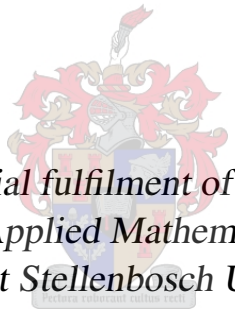
The Impact of Peptide Flanking Residues on Predicting Peptide-MHC-II Binding Interactions using Convolutional Neural Networks

by

Tshenolo T E Daumas



Thesis presented in partial fulfilment of the requirements for the degree of Master of Applied Mathematics in the Faculty of Science at Stellenbosch University

Supervisors: Dr. B. Bah

Dr. A M. Degoot

Prof. W. Ndifon

April 2022

The financial assistance of the Department of Science and Innovation and the Council for Scientific and Industrial Research DSI-CSIR Interbursary Programme towards this research is hereby acknowledged. Opinions expressed and conclusions arrived at, are those of the author and are not necessarily to be attributed to the neither the DSI nor the CSIR.

Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: April 2022 .....

Abstract

The Impact of Peptide Flanking Residues on Predicting Peptide-MHC-II Binding Interactions using Convolutional Neural Networks

TTE. Daumas

Department of Mathematical Sciences,
University of Stellenbosch,
Private Bag X1, Matieland 7602, South Africa.

Thesis: MSc (Applied Mathematics)

April 2022

Major histocompatibility complex class II (MHC-II) is one of three classes of MHC molecules and is located on the surface of professional antigen presenting cells. MHC-II molecules present antigenic peptides derived from pathogens that cause infection, for recognition by $CD4^+$ T lymphocytes. MHC-II molecules are critical components of the chain of intercellular interactions required for the adaptive immune response to be launched successfully, as this chain is thought to begin with the binding of antigenic peptides by MHC-II molecules.

While considerable progress in computational efforts have been made towards understanding peptide-MHC interactions for classes I and II, the case for peptide-MHC-II remains challenging due to MHC-II molecules being highly polymorphic and having open-ended binding grooves. Consequently, MHC-II molecules interact with peptides of varying lengths; therefore, the role that peptide flanking residues (PFRs) play in peptide-MHC-II binding interactions must be considered. We propose an allele-specific convolutional neural network model that simulates binding interactions between peptides and MHC-II molecules that also incorporates PFR information in the input.

Deep learning models for peptide-MHC-II interactions that have been published, such as the allele-specific model, NetMHCII and the transallelic model NetMHCIIpan have demonstrated encouraging predictive performance. When compared, our proposed CNN model outperformed the latest version of the model, NetMHCII-2.3 across all MHC-II alleles considered with mean AUC value of 0.951 as compared

with 0.822 for NetMHCII-3.2. Furthermore, we analysed the impact that PFRs have on modelling peptide-MHC-II binding interactions and laid the foundations of developing a transallelic model based on the CNN model.

Keywords: peptide-MHC-II, binding, convolutional neural networks, peptide flanking residues.

Uittreksel

Die Impak van Peptied Weerskante Oorblyfsels op die Voorspelling van Peptied-MHC-II Bindende Interaksies met Behulp van Konvolusionele Neurale Netwerke

("The Impact of Peptide Flanking Residues on Predicting Peptide-MHC-II Binding Interactions using Convolutional Neural Networks")

TTE. Daumas
Departement van Wiskundige Wetenskappe,
Universiteit van Stellenbosch,
Privaatsak X1, Matieland 7602, Suid Afrika.

Tesis: MSc (Toegepaste Wiskunde)
April 2022

Groot histoversoenbaarheidskompleks klas II (MHC-II) is een van drie klasse van MHC molekules en is geleë op die oppervlak van professionele antigeen-presenterende selle. MHC-II molekules bied antigeniese peptiede aan wat afkomstig is van patogene wat infeksie veroorsaak, vir herkenning deur $CD4^+$ T limfosiete. MHC-II molekules is kritieke komponente van die ketting van intersellulêre interaksies wat nodig is vir die aanpasbare immuunrespons om suksesvol van stapel te stuur, aangesien hierdie ketting vermoedelik begin met die binding van antigeniese peptiede deur MHC-II molekules.

Terwyl aansienlike vordering in berekeningspogings gemaak is om peptied-MHC interaksies vir klasse I en II te verstaan, bly die saak vir peptied-MHC-II uitdagend as gevolg van MHC-II-molekules wat hoogs polimorf is en oop-einde bindingsgroewe het. Gevolglik, MHC-II molekules interaksie met peptiede van verskillende lengtes; daarom, moet die rol wat peptied flankerende residue (PFRs) speel in peptied-MHC-II bindende interaksies oorweeg word. Ons stel 'n alleel-spesifieke konvolusionele neurale netwerk model voor wat bindingsinteraksies tussen peptiede en MHC-II molekules simuleer wat ook PFR-inligting in die inset inkorporeer.

Diep leer modelle vir peptied-MHC-II interaksies wat gepubliseer is, soos die alleelspesifieke model, NetMHCII en die transalleliese model, NetMHCIIpan het bemoedigende voorspellende prestasie getoon. As dit vergelyk word, het ons voorge-

stelde CNN-model beter gevaar as die nuutste weergawe van die model, NetMHCII-2.3 oor alle MHC-II allele wat oorweeg word met gemiddelde AUC waarde van 0,951 in vergelyking met 0,822 vir NetMHCII-3.2. Verder het ons die impak wat PFRe het op die modellering van peptied-MHC-II bindingsinteraksies ontleed en die grondslag gelê van die ontwikkeling van 'n transalleliese model gebaseer op die CNN-model.

Sleutelwoorde: peptied-MHC-II, binding, konvolusionele neurale netwerke, peptied flankerende residue.

Acknowledgements

I would like to express my sincere gratitude to the following people and organisations:

My family, friends, and colleagues for the support and confidence they expressed in me to complete and perform well in my studies.

My supervisors: Dr Bah, Dr Degoot and Prof. Ndifon for going beyond the academics to support me ensuring success in pursuing my MSc degree. Your dedication, patience and guidance has humbled me and I am grateful to you for all you have given me in the years you have helped me find my way.

The DSI-CSIR Interbursary Programme, the African Institute for Mathematical Sciences, South Africa, and Quantum Leap Africa- African Institute for Mathematical Sciences, Rwanda, for the resources to complete my project.

Most importantly, I could not have persevered through this without my faith in The Almighty. All praise is due to The Most High for divine favour.

Contents

List of Figures

List of Tables

Chapter 1

Background

1.1 Introduction

Computational immunology is the study of an organism's capacity to protect itself against infection on a cellular and molecular level using computational methods, such as machine learning models (Bahrami *et al.*, 2019; Tomar, 2020). Computational techniques offer advantageous alternatives and supplement conventional biochemical experiments by leveraging the availability of the enormous volumes of immunological data made available by advancements in protein sequencing technologies.

The goal of this thesis is to investigate the interactions between major histocompatibility complex (MHC) molecules of the human leukocyte antigen (HLA) system and antigenic peptides. There are three classes of MHC molecules in humans: MHC class I, MHC class II, and MHC class III. The primary function of MHC classes I and II is to present antigens at the cell surface for recognition by T cells in order to elicit an immune response upon pathogen invasion of the host organism. We are particularly interested in understanding peptide-MHC-II interactions due to the structural and compositional challenges that MHC-II molecules pose. They are highly polymorphic, they exist in multitudinous allele variants, and they have open-ended binding sites, which increases the amount of peptides with which they can bind in varying conformations (Zhang *et al.*, 2012; Liu *et al.*, 2019), resulting in a dynamic problem to explore. The study of peptide-MHC-II binding interactions is important to understanding the underlying mechanisms of adaptive immunity and can also assist in the development of effective peptide-based vaccines and design of immunotherapies (Lin *et al.*, 2008).

Conventional experimental approaches towards understanding peptide-MHC-II interactions are not scalable and are becoming increasingly arduous due to their labour, time and resource intensive nature. Computational methods offer alternative and complementary approaches to studying biological problems. Notably, machine

learning has made substantial strides in the area of computational immunology in recent years with methods such as the allele-specific artificial neural network (ANN) based NN-align (Nielsen and Lund, 2009) and the NetMHCII (Jensen *et al.*, 2018) models, and the transallelic ANN based NetMHCIIpan (Jensen *et al.*, 2018; Reynisson *et al.*, 2020), and long-short term memory (LSTM) recurrent neural network (RNN) based MHCnuggets (Shao *et al.*, 2020) and DeepSeqPanII (Liu *et al.*, 2021) models.

