAM-bound systems significantly differs from the proportion expected by random chance. This can indicate either enrichment (more overlap than expected) or depletion (less overlap than expected). The results of these tests for the 293 NSP16 protein residues are shown in Table 4.3 and Table 4.4 for the 111 NSP10 protein residues.

Table 4.3: Enrichment fold and the enrichment/depletion p-values from hypergeometric tests of the residues in the top and bottom third of the DRN metrics of NSP16 with SAM compared to without SAM.

Averaged DRN Metric	**No. of Same Mutated Residues (Top Bin)**	**Enrichment Fold**	**P-value**		**No. of Same Mutated Residues (Bottom Bin)**	**Enrichment Fold**	**P-value**	
Average BC	94	2.87	3.44×10^{-66}	***	92	2.86	4.81×10^{-63}	***
Average CC	96	2.93	1.46×10^{-72}	***	93	2.90	6.73×10^{-66}	***
Average DC	92	2.81	1.21×10^{-60}	***	93	2.90	6.73×10^{-66}	***
Average EC	92	2.81	1.21×10^{-60}	***	94	2.93	5.93×10^{-69}	***
Average ECC	93	2.84	2.47×10^{-63}	***	93	2.90	6.73×10^{-66}	***
Average Katz	95	2.90	3.01×10^{-69}	***	93	2.90	6.73×10^{-66}	***
Average L	96	2.93	1.46×10^{-72}	***	93	2.90	6.73×10^{-66}	***
Average PR	94	2.87	3.44×10^{-66}	***	94	2.93	5.93×10^{-69}	***

Table 4.4: Enrichment fold and the enrichment/depletion p-values from hypergeometric tests of the residues in the top and bottom third of the DRN metrics of NSP10 with SAM compared to without SAM.

Averaged DRN Metric	**No. of Same Mutated Residues (Top Bin)**	**Enrichment Fold**	**P-value**		**No. of Same Mutated Residues (Bottom Bin)**	**Enrichment Fold**	**P-value**	
Average BC	36	2.92	7.07×10^{-27}	***	36	2.92	7.07×10^{-27}	***
Average CC	36	2.92	7.07×10^{-27}	***	37	3.00	2.58×10^{-30}	***
Average DC	36	2.92	7.07×10^{-27}	***	36	2.92	7.07×10^{-27}	***
Average EC	36	2.92	7.07×10^{-27}	***	36	2.92	7.07×10^{-27}	***
Average ECC	37	3.00	2.58×10^{-30}	***	36	2.92	7.07×10^{-27}	***
Average Katz	36	2.92	7.07×10^{-27}	***	35	2.84	4.65×10^{-24}	***
Average L	36	2.92	7.07×10^{-27}	***	37	3.00	2.58×10^{-30}	***
Average PR	36	2.92	7.07×10^{-27}	***	36	2.92	7.07×10^{-27}	***

As seen in Table 4.3, between 92 and 96 residues were shared among the 98 NSP16 residues selected for the top bin of both systems (with and without SAM). This high degree of overlap resulted in substantial enrichment fold values, ranging from 2.81 to 2.93 across the averaged DRN metrics. The hypergeometric test p-values for these enrichments were highly significant, ranging from 1.21×10^{-60} to 1.46×10^{-72}, with all values marked as highly significant (***). Similarly, the residues in the bottom bins exhibited enrichment fold values between 2.86 and 2.93, with p-values ranging from 4.81×10^{-63} to 5.93×10^{-69}, also indicating strong statistical significance.

The hypergeometric test was applied to evaluate whether there is a statistically significant overlap in the residues assigned to the top and bottom bins of these two systems based on their average DRN metrics. The strong statistical significance observed across all metrics meant the null hypothesis was rejected, and the alternate hypothesis that the high proportion of residues shared between the top or bottom bins of the system with SAM and the system without SAM is unlikely to be due to random chance was found. This suggests that the systems are broadly similar, and even with the addition of SAM, many of the most central residues remain consistent.

The results of the hypergeometric test for the NSP10 protein are very similar to NSP16 in that the test consistently showed highly statistically significant overlaps in the residues selected for the top and bottom bins of the averaged DRN metrics. However, as seen in Table 4.4, NSP10 had fewer residues in these bins, with the number of overlapping residues ranging from 35 to 37. This smaller overlap compared to NSP16 is likely due to NSP10 having only 111 selected residues overall, in contrast to the selected 293 residues in NSP16.

Despite the difference in the total number of residues, the enrichment fold values for NSP10 remained substantial, ranging from 2.84 to 3.00, with p-values ranging from 4.65×10^{-24} to 2.58×10^{-30}. These results also confirm that the overlap between the top and bottom bins of the systems with and without SAM is highly significant and unlikely to occur by chance.

Overall, these findings suggest that while the enrichment fold values differ between NSP10

and NSP16 due to their size disparity, the addition of SAM appears to not significantly alter the top and bottom residues identified by their DRN metrics in either protein. This suggests that the presence of SAM does not produce significant changes that would result in the residues of either NSP16 or NSP16 being grouped into different bins.

4.3.3.1.2 Difference between DRN Predictions of Residue Mutations

A total of 20 900 SNV mutations were identified from the filtered GISAID SARS-CoV-2 sequences for the selected residues of the NSP16 protein (residues 1-293). Of these 293 residues, 209 were found to have one or mutations, and 106 of these had mutation frequencies over 20 (36.18% of residues). The residue with the maximum number of mutations for an NSP16 residue was 3035 for P215. For the selected 111 NSP10 residues from MD simulations (residues 23 - 133), a total of 3551 SNV mutations were identified. 78 of these residues have one or more mutations, and 35 of these residues (31.53%) have mutation frequencies higher than 20. In Chapter 2 and Barozi *et al.* (2024), an M^{pro} residue was classified as mutating if the mutation frequency was over 20 or non-mutating otherwise. This cutoff was also implemented for the residues of the selected NSP16 and NSP10 residues.

Table 4.5 presents the results of the hypergeometric test, investigating whether non-mutated residues in NSP16 from simulations without SAM are overrepresented (enriched) or underrepresented (depleted) in the top and bottom thirds of the eight DRN metrics. Table 4.6 provides the corresponding hypergeometric test results for the NSP16 residue DRN metrics averaged from the simulations with SAM. These tables allow for a direct comparison of enrichment and depletion folds between the SAM-free and ligand-bound forms of NSP16.

Table 4.5: Enrichment fold and the enrichment/depletion p-values from hypergeometric tests of non-mutated residues in the top and bottom third of the DRN metrics for NSP16 without SAM (non-mutated cut-off of 20).

Averaged DRN Metric	**No. of Non-Mutated Residues (Top Bin)**	**Enrichment Fold**	**P-value**		**No. of Non-Mutated Residues (Bottom Bin)**	**Depletion Fold**	**P-value**	
Average BC	75	1.20	8.72×10^{-4}	***	45	1.38	1.29×10^{-5}	***
Average CC	75	1.20	8.72×10^{-4}	***	47	1.32	1.10×10^{-4}	***
Average DC	82	1.31	2.14×10^{-7}	***	48	1.29	2.92×10^{-4}	***
Average EC	71	1.14	0.0194	*	55	1.13	0.0495	*
Average ECC	78	1.25	4.00×10^{-5}	***	52	1.19	7.82×10^{-3}	**
Average L	75	1.20	8.72×10^{-4}	***	47	1.32	1.10×10^{-4}	***
Average PR	76	1.22	3.37×10^{-4}	***	46	1.35	3.89×10^{-5}	***
Average Katz	77	1.23	1.21×10^{-4}	***	48	1.29	2.92×10^{-4}	***

Table 4.6: Enrichment fold and the enrichment/depletion p-values from hypergeometric tests of non-mutated residues in the top and bottom third of the DRN metrics for NSP16 with SAM (non-mutated cut-off of 20).

Averaged DRN Metric	**No. of Non-Mutated Residues (Top Bin)**	**Enrichment Fold**	**P-value**		**No. of Non-Mutated Residues (Bottom Bin)**	**Depletion Fold**	**P-value**	
Average BC	76	1.22	3.37×10^{-4}	***	47	1.32	1.10×10^{-4}	***
Average CC	74	1.18	2.10×10^{-3}	**	48	1.29	2.92×10^{-4}	***
Average DC	81	1.30	8.99×10^{-7}	***	48	1.29	2.92×10^{-4}	***
Average EC	68	1.09	0.1004		55	1.13	4.95×10^{-2}	*
Average ECC	75	1.20	8.72×10^{-4}	***	49	1.26	7.27×10^{-4}	***
Average L	74	1.18	2.10×10^{-3}	**	48	1.29	2.92×10^{-4}	***
Average PR	79	1.26	1.23×10^{-5}	***	46	1.35	3.89×10^{-5}	***
Average Katz	76	1.22	3.37×10^{-4}	***	46	1.35	3.89×10^{-5}	***

In this case, the hypergeometric test has the null hypothesis, where the proportion of non-mutating residues that fall within the top or bottom third of DRN metric values is equal to the proportion of non-mutating residues within the whole protein. Small deviations ($p > 0.05$) from this proportion would be seen as not statistically significant. The alternate hypothesis is that the proportion of non-mutating residues falling within the top third of DRN metric values is overrepresented relative to the total number of non-mutating residues in the protein and underrepresented within the bottom third of DRN metric values compared to the total proportion of non-mutating residues in the protein.

In Table 4.5, all DRN metrics display enrichment (fold values greater than 1), indicating that the non-mutated residues are more prevalent in the top bin than the middle and bottom bins. For example, the *Average DC* metric demonstrates an enrichment fold of 1.31, indicating that non-mutated residues are 31% more likely to be found in the top bin than expected by chance ($p = 2.14 \times 10^{-7}$). Since the p-value is below 0.05, we reject the null hypothesis. As residues within the top bin are the top third of residues with the highest values for a given DRN metric, this enrichment suggests a tendency for non-mutated residues to cluster within central network regions, potentially playing key structural or functional roles. However, some metrics, such as *Average EC*, show lower enrichment fold values (1.14) with only borderline statistical significance ($p = 0.0194$), suggesting variability in the degree of enrichment among different centrality metrics.

In the presence of SAM, the enrichment folds for non-mutated residues in the top bin are consistent across most metrics (Table 4.6). However, these values are sometimes lower than those without the SAM ligand. For example, the *Average DC* metric with SAM shows an enrichment fold of 1.30 ($p=8.99 \times 10^{-7}$), a slight decrease compared to the complex-only form. Similarly, the *Average EC* metric has an enrichment fold of 1.09 in the presence of SAM ($p=0.1004$), which is lower and not statistically significant compared to the without SAM complex. These reductions in the number of non-mutating residues in the top bin values suggest that the presence of SAM may reduce the enrichment strength for some centrality metrics, possibly due to structural or functional changes in the protein.

For residues in the bottom bin, both SAM-free and SAM-bound systems consistently show

non-mutated residues being underrepresented in the bottom bins for the DRN metrics. Without SAM, the *Average BC* metric has a depletion fold of 1.38 (p=1.29×10^{-5}), while with SAM, the depletion fold is reduced to 1.32 (p=1.10×10^{-4}). Similar decreases are observed for other metrics, such as *Average ECC*, where depletion folds are 1.34 (p=3.42×10^{-5}) without SAM and 1.26 (p=7.27×10^{-4}) with SAM. These results indicate that while reduced proportions of non-mutating residues in the bottom third of residues remain in the presence of SAM, the fold magnitude appears to be reduced.

Overall, the comparison between Tables 4.5 and 4.6 indicate that the enrichment and depletion values for non-mutated residues in NSP16 are similar when simulating the NSP10-NSP16 complex both with and without SAM. However, the presence of SAM appears to slightly weaken the enrichment or depletion folds for certain metrics, potentially reflecting changes in the protein network connectivity and residue importance induced by ligand binding. As non-mutated residues are consistently found in the top third bin, containing residues in central, highly connected regions, this suggests their significance in maintaining the structure and essential functions of NSP16.

The enrichment fold values and corresponding p-values from the hypergeometric test for the non-mutated residues in the top and bottom thirds of the DRN metrics for NSP10 systems (with and without SAM) are summarised in Tables 4.7 and 4.8.

In the absence of SAM (Table 4.7), the enrichment fold values for the top-bin non-mutated residues ranged from 1.07 to 1.38 across averaged DRN metrics. Notably, *Average DC*, *Average PR*, and *Average Katz* metrics exhibited the highest enrichment (folds of 1.38, 1.38, and 1.30, respectively), with highly significant p-values of 8.96×10^{-6}, 8.96×10^{-6}, and 5.72×10^{-4}, respectively. For the bottom bin, the depletion fold values were generally higher than those for the top bin, ranging from 1.10 to 1.49. *Average PR* showed the highest depletion fold (1.49) with an extremely significant p-value of 3.98×10^{-4}.

Table 4.7: Enrichment fold and the enrichment/depletion p-values from hypergeometric tests of non-mutated residues in the top and bottom third of the DRN metrics for NSP10 without SAM (non-mutated cut-off of 20).

Averaged DRN Metric	**No. of Non-Mutated Residues (Top Bin)**	**Enrichment Fold**	**P-value**		**No. of Non-Mutated Residues (Bottom Bin)**	**Depletion Fold**	**P-value**	
Average BC	30	1.18	0.0333	*	19	1.33	0.00619	**
Average CC	30	1.18	0.0333	*	21	1.21	0.0494	*
Average DC	35	1.38	8.96×10^{-6}	***	18	1.41	1.71×10^{-3}	**
Average EC	30	1.18	0.0333	*	20	1.27	0.01897	*
Average ECC	27	1.07	0.309		23	1.1	0.213	
Average L	30	1.18	0.0333	*	21	1.21	0.0494	*
Average PR	35	1.38	8.96×10^{-6}	***	17	1.49	3.98×10^{-4}	***
Average Katz	33	1.30	5.72×10^{-4}	***	18	1.41	1.71×10^{-3}	**

Table 4.8: Enrichment fold and the enrichment/depletion p-values from hypergeometric tests of non-mutated residues in the top and bottom third of the DRN metrics for NSP10 with SAM (non-mutated cut-off of 20).

Averaged DRN Metric	**No. of Non-Mutated Residues (Top Bin)**	**Enrichment Fold**	**P-value**		**No. of Non-Mutated Residues (Bottom Bin)**	**Depletion Fold**	**P-value**	
Average BC	30	1.18	0.0333	*	19	1.33	0.00619	**
Average CC	30	1.18	0.0333	*	21	1.21	0.0494	*
Average DC	34	1.34	8.58×10^{-5}	***	18	1.41	1.71×10^{-3}	**
Average EC	30	1.18	0.0333	*	21	1.21	0.0494	*
Average ECC	27	1.07	0.309		23	1.1	0.213	
Average L	30	1.18	0.0333	*	21	1.21	0.0494	*
Average PR	35	1.38	8.96×10^{-6}	***	17	1.49	3.98×10^{-4}	***
Average Katz	33	1.30	5.72×10^{-4}	***	18	1.41	1.71×10^{-3}	**

In the presence of SAM (Table 4.8), enrichment trends in the top bin closely mirrored those observed in the system without SAM, with fold values ranging from 1.07 to 1.38. Notably, *Average DC* and *Average PR* again displayed the highest enrichments (folds of 1.34 and 1.38, respectively) with highly significant p-values of 8.58×10^{-5} and 8.96×10^{-6}, respectively. The consistency between systems suggests a similar centrality distribution across key residues. For the bottom bin, depletion folds remained comparable, with values ranging from 1.10 to 1.49. *Average PR* showed the highest depletion fold (1.49) with a significant p-value of 3.98×10^{-4}.

Across both systems, the *Average ECC* metric consistently exhibited the lowest enrichment and depletion fold values (1.07 and 1.10, respectively) and was not statistically significant ($p > 0.05$). The null hypothesis could not be rejected, meaning that the difference in the proportion of non-mutating residues between the top/bottom bins and the whole protein was likely due to random chance. This suggests that *ECC* metrics may be less sensitive to distinguishing centrality shifts between the top and bottom bins.

Conversely, *Average DC*, *Average PR*, and A*verage Katz* metrics consistently showed the highest enrichments and depletions across both systems, with highly significant p-values ($p < 0.001$). The null hypothesis was rejected, indicating that the proportions of non-mutating residues for the top and bottom bins were not overrepresented and underrepresented by random chance. This highlights these metrics as robust indicators of residue centrality and enrichment across both systems. Although these enrichment and depletion fold values are comparable to those observed in NSP16, the smaller population size in NSP10 systems likely limits their statistical significance.

4.3.3.1.3 DRN Metric Values Difference With and Without SAM

The hypergeometric tests have previously been used to determine whether a subset of the protein residues has been underrepresented or overrepresented within a larger group. However, it is also necessary to determine whether there is a significant difference between the average DRN metric values of the residues from simulations with and without SAM. This can be achieved using the Paired T-test, which determines whether there is or is not a

difference between the means of the two groups. This test can only be used on data which is normally distributed. The Wilcoxon Signed-Rank test was used when the DRN metric values were not normally distributed, determining whether the medians of the two groups are significantly different or not. The Shapiro-Wilks and Kolmogorov-Smirnov (KS) tests were employed to determine whether a given DRN metric has a normal distribution to determine whether to apply the Paired T-test or Wilcoxon Signed-Rank test. The results of these statistical tests can be found for NSP16 in Table 4.9 and Table 4.10 for NSP10.

In Table 4.9, the Shapiro-Wilk and KS tests indicate that most DRN metrics for the NSP16 residues do not follow a normal distribution, with the null-hypothesis of normality being rejected for one or more of these tests ($p < 0.05$). Exceptions include *Average ECC* and *Average PR*, which exhibit p-values over 0.05, suggesting normality. These results guided the choice of paired tests. The Wilcoxon Signed-Rank (WSR) test was chosen for all DRN metrics which did not follow a normal distribution. For *Average ECC* and *Average PR*, a Paired T-test was appropriate due to the normality of the data.

For NSP16, Table 4.9 showed a highly significant difference for the *Average EC* metric ($p = 5.29 \times 10^{-5}$), indicating that the null hypothesis is rejected and the median of the differences between paired DRN values is not zero. Similarly, the median of the differences between the SAM-bound and SAM-free paired DRN *Average Katz* values were significantly deviating from zero ($p = 0.0346$). These results suggest that the presence of SAM induces structural changes in the protein, as reflected by these changes between the systems of these specific metrics. In contrast, other metrics such as *Average BC*, *CC*, *DC*, *L*, and *PR* showed no significant differences ($p > 0.05$), supporting the null hypothesis that the median or mean differences in paired DRN values per residue are zero. This indicates consistent behaviour between the SAM-bound and SAM-free states for these metrics, suggesting they are less affected by the presence of SAM in terms of residue-specific DRN metric changes.

Table 4.9: Results of statistical tests for normality and paired tests on the Average DRN metrics of the NSP16 datasets with and without SAM.

Metric	**SAM Shapiro p-value**	**Apo Shapiro p-value**	**SAM KS p-value**	**Apo KS p-value**	**Test Type**	**p-value**	
Average BC	1.18×10^{-13}	2.36×10^{-13}	2.61×10^{-5}	4.76×10^{-5}	WSR	0.333	
Average CC	0.0139	0.0130	0.473	0.410	WSR	0.199	
Average DC	0.00729	0.0170	0.327	0.495	WSR	0.0742	
Average EC	1.71×10^{-24}	6.98×10^{-24}	1.84×10^{-17}	4.75×10^{-16}	WSR	5.29×10^{-5}	***
Average ECC	0.134	0.115	0.628	0.482	Paired T-test	0.523	
Average katz	6.68×10^{-10}	1.36×10^{-9}	0.0954	0.0613	WSR	0.0346	*
Average L	3.33×10^{-4}	1.88×10^{-4}	0.152	0.122	WSR	0.418	
Average PR	0.0291	0.0388	0.470	0.455	WSR	0.409	

Table 4.10: Results of statistical tests for normality and paired tests on the Average DRN metrics of the NSP10 datasets with and without SAM.

Metric	**SAM Shapiro p-value**	**Apo Shapiro p-value**	**SAM KS p-value**	**Apo KS p-value**	**Test Type**	**p-value**	
Average BC	5.89×10^{-9}	8.05×10^{-9}	3.96×10^{-3}	4.03×10^{-3}	WSR	0.0310	*
Average CC	0.296	0.328	0.810	0.895	Paired T-test	3.49×10^{-14}	***
Average DC	0.0450	0.0597	0.302	0.409	WSR	0.328	
Average EC	1.69×10^{-10}	1.38×10^{-10}	9.70×10^{-4}	1.00×10^{-3}	WSR	0.408	
Average ECC	0.0310	0.0336	0.409	0.415	WSR	4.89×10^{-6}	***
Average katz	6.64×10^{-6}	9.55×10^{-6}	0.119	0.104	WSR	0.588	
Average L	3.98×10^{-4}	3.41×10^{-4}	0.223	0.219	WSR	4.68×10^{-15}	***
Average PR	0.101	0.100	0.689	0.563	Paired T-test	1.00	

For NSP10 (Table 4.10), the normality tests show a similar trend, with most metrics failing the Shapiro-Wilk and KS tests, indicating these metrics are not normally distributed. Notably, *Average CC* and *Average PR* exhibit higher p-values, suggesting normality for both SAM-bound and SAM-free states, where the null hypothesis could not be rejected.

The mean or median of the differences between average system DRN metrics were analysed. For instance, the median of the differences in *Average BC* metric values was found to be significant ($p = 0.0310$), indicating altered residue interactions due to SAM binding. *Average CC* and *ECC* were also highly significant ($p = 3.49 \times 10^{-14}$ and $p = 4.89 \times 10^{-6}$, respectively), suggesting substantial changes in network centrality and connectivity in the presence of SAM. Additionally, *Average L* demonstrated high significance ($p = 4.68 \times 10^{-15}$), highlighting structural differences in the residue network topology between the two states.

In contrast, other NSP10 metrics, including *Average DC*, *EC*, *Katz*, and *PR*, did not show significant differences ($p > 0.05$). These results imply that while certain DRN properties remain stable regardless of SAM binding, others are significantly influenced, underscoring the functional role of SAM in altering the network dynamics of NSP10.

4.3.4 Selection of DRN Datasets for Residue Mutation Predictions

Four datasets were generated, each containing eight DRN centrality metrics for the NSP10 and NSP16 proteins from systems simulated with and without SAM. Analysis of these datasets revealed that nearly all residues remained in the same top, middle, or bottom bins across both conditions, indicating a high degree of consistency in residue categorization regardless of SAM presence. However, certain metrics exhibited statistically significant differences in their mean or median values, suggesting that the inclusion of SAM could influence specific residue interactions. Since the impact of these changes on mutation prediction performance is uncertain, all datasets were included as testing data for M^{pro}-trained ML models to ensure a comprehensive and robust evaluation.

4.4 Part III: Application of M^{pro} models on NSP10 and NSP16 Data

4.4.1 Introduction

To determine the generalisability and robustness of the M^{pro}-trained ANN, SVM and RF models, testing with data from other proteins was conducted. To do this, data sets for the NSP10 and NSP16 proteins with the same features as the M^{pro} data set were produced and provided as test data. No further tuning or training was performed on the M^{pro}-trained models in order to ascertain direct transferability of the models to other SARS-CoV-2 proteins.