This thesis presents a binary classification convolutional neural network (CNN) model that predicts the binding of peptides and MHC-II molecules. The model is trained using peptide-MHC-II interaction data from MHC-II molecules of one of the three loci of the HLA system, the HLA-DR locus. Our CNN model also accounts for information about peptide flanking residues (PFRs), which are the excess portions of peptides beyond the areas that interact with the binding site of the MHC-II molecule. The consideration of PFRs further allows for the analysis of influences that the entire peptide has on the binding interactions between itself and the MHC-II molecule and how it can be used to improve the prediction power of models that simulate these interactions.

The layout of the thesis is as follows: Chapter 1 provides the background divided into a description of the biological context of the study in Section 1.2 and the computational tools used, namely, convolutional neural networks in Section 1.3. Chapter 2 deals with model formulation; the dataset used as input to train and test the model along with the development of the CNN's architecture are described. The results of the allele-specific model are reported and discussed in Chapter 3 and the development of the transallelic model is discussed in Chapter 4. Finally, Chapter 5 is the summary and conclusion of the study; furthermore, future works are also outlined in this chapter.

1.2 The Biology

An organism's immunity is divided into the innate and the adaptive immune system. The innate immune system is active from birth and is an organism's initial defence against infection. Innate immunity is non-specific and comprises a first line of defence that is external defences, such as physical barriers including the skin and mucus, and a second line of defence that is internal defences such as stomach acid and cellular defence mechanisms. The innate immune response works to identify and neutralize potentially harmful agents that threaten the organism and has no memory (Gleichmann, 2020). The adaptive immune response works to targetedly eliminate threats, and prevent and suppress effects of reinfection.

The adaptive immune system continually develops over the lifetime of an organism. It is characterized by specific responses that mediate a protective action against an

initial threat of a potentially harmful agent defined as a pathogen. The adaptive immune system can be divided into two: active and passive immunity. Active immunity is characterized by the production of antibodies from within the host organism and is effective long-term as it has memory. Examples of this are from direct infection and vaccination. Passive immunity can be maternal or artificial and is short-term. It is when antibodies are introduced from outside the organism. Maternal passive immunity is acquired through breastfeeding or passed through the placenta from mother to child during gestation and artificial passive immunity is achieved by the injection of antibodies produced in animals and other organisms to produce antiserums such as snake antivenom (Gleichmann, 2020). A summary of the immune system is illustrated in Figure 1.1.

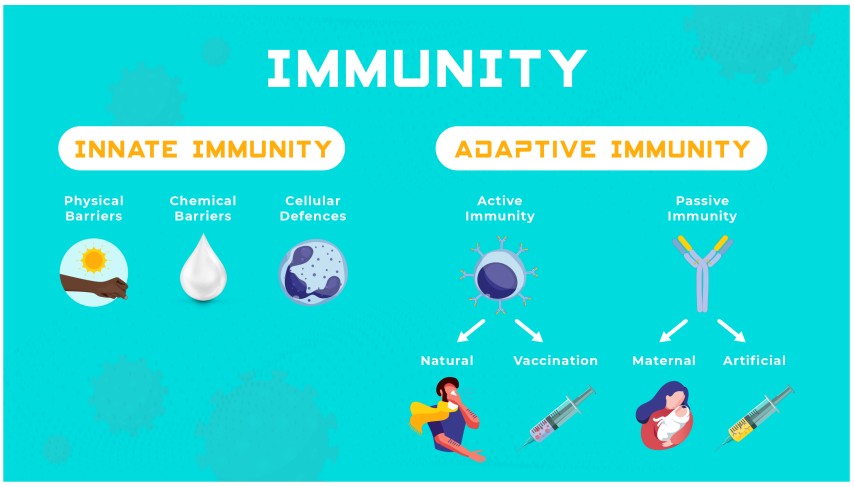


Figure 1.1: Innate and adaptive immune system summarized. Image taken from Gleichmann (2020).

Adaptive immune responses, particularly, resulting from active immunity, are stored in cellular memory and deployed again in subsequent invasions of the pathogen. MHC molecules are a component of the adaptive immune system that play a vital role in the initiation of the immune response. MHC molecules interact with other components of the adaptive immune system, namely, lymphocytes known as T cells and B cells to process antigens resulting from pathogens and initiate immune responses, thereby maintaining the host organism's immunity. Together, the innate and adaptive immune systems work to protect against infection by fighting off disease-causing substances.

1.2.1 The Adaptive Immune System

Infection of a host organism is defined to be the invasion of pathogenic bacteria, fungi, viruses, parasites and other harmful microorganisms that have the potential to cause disease. In the event that innate immunity is ineffective in staving off the infection, the host organism's adaptive immune system takes over and reacts in a systematic way. Pathogens are broken down and antigens are produced, which can be proteins, peptides, lipids and other biomolecules. Antigens function to induce an immune response, by generating antibodies to fight off the infection.

MHC molecules are expressed on cell surfaces of an organism. There are two main classes of MHC molecules which differ in their structural make-up and functional expression pattern: MHC-I and MHC-II (Holland *et al.*, 2013). MHC class I are expressed on the surface of most nucleated cells, and they bind to peptides derived from endogenous cytosolic proteins. MHC class II are expressed mainly on the surface of antigen presenting cells (APCs) such as dendritic cells and macrophages (Degoot *et al.*, 2018), and they bind to peptides derived from exogenous proteins. MHC molecules present peptides at the surface of cells to be recognized by T cell receptors (TCRs), that are expressed on the surface of T cells, for T cell activation and eventually initiate the adaptive immune response.

When TCRs encounter antigenic peptides in complex with MHC molecules, T cells proliferate and differentiate into effector T cells to initiate an immune response. $CD8^+$ T cells recognize peptides in complex with MHC-I molecules. Effector $CD8^+$ T cells function to eliminate cells with intracellular antigenic peptides and cancerous cells with abnormal proteins (Kumar and McNerney, 2005; Rudolph *et al.*, 2006). MHC-II molecules present peptides for recognition by $CD4^+$ T cells. Effector $CD4^+$ T cells trigger the killing of pathogens and the production of antibodies which protect the host organism against future infection by the same pathogen (Murphy *et al.*, 2008; Holland *et al.*, 2013).

Effector T cells can also be regulatory T cells which function to regulate the immune response on the occasion that the immune response may cause damage. Also, T cells that remain long-term after the infection has been eliminated, and are activated by antigens and vaccination, differentiate into memory cells that are responsible for the lifelong immunity that characterizes the adaptive immune system (Murphy *et al.*, 2008). Memory cells convert into effector cells upon subsequent reinfections from pathogens that the immune system recognizes, thereby, speeding up and increasing the chance of the host organism's recovery from reinfection.

1.2.2 The MHC-II Molecule

Major histocompatibility complex class-II molecules are a class of large glycoproteins expressed on the surface membranes of professional APCs, eg., dendritic cells

and macrophages, that patrol the extracellular space for antigens resulting from exogenous proteins. In humans, they are encoded in the human leukocyte antigen (HLA) system comprising of three loci: HLA-DP, HLA-DQ and HLA-DR (Traherne, 2008). They play a decisive role in cell mediated immune responses. Their function is to bind to antigenic peptides and present them for recognition by TCRs of $CD4^+$ T cells, regarded as the first step in inducing the adaptive immune response (Degoot *et al.*, 2018; Bordner and Mittelmann, 2010). Understanding the interactions between MHC-II molecules and antigenic peptides is important for learning the mechanisms that underpin adaptive immunity and for the development of peptide-based vaccines and targeted immunotherapies (Lin *et al.*, 2008).

MHC-II molecules are highly polymorphic, thus, exist in many genetic variations called alleles (Nielsen *et al.*, 2010). MHC-II molecules are structurally heterodimeric; they are formed by two distinct polypeptide chains: α-chain and β-chain (Figure 1.2). These chains each have many genetic variations across all loci, except for the the α-chain of the HLA-DR locus which is monomorphic. The polymorphism of these chains are accountable for the highly polymorphic nature of the MHC-II molecule (Murphy *et al.*, 2008). MHC-II molecules also interact with peptides of a wide range of lengths due to the open-ended nature of the site where the MHC-II molecule binds with peptides, called the MHC-II binding groove. These factors all present complications in the development of predictive models for peptide-MHC-II binding interactions (Jensen *et al.*, 2018).

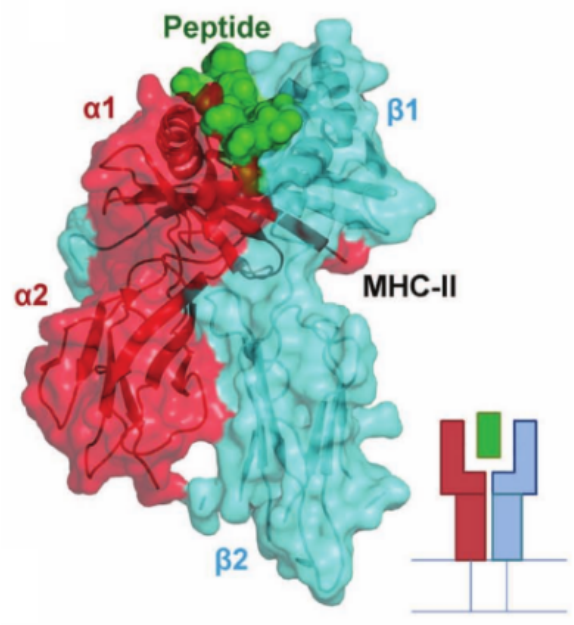


Figure 1.2: Representation of a peptide in complex with an MHC-II molecule. Image was taken from Holland *et al.* (2013).

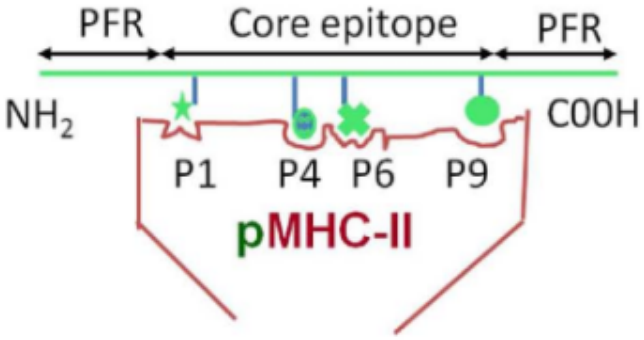


Figure 1.3: The peptide-MHC-II complex showing the characteristics that determine the binding interactions: The peptide epitope, PFRs, N and C peptide termini and the binding pockets of the MHC-II binding groove (Holland *et al.*, 2013).

The amino acid residues of the two chains make up the binding groove of the MHC-II molecule, which consists of nine binding pockets with primary anchor residues at pockets P1, P4, P6 and P9 (Zhang *et al.*, 2012). Peptides that interact with MHC-II molecules can be longer than 9 amino acid residues. The portion of the peptide that is directly engaged by the MHC-II binding groove is known as the peptide binding core or epitope. Peptides are short fragments of proteins and have two termini known as the N and C termini. The open-ended nature of the MHC-II binding groove (Figure 1.2) allows for portions of the peptide at either terminus that are not directly engaged by the MHC-II binding groove to protrude outwards. These are known as peptide flanking residues as shown in Figure 1.3. While peptide-MHC-II interactions are primarily determined by the binding core, it has been shown that PFRs also influence these interactions (Holland *et al.*, 2013). Each of these aspects of the peptides and of the MHC-II molecule are considered in the development of our model and further analysed and discussed in subsequent chapters.

1.3 The Convolutional Neural Network

Convolutional neural networks (CNNs) are a special class of feedforward artificial neural networks. Artificial neural networks, also known as ANNs or simply neural networks, simulate the operations of the human brain by allowing computer programs to recognize patterns and solve problems in the fields of artificial intelligence, machine learning and deep learning (IBM-Cloud-Education, 2020). Neural networks are characterized by layers, namely, an input layer, hidden layers and an output layer, each layer comprising a number of neurons (Figure 1.4).

Feedforward neural networks are ANNs in which data flows in one direction, from the input layer, through the intermediate hidden layers, to the output layer of the network. There are no feedback connections that allow the output to be fed back into the network (Goodfellow *et al.*, 2016). Neural networks that have this feedback mechanism are called recurrent neural networks and are out of the scope of this thesis.

A feedfoward neural network can be thought of as approximating a function of interest, say f^* by learning a set of parameters that optimizes the function approximation through an iterative training process that employs gradient-based learning.

Let us define a mapping,

$$y = f(x; \theta), \tag{1.1}$$

where x represents the input data, y, the output of the network and θ, the set of parameters that are learned by the neural network. Here, f can be defined as a composition of functions $f(x) = f^l(...f^2(f^1(x))...)$ representing l hidden layers of the neural network, where f^k, $k \in \{1, 2, ...l\}$, represents the k-th hidden layer.

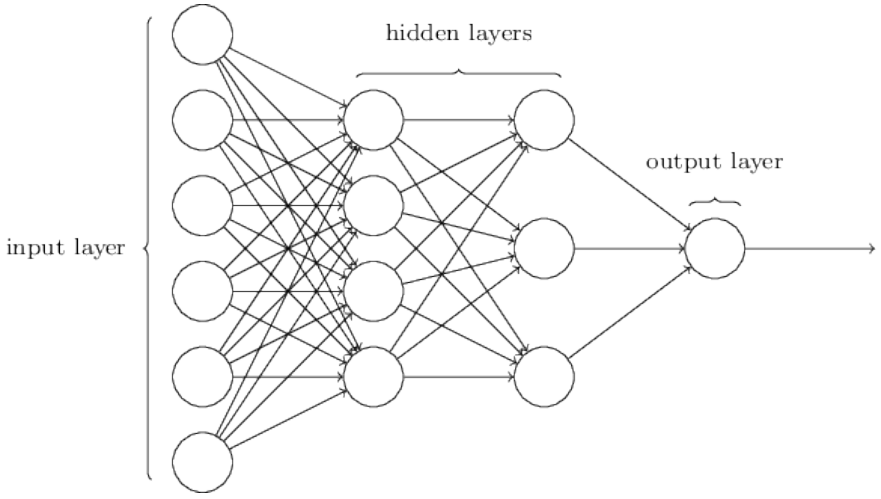


Figure 1.4: Artificial neural network with two hidden layers (Nielsen, 2015).

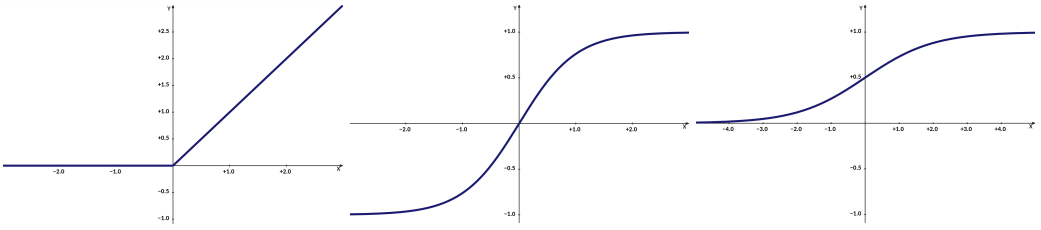
The input layer comprises neurons that initially present the input data, x in Equation 1.1, to the neural network. Each of the subsequent hidden layers, consists of neurons that are all connected to all the neurons in the preceding layer (Figure 1.6). Each neuron in a hidden layer calculates the weighted sum of all the neurons in the preceding layer and adds a bias term (Equation 1.2). An activation function is applied to the result and becomes the input of the neurons in the next layer of the neural network.

Activation is, in actual fact, a characteristic of ANNs; therefore it is inherited by CNNs. The activation function determines whether the result of a neuron is passed on further into the network, hence "activated". A neural network without activation functions is just the composition of linear functions as in Equation 1.2 and thus will act as a linear function itself, only performing linear transformations on the input. Activation functions introduce non-linearity in between layers to enable the network to handle complex tasks. Examples of popular activation functions are the Rectified Linear Unit (ReLU), the hyperbolic tangent function and the sigmoid function given by the equations below.

$$\phi(x) = \max\{0, x\} \qquad \text{(ReLU)}$$

$$\phi(x) = \tanh(x) = \frac{2}{1 + e^{-2x} - 1} \qquad \text{(Hyperbolic tangent)}$$

$$\phi(x) = \frac{1}{1 + e^{-x}} \qquad \text{(Sigmoid)}$$



(A) ReLU (B) Hyperbolic tangent (C) Sigmoid

Figure 1.5: Three examples of widely used activation functions in neural networks: the Rectified Linear Unit function (ReLU), the hyperbolic tangent function (tanh), and the sigmoid function.

The calculation that happens at each neuron in layers after the input layer is expressed in the following equations. For $j > 0$, let the $j-1$-th layer have m neurons, then the value of the i-th neuron in the j-th layer is calculated as follows:

$$x_{i,j} = \sum_{i=1}^{m} W_{i,j-1} z_{i,j-1} + b_{j-1} \tag{1.2}$$

$$z_{i,j} = \phi(x_{i,j}) \tag{1.3}$$

where $z_{i,j}$ is the value of the i-th neuron in layer j after the activation function ϕ is applied to the weighted sum $x_{i,j}$, $z_{i,j-1}$ is the value of the i-th neuron in the $j-1$-th layer, $W_{i,j-1}$ is the weight associated with the i-th neuron in layer $j-1$ and b_{j-1} is the bias term.