4.4.2 Methodology

4.4.2.1 Creation of NSP10 and NSP16 Datasets

Datasets for the NSP10 and NSP16 proteins of SARS-CoV-2 were produced using the methods listed in Barozi *et al.* (2024) for the M^{pro} dataset. This included the performed DRN analysis described in Section 4.3.2.1 on each of the specified residues found with the least fluctuation, which resulted in 8 averaged DRN metrics for each residue of the protein. The B-factor, RMSF and SASA values for each residue were calculated using the `gmx` tool with the commands explained in Barozi *et al.* (2024), Section 2.4. The BLOSUM62 values for each residue were extracted from the BLOSUM62 matrix using the in-house notebook. Mutation frequencies for each NSP10 and NSP16 residue, obtained using the methodology outlined in Section 4.3.2.2, were included as the target values.

4.4.2.2 Performance Metrics for the Chosen ML Models

Similarly to the M^{pro} dataset, uneven class distributions were identified in the NSP10 and NSP16 datasets. However, accuracy was chosen as the performance metric, allowing for a

direct comparison in performance between the ML models tested on the NSP10 and NSP16 datasets and their performance on the M^{pro} dataset in Chapter 2. In addition, confusion matrices containing the accuracy, precision and recall of the models were produced for a small subset of the models to analyse the results more fully.

4.4.2.3 Application and Evaluation of M^{pro}-trained Models on NSP10 and NSP16 Datasets

The classification models using the Experiment 4 architecture were trained on a randomly selected subset of the M^{pro} dataset and tested on the remaining M^{pro} residues. These trained models were subsequently applied to four different NSP10 and NSP16 datasets, simulated with and without SAM. To maintain consistency with the M^{pro} dataset, a cutoff value of 20 was used to identify target residues as mutating. Each ML model underwent five iterations of random training and testing, with the average accuracy and standard deviation computed across these runs. Additionally, three randomly selected ANN models were further analysed by generating confusion matrices for each tested dataset (M^{pro}, NSP10 and NSP16 simulated without SAM). This additional analysis aimed to evaluate the model's performance using supplementary metrics, providing deeper insight into its effectiveness when applied to new datasets.

The M^{pro} dataset was also used to train, validate and test the regression ML models produced according to the given specifications in Experiment 1. Similarly, these trained models were applied to the four NSP10 and NSP16 datasets, and the average R-value and standard deviations for these models were calculated over five different random runs.

Further investigation into the effect of the chosen cutoff was conducted. The confusion matrices of a randomly trained ANN model on the M^{pro} dataset were compared when tested on the NSP16 with SAM dataset under three different cutoff conditions: (1) using the same cutoff as the M^{pro} dataset, (2) matching the proportion of total mutations to the total number of residues in the protein, and (3) maintaining a 70% proportion of non-mutating residues to 30% mutating residues.

4.4.3 Results and Discussion

4.4.3.1 Classification for NSP16 and NSP10 Datasets

Table 4.11 contains the average accuracies for the NSP10 and NSP16 datasets with and without SAM when provided as input to the classification ML models trained and tested on the original 31-feature M^{pro} dataset. An overall average accuracy of around 68% was observed for the NSP10 datasets, which decreased to around 65% for the NSP16 datasets. This is a drop in accuracy compared to the M^{pro} test subset average accuracies of around 76%. This suggests that the current features of these datasets are not fully transferable, likely due to structural and functional differences between the M^{pro}, NSP10 and NSP16 proteins. A slight increase in the average accuracies (around 1%) was observed for both NSP10 and NSP16 when the datasets using the trajectories simulated with the SAM ligand were used as input. These datasets consist of the same 31 features of the original M^{pro} dataset, as well as the NSP10 and NSP16 datasets simulated without SAM. The features include 8 DRN centrality values, RMSF, SASA and B-factor, and the 20 BLOSUM62 metrics per residue. However, it appears that the inclusion of the ligand is still insufficient for overall robust generalisation.

Table 4.11: Average accuracies of the ML models after hyperparameter tuning using a subset of the M^{pro} dataset, tested on the remaining M^{pro} dataset values and the NSP10 and NSP16 datasets with and without SAM.

Model	M^{pro} Training Set Accuracy (Mean ± SD)	M^{pro} Testing Set Accuracy (Mean ± SD)	NSP16 with SAM Accuracy (Mean ± SD)	NSP10 with SAM Accuracy (Mean ± SD)	NSP16 without SAM Accuracy (Mean ± SD)	NSP10 without SAM Accuracy (Mean ± SD)
ANN	82.9245 ± 1.7999	77.3913 ± 2.9488	64.8464 ± 0.4317	68.2883 ± 0.3604	56.7235 ± 3.4966	65.4054 ± 5.2716
	83.2075 ± 1.9009	74.1304 ± 4.1476	66.0751 ± 1.3754	68.4685 ± 0.5698	55.9727 ± 1.1422	68.1081 ± 0.4413
	82.4528 ± 1.6176	75.8696 ± 1.8701	65.6655 ± 0.5951	68.4685 ± 0.0000	56.9283 ± 1.6496	66.4865 ± 3.5308
RF	76.4297 ± 1.2405	76.0870 ± 2.1739	63.8225 ± 0.6105	68.6486 ± 1.0506	65.1877 ± 0.8360	68.1081 ± 0.7207
	77.8272 ± 1.8124	76.9565 ± 3.0279	65.5973 ± 0.7902	67.7477 ± 1.0506	64.9147 ± 0.5461	68.4685 ± 0.5698
	77.1694 ± 1.6944	77.6087 ± 3.5455	64.9147 ± 0.5016	69.1892 ± 1.0506	64.5734 ± 0.5016	67.7477 ± 1.0506
SVM	77.2642 ± 4.9057	70.4348 ± 4.6320	64.2321 ± 0.2554	67.0270 ± 0.4413	65.8020 ± 0.1365	69.3694 ± 0.0000
	81.5094 ± 3.0336	70.8696 ± 4.6320	66.0068 ± 0.5545	66.8468 ± 0.3604	64.3003 ± 0.1672	68.4685 ± 0.0000
	88.4906 ± 4.2358	68.2609 ± 5.7269	64.7782 ± 0.1365	70.2703 ± 0.0000	64.3686 ± 0.2730	68.4685 ± 0.0000

4.4.3.1.1 Confusion Matrices of Selected ANN Models

The only metric used in this study to compare the performance of different classification ML models is accuracy. This was done since comparing a single metric between different models is easier while attempting to improve the model's overall performance. However, it became necessary to produce a more detailed overview of the performance of a subset of the models when applied to the new NSP10 and NSP16 datasets. This is because accuracy alone does not provide enough information about the changes in the classification of these new residues, particularly as the datasets have class imbalances. Therefore, the confusion matrices for three randomly chosen ANN models trained on the M^{pro} dataset and tested on the M^{pro}, NSP10 and NSP16 datasets without SAM were produced, as seen in Figures 4.17 and 4.18.

In Figure 4.17, it is shown that the precision of the training set ranges from 72.9% to 78.4%, while the recall is slightly lower, ranging from 60.9% to 67.2%. Therefore, it appears that the trained ANN models are better at predicting positive cases correctly than finding all the actual positive cases. This also applies to the test subset, where the precision values range from 62.5% and 66.7%, compared to the lower recall values ranging between 35.7% and 57.1%. The accuracy of the test subset is much higher, ranging from 73.9% to 78.3%. The confusion matrices indicate that the highest proportion of residues fall within the true negative category, meaning that the model correctly predicted these residues as non-mutating. This suggests that the model performs well in identifying residues that do not meet the criteria for mutation. However, it may struggle with accurately predicting mutating residues, as reflected in the relatively lower true positive rates and higher false negative rates.

When these trained models were applied to the NSP16 dataset simulated without SAM, the precision improved to values ranging between 73.0% and 100%, as seen in Figure 4.18. However, the recall for these models decreased dramatically, only reaching between 5.1% and 19.7%. This is explained by the decrease in false positives improving precision and the large increase in false negatives reducing recall when this dataset is provided as input.

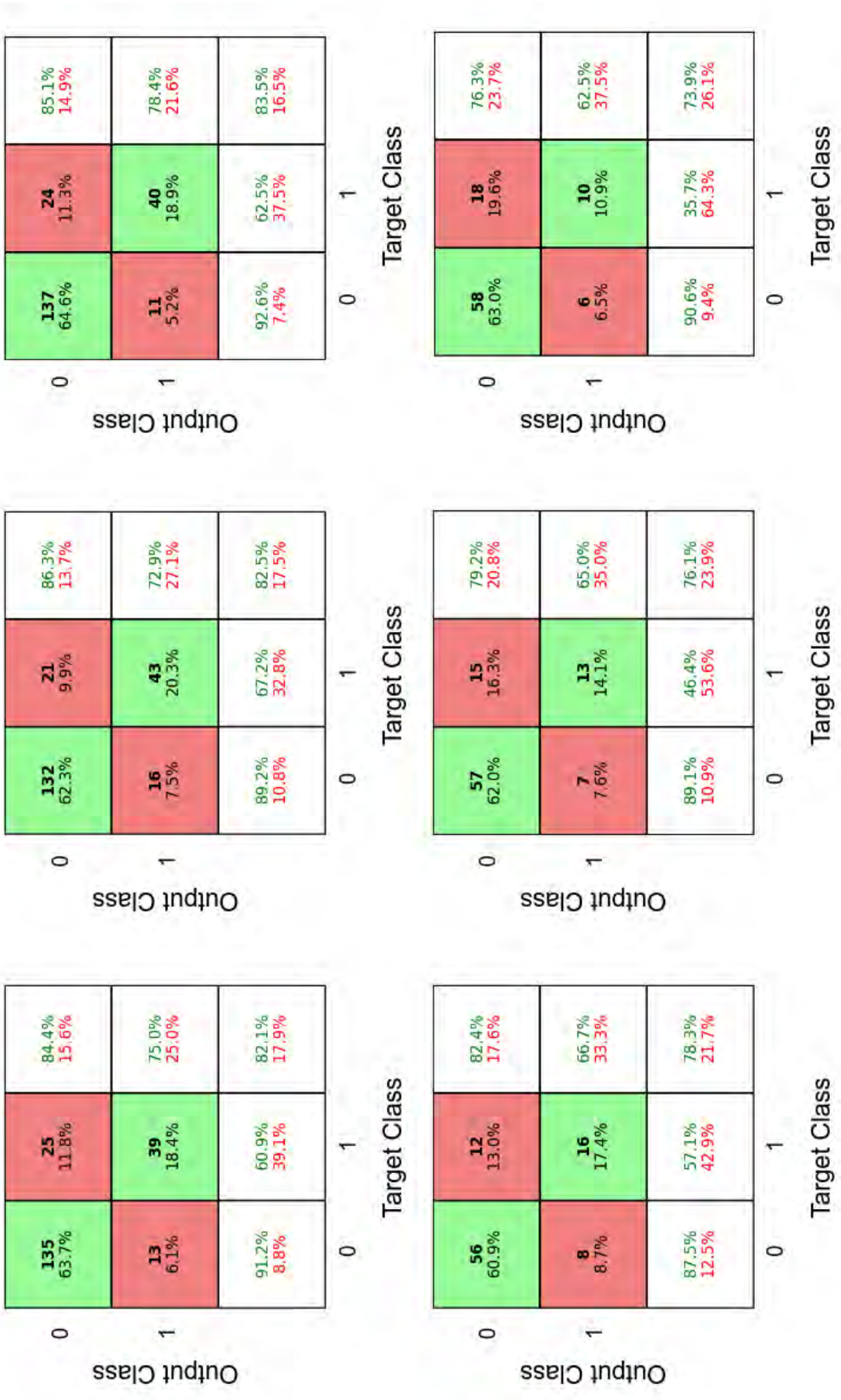

Figure 4.17: Confusion matrices of the training (top row) and test (bottom row) subsets produced from three randomly trained ANN models on the M^{pro} dataset.