Convolutional neural networks are specialized and unique to regular ANNs because they have convolutional layers which allows them to successfully capture spatial

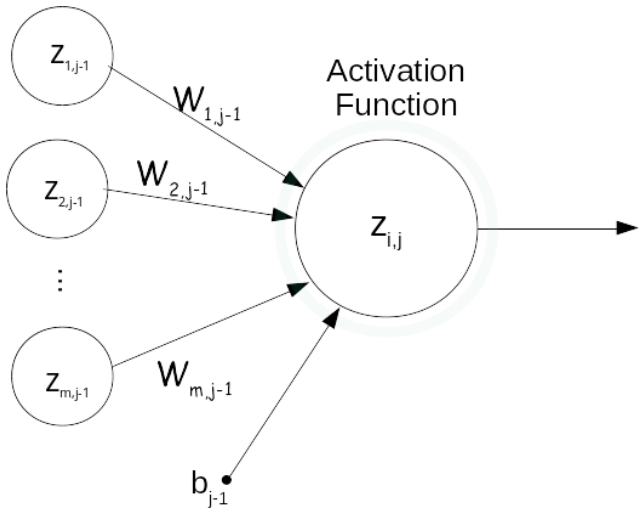


Figure 1.6: The i-th neuron of the j-th layer, $z_{i,j}$ in an ANN with associated weights, W's and bias, b, connected to the neurons z's from the previous layer.

and temporal dependencies of an input. The architecture of the CNN is hierarchical in structure and focusses on low-level features of the input in the earlier layers, then assembles them into higher-level features in subsequent layers of the network (Géron, 2017). The structure of CNNs thus makes them best suited for tasks such as image recognition, voice recognition and natural language processing. Examples of inputs of a CNN are time-series data as one-dimensional vectors of samples taken at regular time intervals, and grayscale images as two-dimensional matrices of pixels. Even higher dimensions like three-dimensional RGB colour images and other tensors can be inputs of CNNs.

CNNs have a few advantages over regular ANNs for handling complex inputs with large sizes, namely, sparse connectivity, parameter sharing and translation invariance. These advantages are due to the convolutional layers of CNNs. There are three stages in convolutional layers, the convolution (Section 1.3.1), activation and pooling (Section 1.3.2). These stages all simplify complex inputs for faster processing and easier understanding of the input, leading to an efficient model, both in terms of computational expense and performance accuracy.

In regular ANNs, each neuron is connected to every neuron in the previous layer; therefore, the output of a neuron is a result of multiplying every neuron of the input and their associated weights plus biases. Sparse connectivity in CNNs means that layers are partially connected, that is, each neuron in a convolutional layer is connected to only a subset of neurons from the previous layer, known as its receptive field. The output of each neuron is obtained through a mathematical operation employed by CNNs called convolution. The advantage of sparse connectivity is that the parameters that need to be learned and stored are significantly reduced,

leading to reduced memory requirements and more efficient processing speeds for the neural network.

1.3.1 Convolution

Convolution requires filters, also known as convolutional kernels, of the dimension and size of the receptive field of the neurons. The filters are populated with the parameters that are to be learned. The filters are applied over equally sized segments of the input from the previous layer successively, to produce a convolved output called a convolved feature map. Figure 1.7 shows the process of a 2-dimensional convolution. The same filter is applied over the whole input, allowing for parameters to be learned once then repeatedly used; therefore, leveraging the parameter-sharing advantage of CNNs.

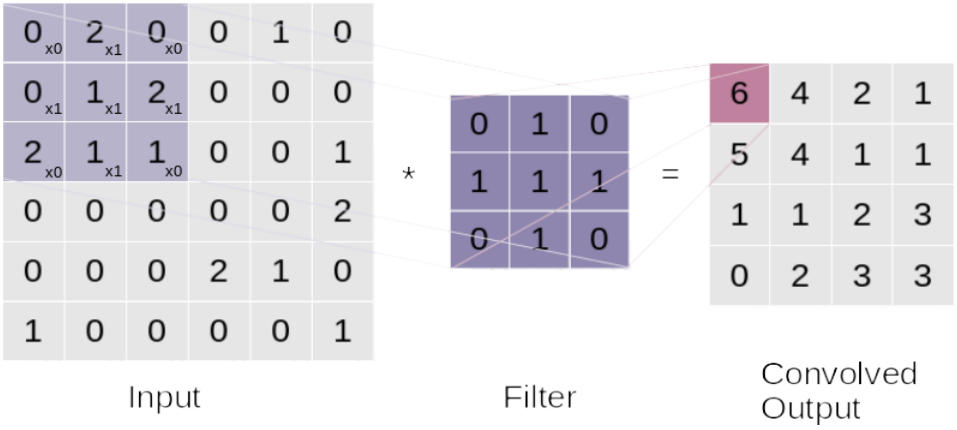


Figure 1.7: Example of a 2-dimensional convolutional operation with a filter of size 3x3

1.3.2 Pooling

Pooling is the process whereby the convolved feature map is downsampled by replacing consecutive equally sized segments of the convolved feature map with a summary statistic of that segment. There are two widely used ways to achieve pooling: max pooling where the summary statistic that replaces each segment is the maximum, and average pooling where the summary statistic is the mean (Figure 1.8). The summarization in pooling ignores positionality and is, therefore, responsible for making the neural network invariant to small translations. Thus, shifting the input does not change the output. Pooling also downsamples the feature map; therefore, reducing the size of the input to be processed in the next layer. This further reduces memory requirements for the storage of parameters and increases computational efficiency.

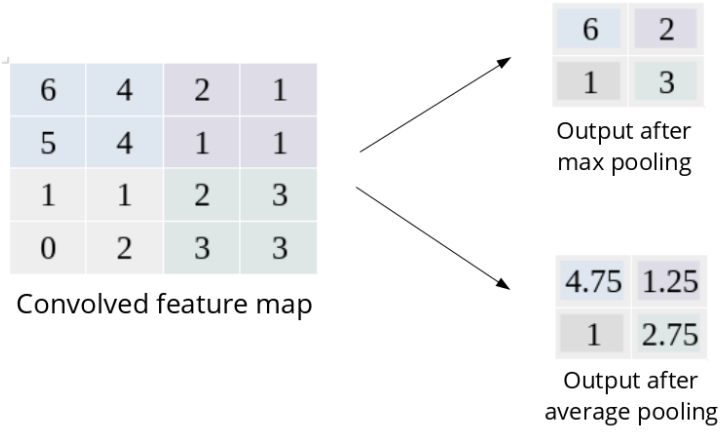


Figure 1.8: Max pooling and average pooling of the convolved output in the previous example (Figure 1.7) with pooling filter of size 2×2.

1.3.3 Backpropagation

A CNN can consist of multiple convolutional layers depending on the depth of the neural network. Following the convolutional layers, the output is usually flattened and connected to a fully connected layer in the same scheme as the regular ANN. The process thus far, from the input presented to the neural network and the computations performed in the hidden layers, results in an output presented by the output layer of the neural network. This process is called the *forward pass* of the backpropagation algorithm used in the training of neural networks.

Following the forward pass, the error between the output of the neural network, y in Equation 1.1, and the true value of the function, f^* associated with the input data x, is computed using a loss function. This is the beginning of the *backward pass* of backpropagation.

The cost function then averages the loss over the entire input dataset. It indicates how well the neural network models the data, the lower the loss the better the performance of the model. Examples of different cost functions are the Mean Square Error (MSE), Binary Cross Entropy, and the Multi-class Cross Entropy, and are

expressed in the equations below.

$$err = \frac{1}{N}\sum_{i=1}^{N}(y_i - y_i^*)^2 \quad \text{(MSE)}$$

$$err = -\frac{1}{N}\left[\sum_{i=1}^{N}[y_i^*(\log(y_i)) + (1 - y_i^*)(\log(1 - y_i))]\right] \quad \text{(Binary cross entropy)}$$

$$err = -\frac{1}{N}\left[\sum_{i=1}^{N}\left[\sum_{j=1}^{M} y_{i,j}^*(\log(y_{i,j}))\right]\right] \quad \text{(Multi-class cross entropy)}$$

In order to optimize the performance of the neural network the neural network's error is minimized with each training iteration by adjusting the network parameters. Gradients of the cost function with respect to each network parameter are computed starting from the last layer backwards towards the first layer. The gradients are then used in a Gradient Descent algorithm to update the network parameters for the next iteration in order to minimize the error. This completes the backward pass. This process is repeated until the neural network achieves sufficiently optimal performance.

1.3.4 Tasks of Neural Networks in Deep Learning

Neural networks are trained on a set of data to perform a variety of machine learning tasks. The tasks fall under two main categories: supervised and unsupervised learning. Other categories worth mentioning are semi-supervised and reinforcement learning. There are also interesting types of neural networks, such as autoencoders, that do not fit neatly into any one category of methods. Autoencoders are used for dimensionality reduction tasks and are unsupervised in that they do not require target labels for training; however, they are usually used for supervised tasks.

Supervised learning is the method of training a neural network to associate an input with an output using given output targets or labels. A neural network is fed a training set of examples, as input, along with their associated training labels, as the expected output. The network is trained on this dataset and is subsequently tested on a test set of additional examples without labels. The performance of the neural network is then measured using a variety of metrics, such as accuracy, precision and recall, by comparing its output with the set of test labels. Examples of supervised learning neural network tasks are regression and classification.

Unsupervised learning is where neural networks are trained without labels; the neural networks in this category learn underlying patterns in the data without "supervisor" intervention, and examples for unsupervised learning are clustering and anomaly detection. The training of ANNs in either category is an iterative process where the weights and biases are adjusted to progressively improve the network's performance of the task for which it was trained.

1.4 CNNs for Peptide-MHC Binding Prediction

CNNs have been used in peptide-MHC-I binding affinity prediction problems such as in the transallelic models: deep convolutional neural networks for pan-specific peptide-MHC class I binding prediction method (Han and Kim, 2017) and the DeepSeqPan method (Liu *et al.*, 2019), and also in the allele-specific DeepMHC method (Hu and Liu, 2017). DeepMHC was shown to be competitive with other existing neural network peptide-MHC-I binding interaction models and this has been attributed to its ability to automate learning of non-linear, high-order dependency among amino acid positions on the peptide, and its use of a large number of filters with small filter sizes (Hu and Liu, 2017).

There exists deep learning models that are applied in predicting peptide-MHC-II interactions. There are conventional feedforward artificial neural network models such as the allele-specific NN-align (Nielsen and Lund, 2009) and updated versions of NetMHCII (Jensen *et al.*, 2018) methods. The transallelic NetMHCIIpan (Jensen *et al.*, 2018; Reynisson *et al.*, 2020) method is also an example of models that use the ANN architecture. There are other deep learning models that predict peptide-MHC-II binding interactions based on RNNs, particularly, LSTMs. These are the transallelic methods: MHCnuggets (Shao *et al.*, 2020) and DeepSeqPanII (Liu *et al.*, 2021). However, to date, we have not been able to find any implementations of CNNs used for predicting peptide-MHC-II interactions.

Following, particularly, the success of the allele-specific DeepMHC method (Hu and Liu, 2017) with the advantages of its CNN architecture aforementioned for the peptide-MHC-I interactions we, therefore, propose a novel CNN model that simulates peptide-MHC-II binding interactions. Moreover, relying on the CNN's ability to capture locality and intricate relationships between input features by way of its convolutional layers, such as how the presence of one amino acid residue at one position can impact the presence of other amino acid residues in other positions on the peptide, further drive the use of CNNs in modelling peptide-MHC-II binding interactions. CNNs further allow for the analysis of the influence that PFRs have on peptide-MHC-II interactions by conformations of the input that make it easy to encode PFR information into the CNN input. The importance of PFRs in these interactions are demonstrated in this thesis and further confirm what has been published about PFR contribution to model performance like in Jensen *et al.* (2018).

We compare the performance of our allele-specific CNN model with the latest version of the state-of-the-art allele specific NetMHCII-2.3 (Jensen *et al.*, 2018) and show that it is competitive in its performance for research of the peptide-MHC-II interaction problem. We also discuss the development of a transallelic version of our CNN model for a more generalized prediction tool for peptide-MHC-II binding interactions and report preliminary results.

Chapter 2

Model Formulation

2.1 Introduction

We created a convolutional neural network model in Python 3.6, and trained and tested it using Google Colab (Google). It was developed using the Model class of the Keras framework, a high-level API (application programming interface) of the open source library for numerical computation and machine learning, Tensorflow 2.0.

In this chapter, we describe the methods and materials relating to the development of the CNN model. The datasets used for the input of the model along with where they were obtained are outlined in Section 2.2, and the methods involved in how the data was processed as input to the model are described in Section 2.3. The architecture of the model is discussed in Section 2.4 and the methods of evaluation are described in Section 2.5.

2.2 Peptide-MHC-II Interaction Data

The dataset utilized to train and test the CNN model was obtained from two databases: the Immuno Epitope Database for the peptide-MHC-II interaction dataset (IEDB) and the Immuno Polymorphism Database (IPD) for the protein sequences of the MHC-II alleles (Robinson *et al.*, 2012). The peptide-MHC-II interaction dataset was also used to train and test NetMHCII-2.3 and NetMHCIIpan-3.2, versions of the allele-specific and transallelic peptide-MHC-II prediction models, released by Jensen *et al.* (2018). The dataset is hosted and administered by the Technical University of Denmark Bioinformatics website (DTU-Bioinformatics).

Our CNN model was trained and tested using five fold cross validation. The dataset was split into five folds as described by Jensen *et al.* (2018) and Table 2.1 shows a sample of the composition of a $.csv$ file of data for one of the 5 folds.

Table 2.1: The IEDB information on peptide-MHC-II interactions included in the format of each of the five folds of training and test datasets.

peptide	affinity	molecule
PKYVKQNTLKLAT	0	HLA-DPA10103-DPB10201
DSDVGEFRAVTELG	0.047212	HLA-DPA10103-DPB10201
AAAAGWQTLSAALDA	0.23891	HLA-DPA10103-DPB10201
⋮	⋮	⋮

Each row describes a peptide and MHC-II allele pair as well as the associated log-transformed IC_{50} binding affinity value. This dataset is obtained from the IEDB and was also utilized in the training and testing of the NetMHCII-2.3 and NetMHCIIpan-3.2 models.

The first column of Table 2.1 is a list of peptide sequences. The second column is a list of log-transformed IC_{50} binding affinity values, these values are obtained by the log transformation of IC_{50} half-maximum inhibition concentration as described by Equation 2.1. The third column is a list of MHC-II allele identifiers of three loci of the human leukocyte antigen system, HLA-DP, HLA-DQ and HLA-DR and the H-2 locus of the mouse MHC-II alleles. Our CNN model is trained and tested only on HLA-DR data, the human MHC-II alleles of the DR locus.

$$\text{binding\_affinity} = 1 - \log(IC_{50})/\log(50000). \tag{2.1}$$

The protein sequences of the MHC-II alleles were also required for the CNN model input data processing, which were obtained from the IPD (Robinson *et al.*, 2012). Data was collected for 24 MHC-II alleles of the HLA-DR locus. Data from 16 of these 24 MHC-II alleles was used in our CNN model. The other alleles were excluded due to lack of information about those alleles and missing peptide-MHC-II binding data, which deemed them unsuitable for the training and testing of our model. The MHC-II allele data was collected in one $.csv$ file that has four columns: the first column is the index number for each row- each row containing information about one allele, the second column, with the heading, "Pre_name", is the MHC-II identifier, the third column, with the heading, "Allele_name", is the alleles' current names and the forth column is the protein sequences of the MHC-II alleles. Table 2.2 shows a sample of the composition of the HLA-DR MHC-II allele dataset $.csv$ file.

Our CNN model was built utilizing a total of 77167 datapoints encapsulating quantitative peptide-MHC-II binding interactions across 16 MHC-II alleles. Table 2.3

Table 2.2: Format of data file containing information about 24 HLA-DR MHC-II alleles obtained from the IPD.

	Pre_name	Allele_name	Sequence
1	DRB1_0101	DRB1*01:01:01	MVCLKLPGGSCMTAL...
2	DRB1_0301	DRB1*03:01:01:01	MVCLRLPGGSCMAVL...
⋮	⋮	⋮	⋮
24	DRB5_0101	DRB5*01:01:01	MVCLKLPGGSYMAKL...

Table showing the format of the MHC-II allele sequences file. Each row outlines the names, identifiers, names and protein sequences of 24 MHC-II alleles of the HLA-DR locus.

highlights the profile of the dataset used.

Table 2.3: Description of the peptide-MHC-II binding dataset used to develop the CNN model.

MHC-II molecule	#Peptides	#Binders	%Binders
DRB1_0101	10412	6376	61.24
DRB1_0301	5352	1457	27.22
DRB1_0401	6317	3022	47.84,
DRB1_0404	3657	1852	50.64
DRB1_0405	3962	1654	41.75
DRB1_0701	6325	3456	54.64
DRB1_0802	4465	2036	45.60
DRB1_0901	4318	2164	50.12
DRB1_1101	6045	2667	44.12
DRB1_1201	2384	759	31.84
DRB1_1302	4477	2249	50.23
DRB1_1501	4850	2107	43.44
DRB3_0101	4633	1415	30.54
DRB3_0301	884	510	57.69
DRB4_0101	3961	1540	38.88
DRB5_0101	5125	2430	47.41
Total	77,167	35,694	46.26

Table 2.3 presents information about the peptide-MHC-II binding data on which the CNN model was trained and tested. For each MHC-II allele, it shows the number of peptides belonging to the allele, the number and the percentage of binder peptides.