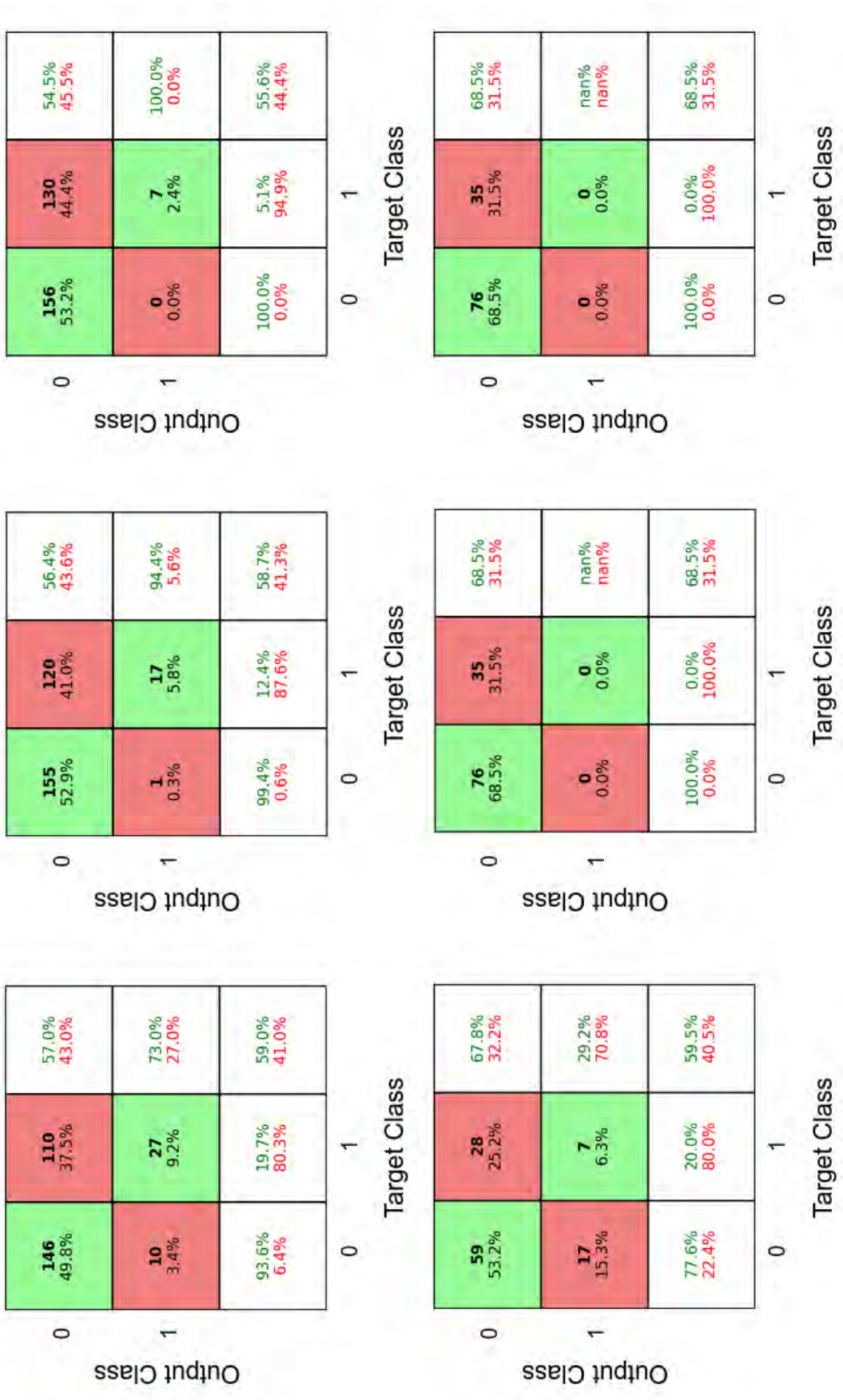

Figure 4.18: Confusion matrices of the NSP16 (top row) and NSP10 (bottom row) datasets simulated without SAM, produced from three randomly trained ANN models on the M^{pro} dataset.

Two of the three randomly trained M^{pro} models only predicted non-mutated residues when the NSP10 without SAM dataset was provided as input. This means that it was not possible to calculate the precision of the model since there were no false or true positives. Also, there were no true positives, so the recall was always 0% for these models. For the last random model, the precision (29.2%) and recall (20.0%) were low. However, the overall accuracy of the models was much higher than the precision and recall, falling between 59.5% and 68.5%. This is because around 68.5% of the residues of this protein are classified as not mutating when using the cutoff of 20.

These findings suggest that the ANN models trained on the M^{pro} data are not generalising well to the NSP10 and NSP16 datasets simulated without SAM. Also, the improvements in precision for the NSP16 dataset came at the cost of a very low recall rate. This indicates that it is necessary to improve the current models further using techniques such as transfer learning or further feature engineering. The tendency of the ANN models to predict non-mutating residues for the new protein datasets was not immediately evident when only looking at the accuracy metrics. This highlights the necessity of using multiple metric values to more deeply understand the performance of the models.

4.4.3.1.2 Effect of Cutoff Choice on Confusion Matrices

Previously, the cutoff for all datasets was set to 20, meaning that any residues with mutation frequencies of 20 or over are classified as mutating, while the rest are non-mutating. However, changes to the cutoff values result in different proportions of residues within the mutating and non-mutating classes, which could impact the overall performance of the models. To investigate this, the NSP16 with SAM dataset was chosen, and the performance of a random model was investigated using three different cutoffs. The cutoff values tested were 10 (matching the proportion of total mutations of the protein to the number of residues of the protein used for M^{pro}), 20 (same cutoff used for M^{pro}), and 30 (for around 70% of residues to be classified as non-mutating, same as M^{pro}). The results are seen in Figure 4.19.

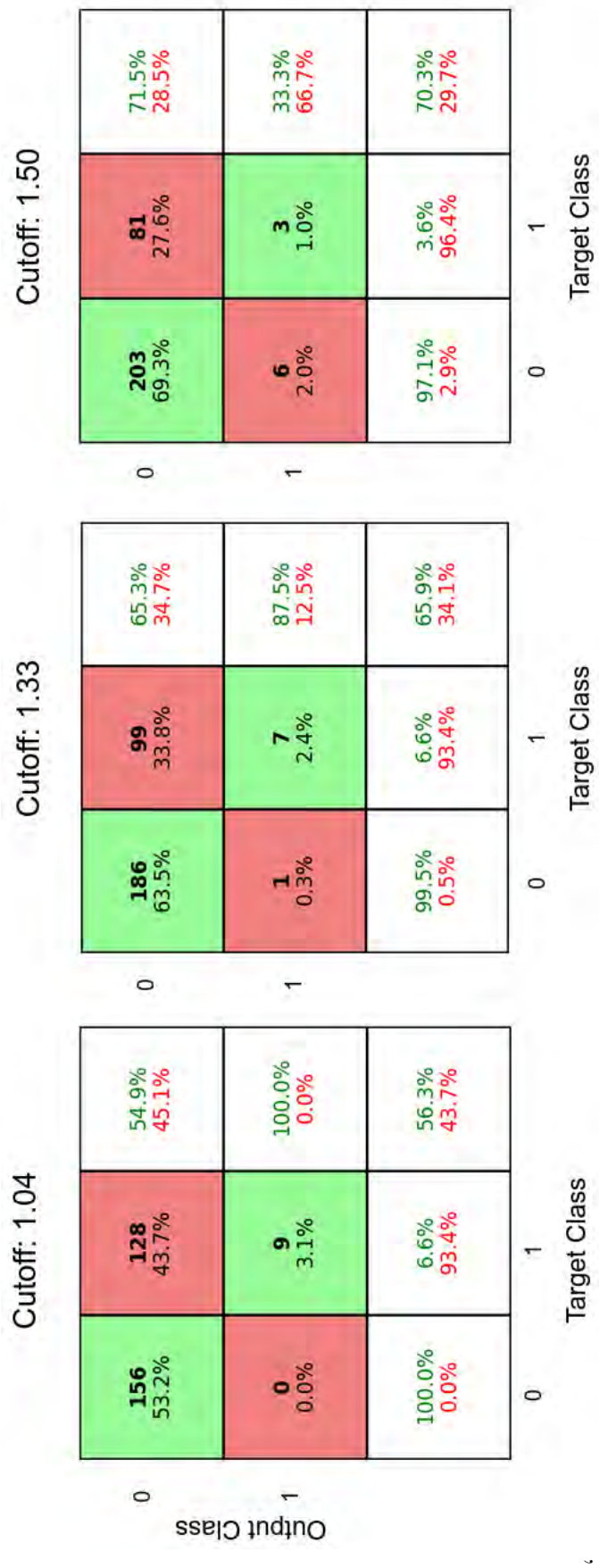

Figure 4.19: Confusion matrices for the NSP16 dataset simulated with SAM, evaluated using M^{pro}-trained models. Mutation frequency cutoffs range from greater than 10 (log-transformed to 1.04) to greater than 30 (log-transformed to 1.50) to classify residues as mutating or non-mutating.

It appeared that by increasing the cutoff value (thereby reducing the number of residues classified as mutating), the number of true negatives increased, while the number of false negatives decreased. This is likely because a larger proportion of non-mutating residues are provided. Consequently, a model predisposed to predicting non-mutating residues is more likely to correctly classify them as true negatives, increasing the true negative count and decreasing the false negative count. However, it was also shown that by increasing the cutoff value, the number of false positives increased while the number of true positives decreased. This means the precision of the model decreased as the cutoff value increased, which can be seen from the precision value of 100% at a cutoff of 10 and only 33.3% at a cutoff of 30. This indicates that it became harder for a mutating residue to be correctly predicted as mutating for the residue. This may suggest that the criteria used to classify a residue as mutating for the M^{pro} dataset do not transfer to the NSP16 dataset. This is supported by the recall values remaining low, moving from 6.6% for low cutoffs to 3.1% for the high cutoff value.

Overall, a change in cutoff value appears to impact the performance of the ANN model, as seen by this example when testing using the NSP16 dataset simulated with SAM. Therefore, care must be taken to determine which cutoff should be applied to a given dataset, as adjustments can lead to trade-offs between accuracy, precision and recall. Determining what mutation frequency should be chosen as the cutoff for each protein through biological criteria may be necessary. Methods for identifying these criteria would need to be investigated in future research.

4.4.3.2 Regression for NSP16 and NSP10 Datasets

Similarly, Table 4.12 shows the average R-values of the regression models when trained on the M^{pro} dataset and applied to the NSP10 and NSP16 datasets with and without SAM. The average of the R-values was around 0.42, with the best average R-value nearly at 0.48. The SVM model's performance was noticeably lower than the ANN and RF models. In particular, negative average R-values were observed along with high variability for the NSP10 datasets, both with and without the SAM ligand. Similar to the classification

models, a drop in performance was observed between the results from testing using the M^{pro} dataset and the NSP10 and NSP16 datasets. This result indicates that the features are not fully transferable for the regression models either. The NSP10 and NSP16 datasets using data simulated with the SAM ligand were again producing slightly higher average R-values when used as input compared to the datasets without SAM but were insufficient for overall robust generalisation.

Table 4.12: Average R-values of the ML models after hyperparameter tuning using a subset of the M^{pro} dataset, tested on the remaining M^{pro} dataset values and the NSP10 and NSP16 datasets with and without SAM.