The first column of Table 2.3 lists the 16 MHC-II alleles under consideration. The second column shows how many peptides are utilized in the training and testing of the CNN model for each MHC-II allele. The third column shows the total number of peptides that bind to each MHC-II molecule, and the forth column shows the percentage of peptides that bind for each MHC-II allele.

2.3 Data Processing

MHC-II molecules consist of a binding groove where the binding interactions with peptides take place. The MHC-II binding groove consists of nine binding pockets which engage nine amino acid residues on the peptide called the peptide epitope or binding core. The amino acid residues on the peptide that do not engage the binding groove of the MHC-II molecule, and thus protrude outwards, are referred to as the peptide flanking residues (PFRs), see Figure 1.3.

Each binding pocket of the MHC-II binding groove comprises amino acid residues found at various positions on the MHC-II protein sequence. The CNN model is trained and tested on the HLA-DR locus of the MHC-II alleles whose α-chain are invariable across alleles. Therefore, the difference in composition of the MHC-II binding pockets is, due to the amino acid residues found on the β-chain of the MHC-II molecule. The positions on the protein sequence of the MHC-II β-chain, where the amino acid residues that make up each binding pocket are found, form a pseudo sequence of numbers. These pseudo sequences for each of the 9 binding pockets of the MHC-II binding groove are known (Karosiene *et al.*, 2013) and summarized in Table 2.4.

Table 2.4: Positions on the MHC-II protein sequence of the amino acid residues that make up each binding pocket of the MHC-II binding groove for all MHC-II alleles.

Pocket	Positions on MHC-II sequence
P1	9, 57, 85, 86, 89, 90
P2	74, 77, 78, 81
P3	47, 67, 71, 74, 78
P4	11, 13, 26, 28, 70, 71, 74, 78
P5	11, 13, 28, 30, 67, 70, 71, 74
P6	9, 11, 13, 28, 30, 71, 78
P7	11, 28, 30, 47, 67, 70, 71, 74, 78
P8	57, 67, 77, 78, 81
P9	9, 30, 57, 81, 85

Table showing the pseudo sequences of positions on the β-chain of the MHC-II molecule where amino acid residues responsible for the composition of each of the nine binding pockets of the MHC-II molecule's binding groove are found.

The first column of Table 2.4 is the list of the nine binding pockets of the MHC-II molecule's binding groove. The second column is the pseudo sequence of each binding pocket listing the positions on the MHC-II protein sequence where the amino acid residues that make up each binding pocket are located.

The input of the CNN model is processed using both the amino acids present in the peptide and the amino acids that appear in the positions described by the MHC-II

binding pocket pseudo sequences (Table 2.4). Each peptide sequence is subdivided into subsequences that are 9 amino acid residues long, referred to as peptide registers. The input of the CNN model is a 3D input tensor stacked along its depth with 2D matrices that encode the relationship between each successive peptide register along the peptide from the N-terminus. The 3D stacked input tensor with depth of n encapsulates information from n peptide registers, corresponding to $n - 1$ PFRs. The peptide register that the MHC-II binding groove engages is the peptide binding core or epitope. The CNN model was assessed over a range of peptide registers by varying the number of PFRs as the peptide binding core was not known before training.

One 2D matrix has the dimension 20×180 and it encodes one peptide register as it relates to the 9 MHC-II binding pockets. There are 20 possible amino acids that make up any protein. The rows of the matrix are the 20 possible amino acids, present in the MHC-II binding pockets. The columns of the matrix are the 20 possible amino acids present in the peptide register and their respective subscripted positions on the peptide register.

The method for processing a 2D matrix that encodes the relationship between one peptide register and MHC-II molecule is illustrated in Figure 2.1 and described as follows:

- Identify the amino acids in the 9 binding pockets of the MHC-II molecule.
- Identify the amino acids in the peptide register and subscript them with their positions on the peptide register.
- Populate all the columns in the 2D matrix that correspond to amino acid residues and their subscripted positions that do not occur in the peptide register with zeros.
- Follow the encoding scheme in the method described in Figure 2.1 to populate the 9 columns that remain that represent the peptide register.
- The rest of the other peptide registers are encoded in a similar fashion creating subsequent 2D matrices.

MHC-II sequence:

```
MVCLKLPGGS CMTALTVTLM VLSSPLALAG DTRPRFLWQL KFECHFFNGT
ERVRLLERCI YNQEESVRFD SDVGEYRAVT ELGRPDAEYW NSQKDLLEQR
RAAVDTYCRH NYGVGESFTV QRRVEPKVTV YPSKTQPLQH HNLLVCSVSG
FYPGSIEVRW FRNGQEEKAG VVSTGLIQNG DWTFQTLVML ETVPRSGEVY
TCQVEHPSVT SPLTVEWRAR SESAQSKMLS GVGGFVLGLL FLGAGLFIYF
RNQKGHSGLQ PTGFLS
```

Peptide sequence:

```
AIGIITLYLGAVVQA
```

9-mer peptide registers resulting from the peptide sequence:

AIGIITLYL, IGIITLYLG, GIITLYLGA, IITLYLGAV, ITLYLGAVV, TLYLGAVVQ, LYLGAVVQA

- Consider the first peptide register, **AIGIITLYL**, with amino acid, 'A', occurring in position 1.
- From Table 2.4, the pseudo sequence of positions on the MHC-II molecule that make up the first binding pocket, P1 is: [9, 57, 85, 86, 89, 90].
- The amino acids that correspond to this pseudo sequence are highlighted on the MHC-II sequence.
- The corresponding amino acids found in the first binging pocket are: [G,E,P,D,Y,W].
- Each amino acid occurs once so column A_1 in the matrix is populated with 1's at each amino acid.
- The process is done for rest of the register, i.e., amino acid, 'I', in position 2 and the second binding pocket, P2, used to populate column I_2 in the matrix, etc.
- The rest of the matrix is populated with zeros.

A_1: [G,E,P,D,Y,W] ,
I_2: [G,R,A,E],
G_3: [F,V,S,G,E],
I_4: [C,T,L,L,D,S,G,A],
I_5: [C,T,L,G,V,D,S,G],
T_6: [G,C,T,L,G,S,A],
L_7: [C,L,G,F,V,D,S,G,A],
Y_8: [E,V,R,A,E],
L_9: [G,G,E,E,P]

	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	...	G_3	...	I_1	I_2	I_3	I_4	I_5	...	L_7	L_8	L_9	...	T_6	...	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7	Y_8	Y_9
A														1		1			1				1									1	
C																1	1		1				1										
D	1															1	1		1														
E	1										1			1							2											2	
F											1								1														
G	1										1			1		1	2		2		2		2										
H																																	
I																																	
K																																	
L																2	1		1				1										
M																																	
N																																	
P	1																				1												
Q																																	
R														1																	1		
S											1					1	1		1				1										
T																1	1						1										
V											1						1															1	
W																																	
Y	1																																

Figure 2.1: Method of processing a 2D matrix that encodes an example of one peptide register (eg. AIGIITLYL) of a peptide sequence and an MHC-II allele sequence for the input.

We devised three ways to formulate the input for the CNN model, depicted in Figure 2.2 and tested which formulation yields the best results by varying the number of PFRs. Each input formulation is derived from a 3D input tensor, encoding an entire peptide, with the 2D matrices described in Figure 2.1 stacked along its depth, i.e., the third axis. Two of the input formulations, *stacked* and *aggregated*, use a slice of the first n matrices along the depth of the input tensor to assess the model for $n-1$ PFRs. The third input formulation uses only the n-th matrix in the depth of the input tensor (Figure 2.2).

The first way we derived the model input is the *stacked input formulation*. The relationship between a peptide and the MHC-II molecule is passed to the model as an input in the form of a 3D tensor of size $20\times180\times n$ for $n-1$ PFRs. The second way is the *aggregated input formulation* in which the input is a matrix of size $20\times180\times1$. This is the aggregation of the first n 2D matrices in the input tensor along its depth for $n-1$ PFRs. The *aggregated input formulation* summarizes the stacked input formulation at each number of PFRs.

The third way is obtained by using the n-th 2D matrix along the depth of the input tensor for $n-1$ PFRs. It is described as the *without-PFR input formulation* and the relationship between each peptide and MHC-II molecule is passed to the model as a matrix of size $20\times180\times1$. This input formulation also gives us the opportunity to investigate the effects of including PFR information in the input on the model performance as its results discount PFR contribution.

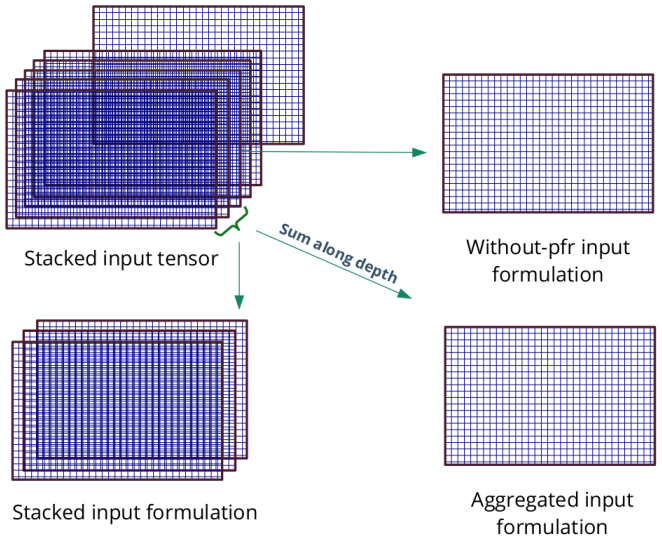


Figure 2.2: Depiction of the derivation of the three input formulations for one peptide: stacked, aggregated and without-PFR.

Furthermore, we considered processing the input to represent peptides-MHC-II binding interactions from the N-terminus, the C-terminus and the combined C and N termini where there is an equal likelihood that binding occurs from either terminus of the peptide. So far, the input data had been processed from the N-terminus. We generated the input tensors for each input terminus configuration as depicted in Figure 2.3.

The *C-terminus input configuration* was achieved by reversing the order of peptide registers being processed for the *N-terminus input configuration*. The last non-zero 2D matrix along the depth of each *N-terminus* input tensor, representing the last peptide register from the N-terminus of the peptide, thus, the first peptide register from the C-terminus, becomes the first 2D matrix along the depth of the *C-terminus* input tensor. The *Combined C and N termini input configuration* represents binding occurring from both termini of the peptide. It is obtained by the element-wise addition of the input tensor of the *N-terminus* and of the *C-terminus input configuration*.

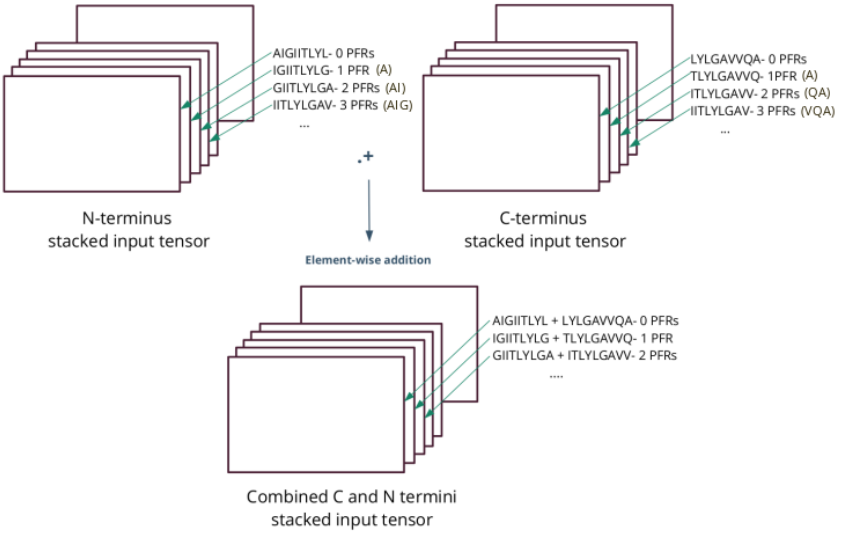


Figure 2.3: Illustration using the example peptide, AIGIITLYLGAVVQA, from Figure 2.1 to depict the N-terminus, C-terminus and Combined C and N termini input configurations.

The CNN model was trained on the *aggregated input formulation* for each of the terminus configurations and assessed for varying numbers of PFRs. The comparative study of the model performance results for the *C-terminus* and the *combined C and N termini input configurations* against the *N-terminus input configuration* are reported in Section 3.6.

The CNN model is a supervised learning binary classification model. Therefore, the input data includes associated target labels by which the model is trained. The

binding affinities in Table 2.1 are values between 0 and 1 that describe how strongly a peptide and an MHC-II molecule bond. The stronger bond, the larger the affinity value. For the CNN model, we are concerned with whether or not the peptide-MHC-II pair bind. As a result, we binarize these binding affinity values using the moderate binding threshold of 0.426 used by other peptide-MHC-II models too, e.g, NetMHCII. Thus, the target label for each input tensor encoding peptide-MHC-II binding interaction will either be 0 for the non-binding class or 1 for the binding class. The input tensors for each pair, regardless of whether they bind or not are processed the same way.

2.4 Model Architecture

Our model is a convolutional neural network that predicts whether or not a given peptide will bind with a given MHC-II molecule. The input dataset on which the CNN was trained and tested is a set of multidimensional arrays along with associated binary target labels (1 for "binders" and 0 for "non-binders") processed as in Section 2.3 which encode the relationship between peptide-MHC-II pairs. The architecture of our CNN model, illustrated in Figure 2.4, was attained through systematic and empirical experimentation by varying the model's hyperparameters. For the experiment, the CNN model was trained and tested on a subset of the input dataset and the conformation that optimized the model's performance was chosen for the architecture.

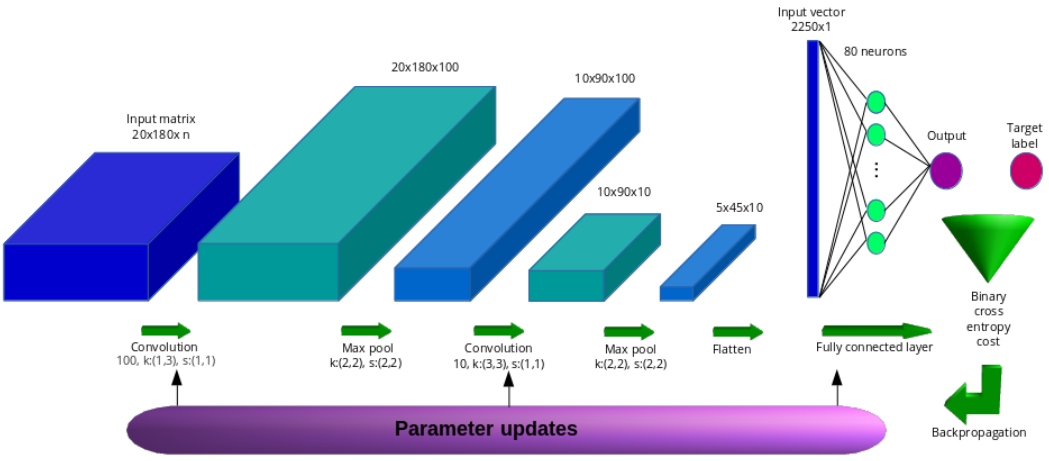


Figure 2.4: Binary classification convolutional neural network model architecture predicting peptide-MHC-II binding interactions.

Input layer. The input layer presents minibatches of 128 input matrices at a time to the CNN for the three different input formulations processed in the three terminus configurations. The stacked input matrix for each peptide-MHC-II pair is of the size $20 \times 180 \times n$ for the $n-1$-th PFR. The aggregated and without-PFR input matrices for each peptide-MHC-II pair are of the size $20 \times 180 \times 1$. The *aggregated input formulation* was used in the varying of the terminus configurations: the *N-terminus*, *C-terminus* and *combined C and N termini input configurations*.

Hidden layers. The number of hidden convolutional layers was varied and we found that the model's performance beyond 2 layers either deteriorated or did not improve significantly enough to warrant deepening the network further. We trained the network with a combination of number of neurons in each layer, varying the number between 10, 50, 100, 200, and 400 neurons. Furthermore, dimensions were also varied of the filters which are made up of the weights and biases, i.e., the network parameters to be learned by the CNN. The filters function to perform the convolution operation and detect spatial patterns in the input. We experimented with combinations of two-dimensional convolutional filter sizes between 3×1, 5×1, 7×1, 3×3, 5×5, and 7×7 in each layer. The best conformation of the hidden convolutional layers obtained was such that the first layer comprises 100 neurons, with filter size of 1×3 and stride size of $(1, 1)$ and the second layer comprises 10 neurons, with filter size of 3×3 and stride $(1, 1)$.