Model	M^{pro} Training Set R-value (Mean ± SD)	M^{pro} Validation Set R-value (Mean ± SD)	M^{pro} Testing Set R-value (Mean ± SD)	NSP16 with SAM R-value (Mean ± SD)	NSP10 with SAM R-value (Mean ± SD)	NSP16 without SAM R-value (Mean ± SD)	NSP10 without SAM R-value (Mean ± SD)
ANN	0.7249 ± 0.0198	0.3867 ± 0.0932	0.3975 ± 0.0843	0.4280 ± 0.0065	0.4683 ± 0.0422	0.4132 ± 0.0821	0.3635 ± 0.2936
	0.7191 ± 0.0169	0.4088 ± 0.1317	0.4681 ± 0.0255	0.4376 ± 0.0210	0.5014 ± 0.0159	0.3900 ± 0.0269	0.4545 ± 0.0519
	0.7037 ± 0.0259	0.4589 ± 0.0600	0.4877 ± 0.0831	0.3914 ± 0.0347	0.4565 ± 0.0248	0.3979 ± 0.0352	0.2475 ± 0.2038
RF	0.9176 ± 0.0304	0.5305 ± 0.1003	0.4893 ± 0.1214	0.4316 ± 0.0318	0.3701 ± 0.0518	0.4383 ± 0.0483	0.3999 ± 0.0524
	0.9038 ± 0.0498	0.4643 ± 0.1148	0.5612 ± 0.0990	0.4425 ± 0.0207	0.4004 ± 0.0524	0.4171 ± 0.0145	0.3987 ± 0.0369
	0.8680 ± 0.0494	0.5039 ± 0.1022	0.4984 ± 0.0827	0.4548 ± 0.0290	0.4352 ± 0.0419	0.4478 ± 0.0266	0.4161 ± 0.0439
SVM	0.6800 ± 0.0238	0.4910 ± 0.1164	0.5280 ± 0.0644	0.3161 ± 0.0132	-0.2162 ± 0.0648	0.3463 ± 0.0834	0.0227 ± 0.2591
	0.6326 ± 0.0688	0.5256 ± 0.1093	0.4898 ± 0.0613	0.3977 ± 0.0812	0.1333 ± 0.3313	0.2787 ± 0.1244	-0.0662 ± 0.3097
	0.5874 ± 0.0592	0.4994 ± 0.0762	0.5290 ± 0.1018	0.4259 ± 0.0777	0.1431 ± 0.3206	0.4294 ± 0.0839	0.2290 ± 0.3642

4.5 Summary

The PDB file 6W4H was identified as the most suitable starting structure for simulating the NSP10-NSP16 complex. The complex, including its zinc binding sites and SAM ligand, was parameterised for MD simulations. In terms of simulation settings, the V-rescale thermostat was selected for temperature control, the C-rescale barostat for equilibration, and the Parrinello-Rahman barostat was chosen for the production run, as this combination provided the best stability.

Four separate 1-microsecond simulations were performed on the system, both with and without SAM. During the simulations, it was observed that the RMSD fluctuations were generally lower for the complex when simulated with SAM compared to the system without SAM. However, in some cases, the RMSD exceeded 1 Å, suggesting that not all of the simulations equilibrated properly for both systems. Selected tail residues from both proteins were excluded due to the high RMSF values observed during the simulations, indicating they were very flexible and did not contribute to a stable simulation.

DRN analysis was conducted on selected residues 23–133 of NSP10 and residues 1–293 of NSP16. It was found that the residues in the top and bottom third of DRN metrics were nearly identical between the systems with and without SAM, both for the NSP10 and NSP16 proteins. Despite this, the difference in metric values for a given residue between the two systems appeared statistically significant for some metrics, making it unclear which system would be better suited for ML model testing. It was decided that DRN values from all simulations, regardless of SAM presence, would be used as input for ML datasets.

The models trained on the M^{pro} dataset did not appear to generalise well when applied to the other SARS-CoV-2 datasets (the NSP10 and NSP16 datasets with and without SAM), characterised by the decrease in average R-values and accuracies for these datasets compared to the test subset of M^{pro}. A slight improvement in average accuracy was observed when using the NSP16 dataset from simulations with SAM, compared to the average accuracy when using the NSP16 dataset simulated without SAM. This suggests

that different environmental conditions within the simulation can lead to changes in the performance of the models when run on that data.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this study, the ANN and RF models from Barozi *et al.* (2024) were fine-tuned, and new SVM models were developed in an attempt to enhance performance in predicting residue mutations for the SARS-CoV-2 M^{pro} protein. While hyperparameter tuning and PCA were applied to improve model performance, no significant enhancements were observed. The ANN and RF models exhibited moderate performance, with similar average accuracies and R-values to those in Barozi *et al.* (2024). However, the SVM model slightly underperformed compared to the other two models with the introduction of PCA. An experiment using fixed train-validation-test splits revealed that the variance in model performance was mainly due to the data splits due to a reduction in variance, highlighting the necessity of additional data samples to improve model performance. With the validation subset removed, the final classification models achieved an average accuracy limit of 76% for the test subset, similar to the average accuracy of the test subsets in Barozi *et al.* (2024). The ANN, SVM, and RF models demonstrated comparable performance, with the SVM model performing slightly worse than the ANN and RF models in classification tasks. This indicates that the choice of ML model may not significantly impact predictive accuracy for the M^{pro} dataset.

To address the potential loss of information from converting trajectory coordinates to averaged metrics per residue in the M^{pro} dataset, a new dataset was created using only raw trajectory coordinates from the MD simulations of the M^{pro} protein. Tuned CNN models were applied to this raw trajectory dataset, but their performance fell short of the best results of RF models trained on the M^{pro} dataset with processed features. This reduction in performance may be due to an incorrect choice of ML model. However, it could also be due to the lack of normalisation or grid-based formatting for the raw coordinate inputs, which may have hindered the ability of the models to identify any meaningful patterns within the data.

To test the generalisability of the M^{pro}-trained ANN, SVM and RF models, it was necessary to create new datasets of other SARS-CoV-2 proteins. MD simulations of the NSP10-NSP16 complex were conducted to produce stable trajectories, which were then analysed and processed as features for new NSP10 and NSP16 datasets. After selecting an appropriate PDB structure with ID 6W4H, zinc-binding sites in NSP10 and the SAM ligand were parameterised, and the best combination of MD parameters was identified through short equilibration runs. This led to four 1-microsecond MD simulation runs of the complex, simulated with and without SAM, to determine if the presence of the ligand improved the stability of the complex. However, neither of the systems had all replicate runs equilibrate, as fluctuations in RMSD values were over 1 Å. Tail residues of both proteins exhibited high fluctuations, prompting their removal in subsequent analyses to reduce overall RMSD and enhance interpretability.

The MD simulation trajectories were further processed using DRN analysis. It was determined that the residues with the top and bottom third DRN metrics remained almost identical between the systems with and without SAM. However, the difference in metric values between a given residue appeared to be statistically significant between the systems with and without SAM for some residues. This meant it was difficult to determine which of the systems would be better suited as input for residue mutation prediction. Therefore, the resultant DRN metrics, RMSF, SASA, B-factor and BLOSUM values for each residue were included as features, creating four input ML datasets (NSP10 and NSP16, each with and without SAM). In this way, it was investigated whether simulating the complex with

or without the ligand would affect the overall performance of the ML models.

When the M^{pro}-trained models were applied to NSP10 and NSP16 datasets, accuracies and R-values decreased compared to results on the M^{pro} test set. The models showed a bias toward predicting non-mutating residues, particularly for NSP10. These results indicate that the M^{pro}-trained ML models did not generalise well when applied to datasets of other SARS-CoV-2 proteins. In addition, it was found that the chosen cutoff for classifying residues mutating or not affected the overall performance of the models, increasing accuracy but reducing recall and precision.

This study identified critical challenges and limitations in predicting residue mutations in SARS-CoV-2 proteins. This includes small datasets with class imbalance, structural instabilities within the complex during MD simulation runs and computational power limits based on memory size. However, these challenges also highlight opportunities for improvement, and the results of this study demonstrate the potential of using ML to address these issues. Key findings included the importance of SAM for NSP16 stability, the role of feature selection in ML model performance, and the potential for alternative input formats and model types for residue mutation prediction using raw trajectory input. Further research with larger datasets and refined methodologies will be essential to overcoming these challenges and improving the predictive accuracy of ML models for SARS-CoV-2 proteins.

5.2 Future Work

Future work will address the challenges and limitations identified in this study and explore potential avenues to improve model performance.

One key priority is increasing the number of samples in the processed feature datasets. This may be possible by incorporating residues of the same protein (e.g., M^{pro}) found in other coronaviruses, such as SARS-CoV and MERS-CoV. Additionally, it would be useful to extend the investigation to other proteins of the SARS-CoV-2 virus to determine

whether the generalisations of the M^{pro}-trained models remain reduced. Another approach involves exploring alternative feature sets beyond the current combination of DRN metrics, SASA, B-factor, RMSF, and BLOSUM, followed by further hyperparameter tuning to enhance performance.

As class imbalance was observed for all datasets, future analyses should incorporate the F1-score as a secondary metric alongside accuracy to provide a more balanced model performance evaluation. Moreover, the cutoff frequency from GISAID used to define residue mutations requires re-examination. Investigating whether a biologically driven decision for setting this threshold exists could help refine model predictions, as the chosen cutoff directly impacts performance.

The poor performance of CNNs when using raw coordinate inputs indicates the need for further investigation. Future work should explore whether normalising the coordinate values (e.g., using a scalar) or converting the data into a grid-like format, where the presence or absence of atoms in specific 3D spaces is indicated, improves model outcomes. Additionally, simplifying the data by focusing solely on C-alpha atoms and removing coordinates with zero values may also help to determine whether the sparsity of the dataset contributed to the low performance. In addition, this would reduce memory requirements, potentially allowing for all frames to be provided as input.

Instead of applying CNNs to raw trajectory data, future investigation into other deep neural network models, such as graph neural networks (GNNs), may be important. GNNs inherently account for the spatial relationships and distances between atoms, making them potentially better suited to the structural nature of the current data, which is not inherently grid-like. These models may offer improved performance by naturally accounting for the connectivity of the atoms and residues within the protein structures.

Finally, further exploration of alternative ML models for residue mutation prediction is critical. Gaussian Processes are a promising option as they allow for classification and regression tasks and return not only the predicted value but also the confidence interval for this prediction. This could help identify residues for which the model has high or

low confidence, offering insights into residue-level prediction difficulty and guiding model refinement.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. TensorFlow: Large-scale machine learning on heterogeneous systems. 2015. Software available from tensorflow.org. URL https://www.tensorflow.org/

Abdi, H. and Williams, L. J. Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(4):433–459, 2010.

Abraham, M., Alekseenko, A., Basov, V., Bergh, C., Briand, E., Brown, A., Doijade, M., Fiorin, G., Fleischmann, S., Gorelov, S., Gouaillardet, G., Gray, A., Irrgang, M. E., Jalalypour, F., Jordan, J., Kutzner, C., Lemkul, J. A., Lundborg, M., Merz, P., Miletic, V., Morozov, D., Nabet, J., Pall, S., Pasquadibisceglie, A., Pellegrino, M., Santuz, H., Schulz, R., Shugaeva, T., Shvetsov, A., Villa, A., Wingbermuehle, S., Hess, B., and Lindahl, E. GROMACS 2024.4 manual. October 2024. doi:10.5281/zenodo.14016613. URL https://doi.org/10.5281/zenodo.14016613

Allen, M. P. and Tildesley, D. J. Computer Simulation of Liquids. Oxford University Press, Oxford, UK, 2017.

Anandakrishnan, R., Aguilar, B., and Onufriev, A. V. H++ 3.0: Automating pK prediction and the preparation of biomolecular structures for atomistic molecular modeling and simulations. *Nucleic Acids Research*, 40(W1):W537–W541, 2012.

Andersen, H. C. Molecular dynamics simulations at constant pressure and/or temperature. *The Journal of Chemical Physics*, 72(4):2384–2393, 1980.

Asuero, A. G., Sayago, A., and González, A. The correlation coefficient: An overview. *Critical Reviews in AnalyticalCchemistry*, 36(1):41–59, 2006.

Barozi, V., Chakraborty, S., Govender, S., Morgan, E., Ramahala, R., Graham, S. C., Bishop, N. T., and Özlem Tastan Bishop. Revealing SARS-CoV-2 Mpro mutation cold and hot spots: Dynamic residue network analysis meets machine learning. *Computational and Structural Biotechnology Journal*, 23:3800–3816, 2024. ISSN 2001-0370. doi:https://doi.org/10.1016/j.csbj.2024.10.031.

Berendsen, H. J., Postma, J. v., Van Gunsteren, W. F., DiNola, A., and Haak, J. R. Molecular dynamics with coupling to an external bath. *The Journal of Chemical Physics*, 81(8):3684–3690, 1984.

Berishvili, V. P., Perkin, V. O., Voronkov, A. E., Radchenko, E. V., Syed, R., Venkata Ramana Reddy, C., Pillay, V., Kumar, P., Choonara, Y. E., Kamal, A., and Palyulin, V. A. Time-domain analysis of molecular dynamics trajectories using deep neural networks: Application to activity ranking of tankyrase inhibitors. *Journal of Chemical Information and Modeling*, 59(8):3519–3532, Aug 2019. ISSN 1549-9596. doi:10.1021/acs.jcim.9b00135.

Berman, H., Henrick, K., and Nakamura, H. Announcing the Worldwide Protein Data Bank. *Nature Structural & Molecular Biology*, 10(12):980–980, 2003.

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., and Bourne, P. E. The Protein Data Bank. *Nucleic Acids Research*, 28(1):235–242, 2000.

Bouvet, M., Debarnot, C., Imbert, I., Selisko, B., Snijder, E. J., Canard, B., and Decroly, E. In vitro reconstitution of SARS-coronavirus mRNA cap methylation. *PLoS Pathogens*, 6(4):e1000863, 2010.

Braun, E., Gilmer, J., Mayes, H. B., Mobley, D. L., Monroe, J. I., Prasad, S., and Zuckerman, D. M. Best practices for foundations in molecular simulations [Article v1.0]. *Living Journal of Computational Molecular Science*, 1(1):5957, 2019. ISSN 2575-6524. doi:10.33011/livecoms.1.1.5957.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. Classification and Regression Trees. Wadsworth International Group, Belmont, CA, USA, 1984. ISBN 9780412048418.

Brown, D. K., Penkler, D. L., Sheik Amamuddy, O., Ross, C., Atilgan, A. R., Atilgan, C., and Tastan Bishop, Ö. MD-TASK: A software suite for analyzing molecular dynamics trajectories. *Bioinformatics*, 33(17):2768–2771, 2017.

Case, D. A., Aktulga, H. M., Belfon, K., Cerutti, D. S., Cisneros, G. A., Cruzeiro, V. W. D., Forouzesh, N., Giese, T. J., Götz, A. W., Gohlke, H. *et al.* AmberTools. *Journal of Chemical Information and Modeling*, 63(20):6183–6191, 2023.

Chen, C.-F. R., Panda, R., Ramakrishnan, K., Feris, R., Cohn, J., Oliva, A., and Fan, Q. Deep analysis of CNN-based spatio-temporal representations for action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6165–6175. 2021.

Chen, Y., Su, C., Ke, M., Jin, X., Xu, L., Zhang, Z., Wu, A., Sun, Y., Yang, Z., Tien, P., Ahola, T., Liang, Y., Liu, X., and Guo, D. Biochemical and structural insights into the mechanisms of SARS coronavirus RNA ribose 2'-O-methylation by nsp16/nsp10 protein complex. *PLOS Pathogens*, 7(10):e1002294, October 2011. ISSN 1553-7374. doi:10.1371/journal.ppat.1002294.

Collaborative Computational Project Number 4. REFMAC5: Principal X-ray keywords. 2024. Accessed: 2024-07-03.

URL https://www.ccp4.ac.uk/html/refmac5/keywords/xray-principal.html

Cortes, C. Support-vector networks. *Machine Learning*, 1995.

Deng, J., Yang, S., Li, Y., Tan, X., Liu, J., Yu, Y., Ding, Q., Fan, C., Wang, H., Chen, X., Liu, Q., Guo, X., Gong, F., Zhou, L., and Chen, Y. Natural evidence of coronaviral 2'-O-methyltransferase activity affecting viral pathogenesis via improved substrate RNA binding. *Signal Transduction and Targeted Therapy*, 9(1):1–16, May 2024. ISSN 2059-3635. doi:10.1038/s41392-024-01860-x.

Eissa, I. H., Alesawy, M. S., Saleh, A. M., Elkaeed, E. B., Alsfouk, B. A., El-Attar, A.-A. M. M., and Metwaly, A. M. Ligand and structure-based in silico determination of the most promising SARS-CoV-2 nsp16-nsp10 2'-O-methyltransferase complex inhibitors among 3009 FDA approved drugs. *Molecules*, 27(7), 2022. ISSN 1420-3049. doi:10.3390/molecules27072287.

Ferron, F., Decroly, E., Selisko, B., and Canard, B. The viral RNA capping machinery as a target for antiviral drugs. *Antiviral Research*, 96(1):21–31, October 2012. ISSN 0166-3542. doi:10.1016/j.antiviral.2012.07.007.

Fukuya, T. and Shibuta, Y. Machine learning approach to automated analysis of atomic configuration of molecular dynamics simulation. *Computational Materials Science*, 184:109880, 2020. ISSN 0927-0256. doi:https://doi.org/10.1016/j.commatsci.2020.109880.

Gorbalenya, A. E., Baker, S. C., Baric, R. S., de Groot, R. J., Drosten, C., Gulyaeva, A. A., Haagmans, B. L., Lauber, C., Leontovich, A. M., Neuman, B. W., Penzar, D., Perlman, S., Poon, L. L. M., Samborskiy, D. V., Sidorov, I. A., Sola, I., Ziebuhr, J., and Coronaviridae Study Group of the International Committee on Taxonomy of Viruses. The species Severe acute respiratory syndrome-related coronavirus: classifying 2019-nCoV and naming it SARS-CoV-2. *Nature Microbiology*, 5(4), April 2020. ISSN 2058-5276. doi:10.1038/s41564-020-0695-z.

Gore, S., Sanz García, E., Hendrickx, P. M., Gutmanas, A., Westbrook, J. D., Yang, H., Feng, Z., Baskaran, K., Berrisford, J. M., Hudson, B. P., Ikegawa, Y., Kobayashi, N., Lawson, C. L., Mading, S., Mak, L., Mukhopadhyay, A., Oldfield, T. J., Patwardhan, A., Peisach, E., Sahni, G., Sekharan, M. R., Sen, S., Shao, C., Smart, O. S., Ulrich, E. L., Yamashita, R., Quesada, M., Young, J. Y., Nakamura, H., Markley, J. L., Berman, H. M., Burley, S. K., Velankar, S., and Kleywegt, G. J. Validation of structures in the Protein Data Bank. *Structure(London, England:1993)*, 25(12):1916–1927, December 2017. ISSN 0969-2126. doi:10.1016/j.str.2017.10.009.

Harris, C. R., Millman, K. J., Van Der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J. *et al.* Array programming with NumPy. *Nature*, 585(7825):357–362, 2020.

Hay, B. P. Methods for molecular mechanics modeling of coordination compounds. *Coordination Chemistry Reviews*, 126(1-2):177–236, 1993.

Held, A. and Nagan, M. Tutorial 8: Parametrization of a non-standard residue using Antechamber. https://ambermd.org/tutorials/basic/tutorial8/index.php, 2021. Accessed: 2024-08-16.

Hess, B., Bekker, H., Berendsen, H. J., and Fraaije, J. G. LINCS: a linear constraint solver for molecular simulations. *Journal of Computational Chemistry*, 18(12):1463–1472, 1997.

Hinton, G. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

Hoover, W. G. Canonical dynamics: Equilibrium phase-space distributions. *Physical review A*, 31(3):1695, 1985.

Hu, Q., Xiong, Y., Zhu, G.-H., Zhang, Y.-N., Zhang, Y.-W., Huang, P., and Ge, G.-B. The SARS-CoV-2 main protease (Mpro): structure, function, and emerging therapies for COVID-19. *MedComm*, 3(3):e151, 2022.

Humphrey, W., Dalke, A., and Schulten, K. VMD: Visual Molecular Dynamics. *Journal of Molecular Graphics*, 14(1):33–38, 1996.

Ioffe, S. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Kecman, V. Support vector machines–an introduction. In *Support vector machines: theory and applications*, pages 1–47. Springer, 2005.

Khalili Yazdi, A., Li, F., Devkota, K., Perveen, S., Ghiabi, P., Hajian, T., Bolotokova, A., and Vedadi, M. A high-throughput radioactivity-based assay for screening SARS-CoV-2 nsp10-nsp16 complex. *SLAS Discovery*, 26(6):757–765, 2021. ISSN 2472-5552. doi:https://doi.org/10.1177/24725552211008863. Special Collection: Drug Discovery Targeting COVID-19.

Krafcikova, P., Silhan, J., Nencka, R., and Boura, E. Structural analysis of the SARS-CoV-2 methyltransferase complex involved in RNA cap creation bound to sinefungin. *Nature Communications*, 11(1):3717, July 2020. ISSN 2041-1723. doi: 10.1038/s41467-020-17495-9.

Lemkul, J. A. From proteins to perturbed Hamiltonians: A suite of tutorials for the GROMACS-2018 molecular simulation package [article v1.0]. *Living Journal of Computational Molecular Science*, 1(1):5068, 2018. doi:10.33011/livecoms.1.1.5068.

Li, G., Hilgenfeld, R., Whitley, R., and De Clercq, E. Therapeutic strategies for COVID-19: progress and lessons learned. *Nature Reviews Drug Discovery*, 22(6):449–475, June 2023. ISSN 1474-1784. doi:10.1038/s41573-023-00672-y.

Li, P. and Merz, K. M. J. MCPB.py: A Python based Metal Center Parameter Builder. *Journal of Chemical Information and Modeling*, 56(4):599–604, April 2016. ISSN 1549-9596. doi:10.1021/acs.jcim.5b00674.

Li, P., Roberts, B. P., Chakravorty, D. K., and Merz Jr, K. M. Rational design of particle mesh Ewald compatible Lennard-Jones parameters for +2 metal cations in explicit solvent. *Journal of Chemical Theory and Computation*, 9(6):2733–2748, 2013.

Lin, S., Chen, H., Ye, F., Chen, Z., Yang, F., Zheng, Y., Cao, Y., Qiao, J., Yang, S., and Lu, G. Crystal structure of SARS-CoV-2 nsp10/nsp16 2'-O-methylase and its implication on antiviral drug design. *Signal Transduction and Targeted Therapy*, 5(1):1–4, July 2020. ISSN 2059-3635. doi:10.1038/s41392-020-00241-4.

Madhavi Sastry, G., Adzhigirey, M., Day, T., Annabhimoju, R., and Sherman, W. Protein and ligand preparation: parameters, protocols, and influence on virtual screening enrichments. *Journal of Computer-Aided Molecular Design*, 27:221–234, 2013.

Maier, J. A., Martinez, C., Kasavajhala, K., Wickstrom, L., Hauser, K. E., and Simmerling, C. ff14SB: improving the accuracy of protein side chain and backbone parameters from ff99SB. *Journal of Chemical Theory and Computation*, 11(8):3696–3713, 2015.

McGibbon, R. T., Beauchamp, K. A., Harrigan, M. P., Klein, C., Swails, J. M., Hernández, C. X., Schwantes, C. R., Wang, L.-P., Lane, T. J., and Pande, V. S. MDTraj: A modern open library for the analysis of molecular dynamics trajectories. *Biophysical Journal*, 109(8):1528 – 1532, 2015. doi:10.1016/j.bpj.2015.08.015.

McLaughlin Jr, R. N., Poelwijk, F. J., Raman, A., Gosal, W. S., and Ranganathan, R. The spatial architecture of protein function and adaptation. *Nature*, 491(7422):138–142, 2012.

Minasov, G., Shuvalova, L., Rosas-Lemus, M., Kiryukhina, O., Wiersum, G., Godzik, A., Jaroszewski, L., Stogios, P., Skarina, T., and Satchell, K. 1.80 Angstrom resolution crystal structure of NSP16-NSP10 complex from SARS-CoV-2. *Center for Structural Genomics of Infectious Diseases (CSGID)*, 2020.

Müller, A. C. and Guido, S. Introduction to Machine Learning with Python: A Guide for Data Scientists. "O'Reilly Media, Inc.", September 2016. ISBN 978-1-4493-6990-3.

Nosé, S. A unified formulation of the constant temperature molecular dynamics methods. *The Journal of Chemical Physics*, 81(1):511–519, 1984.

Okeke, C. J., Musyoka, T. M., Sheik Amamuddy, O., Barozi, V., and Tastan Bishop, O. Allosteric pockets and dynamic residue network hubs of falcipain 2 in mutations including those linked to artemisinin resistance. *Computational and Structural Biotechnology Journal*, 19:5647–5666, January 2021. ISSN 2001-0370. doi: 10.1016/j.csbj.2021.10.011.

O'Shea, K. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

Páll, S., Zhmurov, A., Bauer, P., Abraham, M., Lundborg, M., Gray, A., Hess, B., and Lindahl, E. Heterogeneous parallelization and acceleration of molecular dynamics simulations in gromacs. *The Journal of Chemical Physics*, 153(13), 2020.

Park, S. and Kwak, N. Analysis on the dropout effect in convolutional neural networks. In *Computer Vision–ACCV 2016: 13th Asian Conference on Computer Vision, Taipei, Taiwan, November 20-24, 2016, Revised Selected Papers, Part II 13*, pages 189–204. Springer, 2017.

Parrinello, M. and Rahman, A. Polymorphic transitions in single crystals: A new molecular dynamics method. *Journal of Applied Physics*, 52(12):7182–7190, 1981.

Penkler, D. L., Atilgan, C., and Tastan Bishop, O. Allosteric modulation of human Hsp90α conformational dynamics. *Journal of Chemical Information and Modeling*, 58(2):383–404, February 2018. ISSN 1549-9596. doi:10.1021/acs.jcim.7b00630.

Peters, M. B., Yang, Y., Wang, B., Fusti-Molnar, L., Weaver, M. N., and Merz Jr, K. M. Structural survey of zinc-containing proteins and development of the zinc AMBER force field (ZAFF). *Journal of Chemical Theory and Computation*, 6(9):2935–2947, 2010.

Pettersen, E. F., Goddard, T. D., Huang, C. C., Couch, G. S., Greenblatt, D. M., Meng, E. C., and Ferrin, T. E. UCSF Chimera — a visualization system for exploratory research and analysis. *Journal of Computational Chemistry*, 25(13):1605–1612, 2004.

Rapaport, D. C. The Art of Molecular Dynamics Simulation, page 11–43. Cambridge University Press, 2004.

Rawat, P., Sharma, D., Pandey, M., Prabakaran, R., and Gromiha, M. M. Understanding the mutational frequency in SARS-CoV-2 proteome using structural features. *Computers in Biology and Medicine*, 147:105708, August 2022. ISSN 0010-4825. doi:10.1016/j.compbiomed.2022.105708.

Rosas-Lemus, M., Minasov, G., Shuvalova, L., Inniss, N. L., Kiryukhina, O., Wiersum, G., Kim, Y., Jedrzejczak, R., Maltseva, N. I., Endres, M., Jaroszewski, L., Godzik, A., Joachimiak, A., and Satchell, K. J. F. The crystal structure of nsp10-nsp16 heterodimer from SARS-CoV-2 in complex with S-adenosylmethionine. *bioRxiv*, page 2020.04.17.047498, April 2020. doi:10.1101/2020.04.17.047498.

Saldivar-Espinoza, B., Macip, G., Garcia-Segura, P., Mestres-Truyol, J., Puigbò, P., Cereto-Massagué, A., Pujadas, G., and Garcia-Vallve, S. Prediction of recurrent mutations in SARS-CoV-2 using artificial neural networks. *International Journal of Molecular Sciences*, 23(23):14683, November 2022. ISSN 1422-0067. doi:10.3390/ijms232314683.

Schindewolf, C., Lokugamage, K., Vu, M. N., Johnson, B. A., Scharton, D., Plante, J. A., Kalveram, B., Crocquet-Valdes, P. A., Sotcheff, S., Jaworski, E., Alvarado, R. E., Debbink, K., Daugherty, M. D., Weaver, S. C., Routh, A. L., Walker, D. H., Plante, K. S., and Menachery, V. D. SARS-CoV-2 uses nonstructural protein 16 to evade restriction by IFIT1 and IFIT3. *Journal of Virology*, 97(2):e01532–22, February 2023. doi:10.1128/jvi.01532-22.

Schober, P., Boer, C., and Schwarte, L. A. Correlation coefficients: Appropriate use and interpretation. *Anesthesia & Analgesia*, 126(5):1763–1768, 2018.

Schrödinger, LLC. The PyMOL molecular graphics system, version 1.8, November 2015.

Schrödinger, L. Schrödinger Release 2022-1: Epik. New York, NY, 2022a.

Schrödinger, L. Schrödinger Release 2022-1: Maestro. New York, NY, 2022b.

Sheik Amamuddy, O., Afriyie Boateng, R., Barozi, V., Wavinya Nyamai, D., and Tastan Bishop, O. Novel dynamic residue network analysis approaches to study allosteric modulation: SARS-CoV-2 Mpro and its evolutionary mutations as a case study. *Computational and Structural Biotechnology Journal*, 19:6431–6455, January 2021a. ISSN 2001-0370. doi:10.1016/j.csbj.2021.11.016.

Sheik Amamuddy, O., Glenister, M., Tshabalala, T., and Tastan Bishop, O. MDM-TASK-web: MD-TASK and MODE-TASK web server for analyzing protein dynamics. *Computational and Structural Biotechnology Journal*, 19:5059–5071, September 2021b. ISSN 2001-0370. doi:10.1016/j.csbj.2021.08.043.

Shu, Y. and McCauley, J. GISAID: Global initiative on sharing all influenza data - from vision to reality. *Eurosurveillance*, 22(13):30494, 2017.

Sk, M. F., Jonniya, N. A., Roy, R., Poddar, S., and Kar, P. Computational investigation of structural dynamics of SARS-CoV-2 methyltransferase-stimulatory factor heterodimer nsp16/nsp10 bound to the cofactor SAM. *Frontiers in Molecular Biosciences*, 7, November 2020. ISSN 2296-889X. doi:10.3389/fmolb.2020.590165.

Sousa da Silva, A. W. and Vranken, W. F. ACPYPE - Antechamber Python parser interface. *BMC research notes*, 5:1–8, 2012.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Tian, C., Kasavajhala, K., Belfon, K. A., Raguette, L., Huang, H., Migues, A. N., Bickel, J., Wang, Y., Pincay, J., Wu, Q. *et al.* ff19SB: Amino-acid-specific protein backbone parameters trained against quantum mechanics energy surfaces in solution. *Journal of Chemical Theory and Computation*, 16(1):528–552, 2019.

UniProt Consortium, T. UniProt: the universal protein knowledgebase. *Nucleic acids research*, 46(5):2699–2699, 2018.

Vakalopoulou, M., Christodoulidis, S., Burgos, N., Colliot, O., and Lepetit, V. Deep learning: basics and convolutional neural networks (CNNs). *Machine Learning for Brain Disorders*, pages 77–115, 2023.

Vanommeslaeghe, K., Guvench, O., and MacKerell, A. D. Molecular Mechanics. *Current Pharmaceutical Design*, 20(20):3281–3292, 2014. ISSN 1381-6128.

Viswanathan, T., Arya, S., Chan, S.-H., Qi, S., Dai, N., Misra, A., Park, J.-G., Oladunni, F., Kovalskyy, D., Hromas, R. A., Martinez-Sobrido, L., and Gupta, Y. K. Structural basis of RNA cap modification by SARS-CoV-2. *Nature Communications*, 11(1):3718, July 2020. ISSN 2041-1723. doi:10.1038/s41467-020-17496-8.

Wang, J., Wang, W., Kollman, P. A., and Case, D. A. Antechamber: an accessory software package for molecular mechanical calculations. *J. Am. Chem. Soc*, 222(1):2001, 2001.

Wang, J., Wolf, R. M., Caldwell, J. W., Kollman, P. A., and Case, D. A. Development and testing of a general AMBER force field. *Journal of Computational Chemistry*, 25(9):1157–1174, 2004.

Worldwide Protein Data Bank. Validation: Frequently Asked Questions (FAQs). 2024. Accessed: 2024-07-03.
URL https://www.wwpdb.org/validation/FAQs#RSRZ

Wu, J. Introduction to convolutional neural networks. *National Key Lab for Novel Software Technology. Nanjing University. China*, 5(23):495, 2017.

Yamashita, R., Nishio, M., Do, R. K. G., and Togashi, K. Convolutional neural networks: An overview and application in radiology. *Insights into Imaging*, 9:611–629, 2018.

Yan, S. and Wu, G. Application of neural network to predict mutations in proteins from influenza A viruses - A review of our approaches with implication for predicting mutations in coronaviruses. *Journal of Physics: Conference Series*, 1682(1):012019, November 2020a. ISSN 1742-6596. doi:10.1088/1742-6596/1682/1/012019.

Yan, S. and Wu, G. Prediction of mutations in H7 hemagglutinins from influenza A virus. *Journal of Biomedical Science and Engineering*, 13:175–186, 2020b. doi: 10.4236/jbise.2020.138017.

Yang, H. and Rao, Z. Structural biology of SARS-CoV-2 and implications for therapeutic development. *Nature Reviews Microbiology*, 19(11):685–700, 2021.

Zafar, A., Aamir, M., Mohd Nawi, N., Arshad, A., Riaz, S., Alruban, A., Dutta, A. K., and Almotairi, S. A comparison of pooling methods for convolutional neural networks. *Applied Sciences*, 12(17):8643, 2022.

Zhang, H., Ju, Z., Zhang, J., Li, X., Xiao, H., Chen, X., Li, Y., Wang, X., and Wei, Y. Exploring pathogenic mutation in allosteric proteins: The prediction and beyond. *bioRxiv*, page 2024.03, 2024. doi:10.1101/2024.03.20.486383.

Appendix A

Additional PDB Validation Results

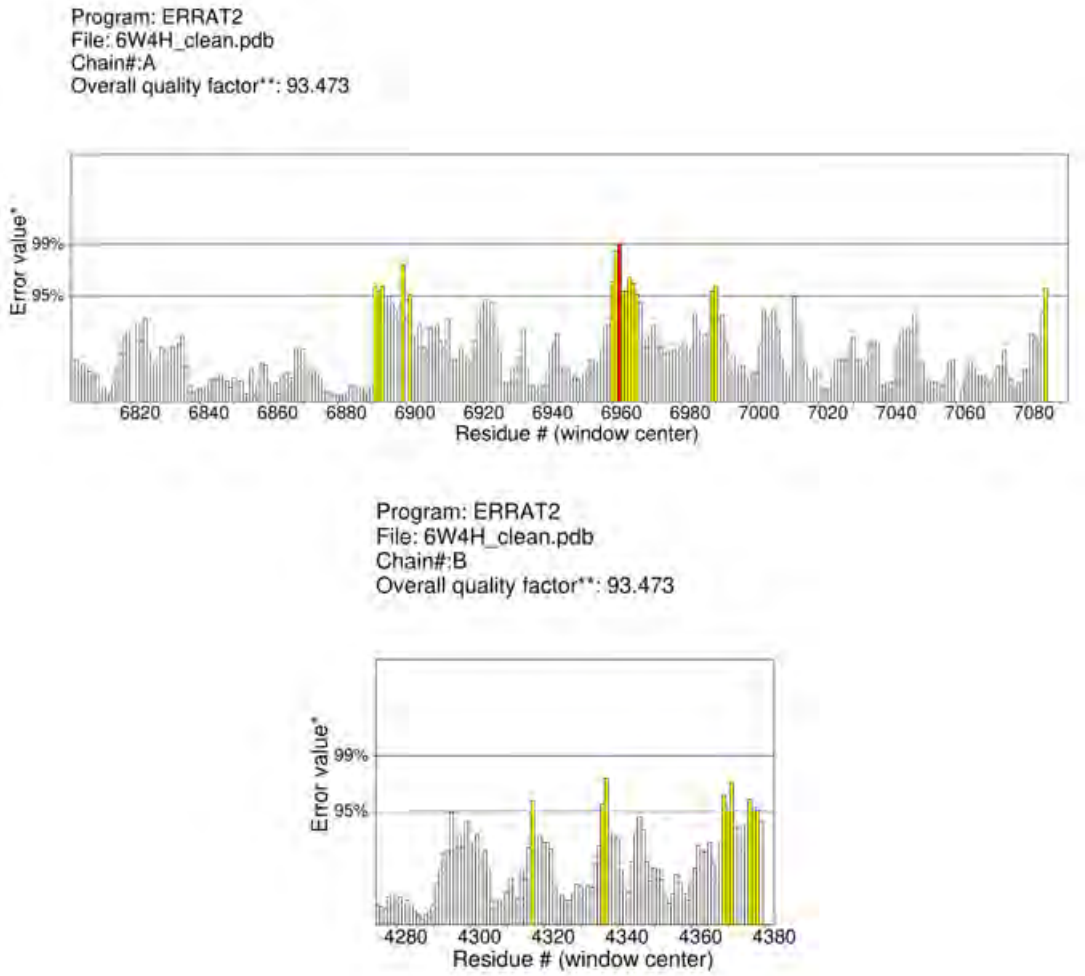

Figure A.1: ERRAT plots of the NSP10-NSP16 complex structure with PDB ID 6W4H. The two plots are for Chain A (NSP16) and Chain B (NSP10).

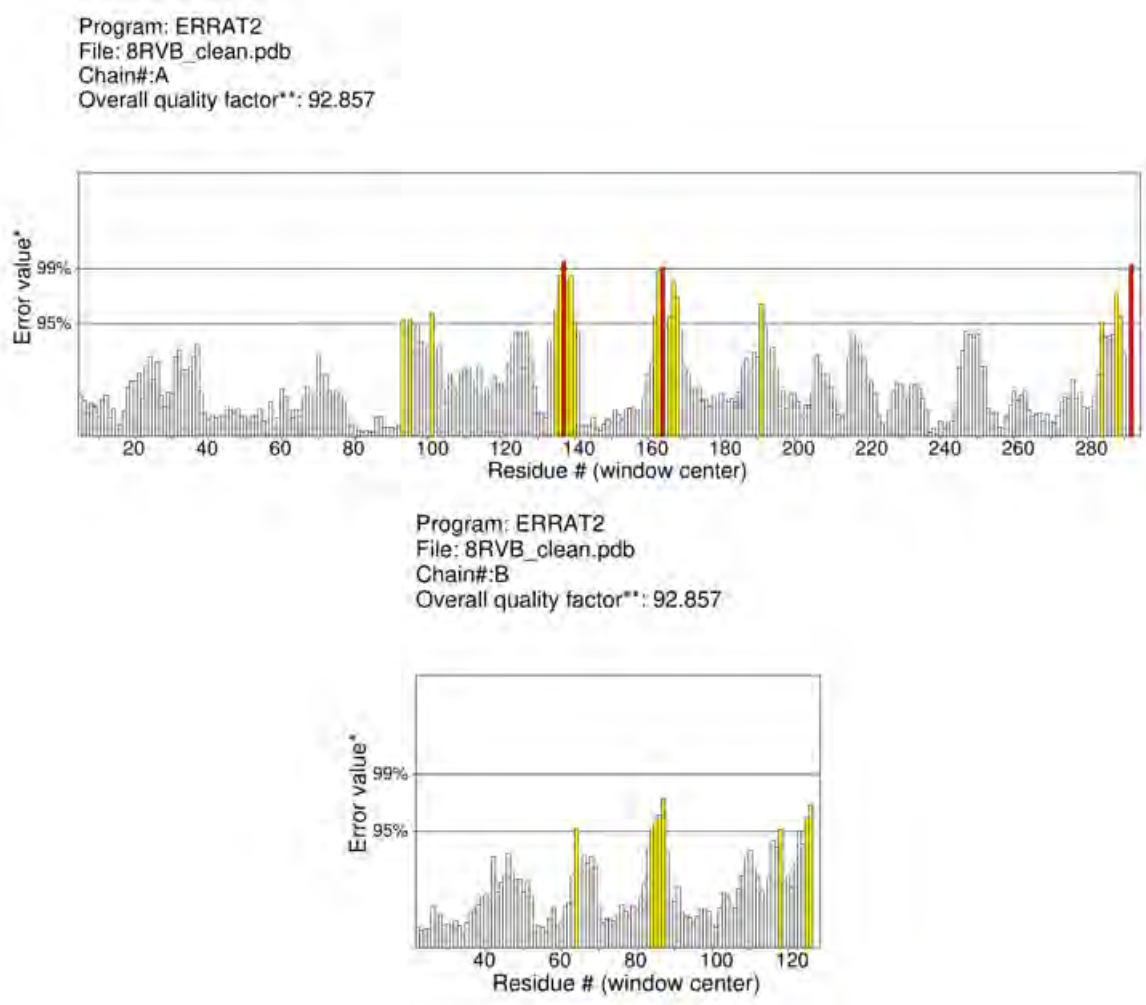

Figure A.2: ERRAT plots of the NSP10-NSP16 complex structure with PDB ID 8RVB. The two plots are for Chain A (NSP16) and Chain B (NSP10).

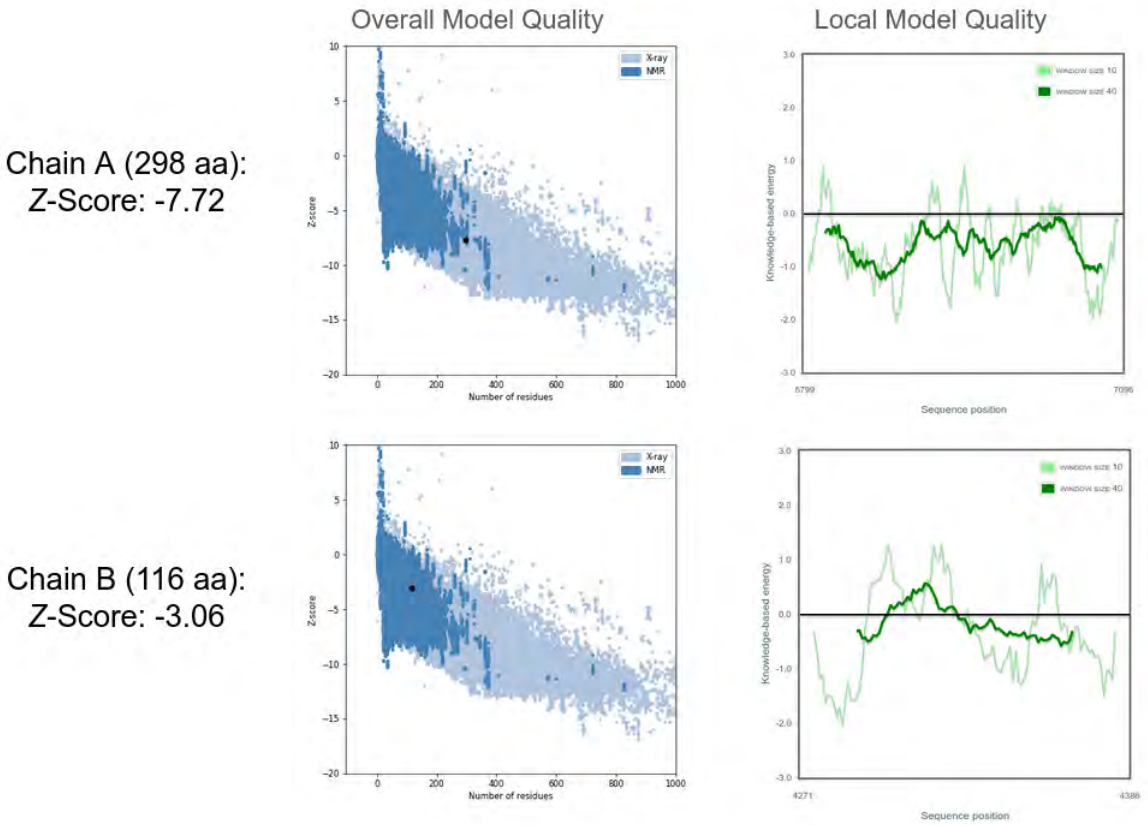

Figure A.3: ProSA plots for the NSP10-NSP16 complex structure (PDB ID: 6W4H). The top two plots display the Z-score analysis for Chain A (NSP16), providing insights into the overall and local model quality relative to structures of similar size in the PDB database. The bottom two plots show the corresponding Z-score analysis for Chain B (NSP10).

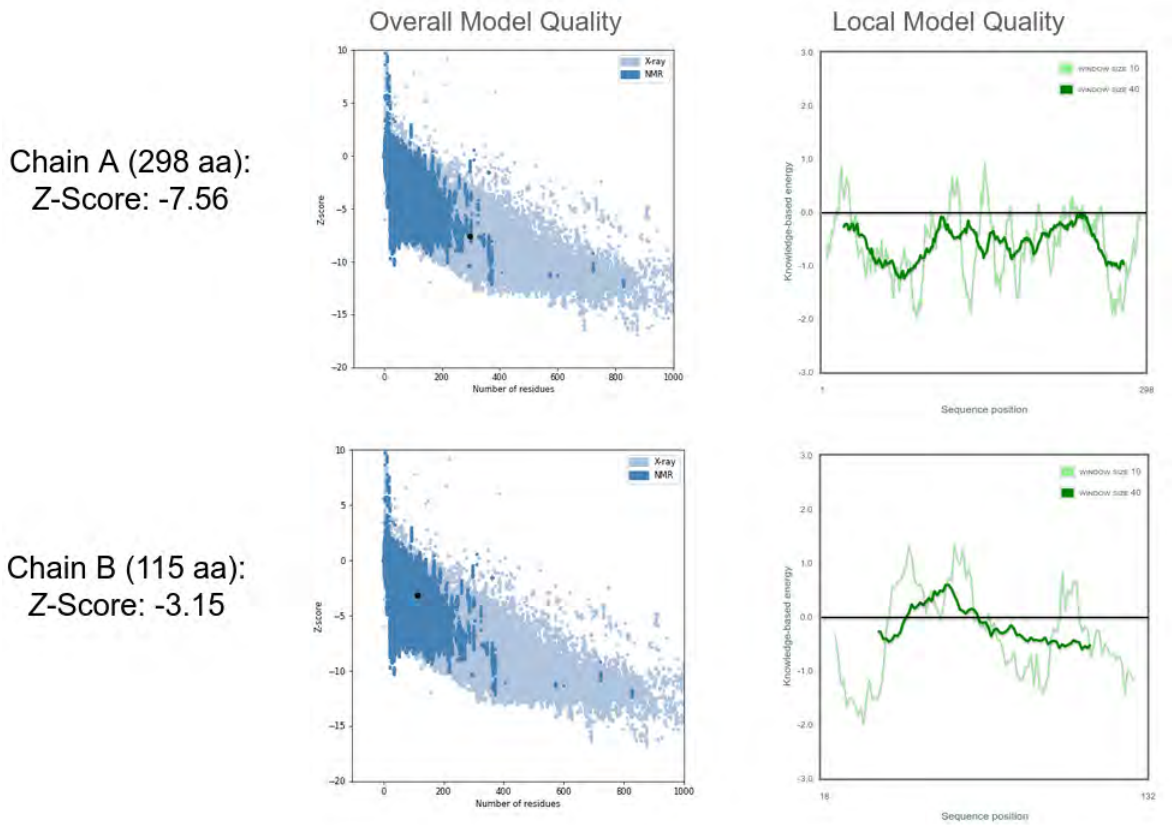

Figure A.4: ProSA plots for the NSP10-NSP16 complex structure (PDB ID: 8RVB). The top two plots display the Z-score analysis for Chain A (NSP16), providing insights into the overall and local model quality relative to structures of similar size in the PDB database. The bottom two plots show the corresponding Z-score analysis for Chain B (NSP10).