We chose ReLU activation for each convolutional layer due to its near-linear behaviour in order to avoid the vanishing gradient problem and to leverage its computational simplicity (Glorot *et al.*, 2011). There are no exponentials to compute in the ReLU activation function unlike in the case of the Hyperbolic tangent and Sigmoid activation functions. Each convolutional layer is followed it max pooling with filter size of 2×2 and stride size of $(2, 2)$ to downsample features of the input. The model seemed to be overfitting, primarily indicated by a considerable difference between training and test AUC scores. Hence, we included 50% dropout regularization between layers to mitigate the risk of model overfitting; consequently, this increased the test AUC scores, which was the desired outcome.

Fully connected layer. The output from the last hidden layer with dimension $5 \times 45 \times 10$ is flattened into a vector of size 2250×1. This vector is the input to a fully connected layer of 80 neurons also with ReLU activation. 80 neurons worked best out of varying the fully connected layer with 10, 60, 80, 100, 150 and 400 neurons. For the same reason as in the convolutional hidden layers, including 50% dropout regularization following the fully connected layer boosted the CNN model's performance and generalizability.

Output layer. The output layer has one neuron with Sigmoid activation. It generates a value $y \in [0, 1]$ which is the probability of predicting either class (peptide-MHC-II binding or not).

The CNN was compiled with the adam optimizer, learning rate of 0.001, and Binary cross entropy loss. The CNN model was trained using 5-fold cross-validation, where it was evaluated in terms of AUC-ROC (area under the receiver operating characteristic curve) values. The test scores were obtained by taking the average over the scores of the test sets in each of the 5 folds. The training was first carried out over a maximum of 50 epochs but later increased to 200 epochs with the patience parameter at 20 epochs. The significance of the patience parameter is that should the model not improve in 20 epochs, the model saves the network parameters learnt until that point then effects early-stopping which contributes to mitigating overfitting.

2.5 Evaluation Metrics

Evaluation metrics quantify how well the model performs; therefore, careful and justifiable consideration must go into the choice. We chose two metrics, area under the receiver operator characteristic curve (AUC-ROC) for model training, testing and for performance analysis, and accuracy only for model performance analysis.

Accuracy is a standard evaluation metric that works well for most problems. Our motivation for using accuracy was to determine the learning capacity of the model for each allele and at each number of PFRs. This is done by computing the baseline accuracy and comparing the model accuracy against the baseline accuracy. One major limitation of accuracy is that it is not robust against class imbalance (Yanminsun *et al.*, 2011) and the dataset we use as input to our CNN model is imbalanced for most MHC-II alleles; thus, can diminish the efficacy of accuracy as a metric. Consequently, accuracy is not sufficient as the sole evaluation metric for our CNN model.

The peptide-MHC-II binding data of an allele is balanced when it has 50% binder peptides and 50% non-binder peptides. For an arbitrary error margin of 1%, the data we used as input to the CNN model has the characteristic of class imbalance for 13 out of the 16 MHC-II alleles, also for the total dataset at 46.26% as in Table 2.3 in Section 2.2. This is a considerable level of class imbalance; thus influenced the choice of evaluation metrics.

The AUC-ROC is a metric commonly used in binary classification problems and reports on how well the model can distinguish between classes. The advantage of AUC-ROC is that it is a ranking metric; hence, does not make assumptions about class distribution (Fawcett, 2006). This quality of AUC-ROC makes it appropriate for datasets with class imbalance, as is the case for our input dataset. AUC-ROC is also used as an evaluation metric of other peptide-MHC-II binding prediction models in literature, which made it possible to compare our CNN model's performance

with the performance of other peptide-MHC-II binding prediction models, particularly, the NetMHCII-2.3 model (Jensen *et al.*, 2018).

The ROC curve represents the relationship between the false positive rate (FPR), and the true positive rate (TPR) also known as sensitivity or recall. The TPR signifies what percentage of the positives is correctly classified as positive and the FPR, what percentage of the negatives are erroneously classified as positive. Figure 2.5 is an example of one of the ROC curves from our CNN model and AUC scores achieved during training. A higher TPR and lower FPR is desired; therefore, the larger the area under the curve the better the performance of the model. The dotted line as in Figure 2.5 represents a purely random classifier. So, the aim is to get the ROC curve to be as steep as possible to improve model performance.

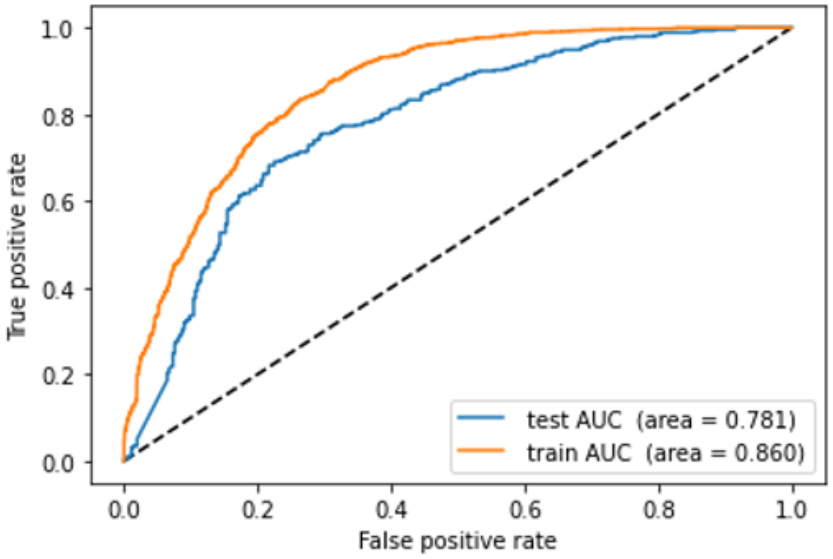


Figure 2.5: Example of training and test ROC curves with associated AUC values for one fold during cross validation training of our CNN model.

Chapter 3

Allele-specific Model

3.1 Introduction

We implemented a convolutional neural network model, described in Chapter 2, to predict the binding interactions of peptides and MHC-II molecules that encode the human leukocyte antigen (HLA) system in the DR locus. Our CNN model is allele-specific; that is, it was trained and tested on peptide-MHC-II binding data belonging to an individual allele at a time. We consider 16 MHC-II alleles in this study, and the number of peptides for each of the alleles range from 884 for allele *DRB3_0301* to 10412 for allele *DRB1_0101*. Table 2.3 in Section 2.2 summarizes the data we have utilized in this study.

In this chapter, we report on the results of allele-specific CNN based model we have developed to predict peptide-MHC-II binding interaction. We show its performance for 3 different formulations of the model input in Section 3.2 and systematically assess the effect of PFRs on model performance in terms of two performance metrics, AUC and accuracy in Section 3.3. In Sections 3.4 and 3.5, we analyse the effects of zero padding of the input tensors and discuss the implementation of a stopping criterion for the investigation of the number of PFRs to be included in the input of the model. We also analyse and compare the model performance when processing inputs from different peptide terminus configurations and report on the best results for each MHC-II allele in Section 3.6. Lastly, we compare the best result from our CNN model with the state-of-the art allele-specific model that predicts peptide-MHC-II binding interactions, the NetMHCII-2.3 method (Jensen *et al.*, 2018) in Section 3.7.

3.2 Input Formulation Investigation

We assess the model performance for each input formulation using two metrics: Area Under the Receiver Operating Characteristic Curve (AUC) values and Accuracy. The range of AUC values is 0 to 1 and the larger the AUC value, the better the performance of the model. The AUC metric has a number of advantages, one major advantage being that it is robust against imbalanced data (Fawcett, 2006). In the dataset we used as input for developing our CNN model, several alleles have imbalanced data, such as *DRB1_0301* and *DRB3_0101* with percentages of binder peptides at 27.22% and 30.54%. Additionally, reporting on the accuracy metric provides important model performance information like whether or not the model learns by comparing model accuracy against baseline accuracy.

Our CNN model's purpose is to predict the binding interactions between peptides and MHC-II molecules. It is imperative that the input along with the model itself represent the problem for which it is designed to find solutions well. A very important part of model development is optimization. For our CNN model we started with experimentally tuning the hyperparameters of the CNN in Chapter 2 Section 2.4. In this section we determine the best representation of the input to further optimize efficiency of the model by assessing the model performance for each input representation. Three input formulations were derived: *stacked*, *aggregated* and the *without-PFR* as described in Chapter 2 Section 2.3.

Figure 3.1 shows accuracy curves of the *stacked*, *aggregated* and *without-PFR input formulations* for each MHC-II allele over 0-14 PFRs. The plots include the baseline accuracy for each allele to show whether the model is learning or not. The model's learning capacity is indicated by its accuracy curve being above the baseline accuracy curve. Each of the 16 plots in Figure 3.1 outlines four curves of accuracy on the y-axis against the number of PFRs on the x-axis. The blue curve is for the baseline accuracy and the red, green, and cyan curves describe the model performance for the *stacked*, *aggregated* and *without-PFR input formulations*, respectively.

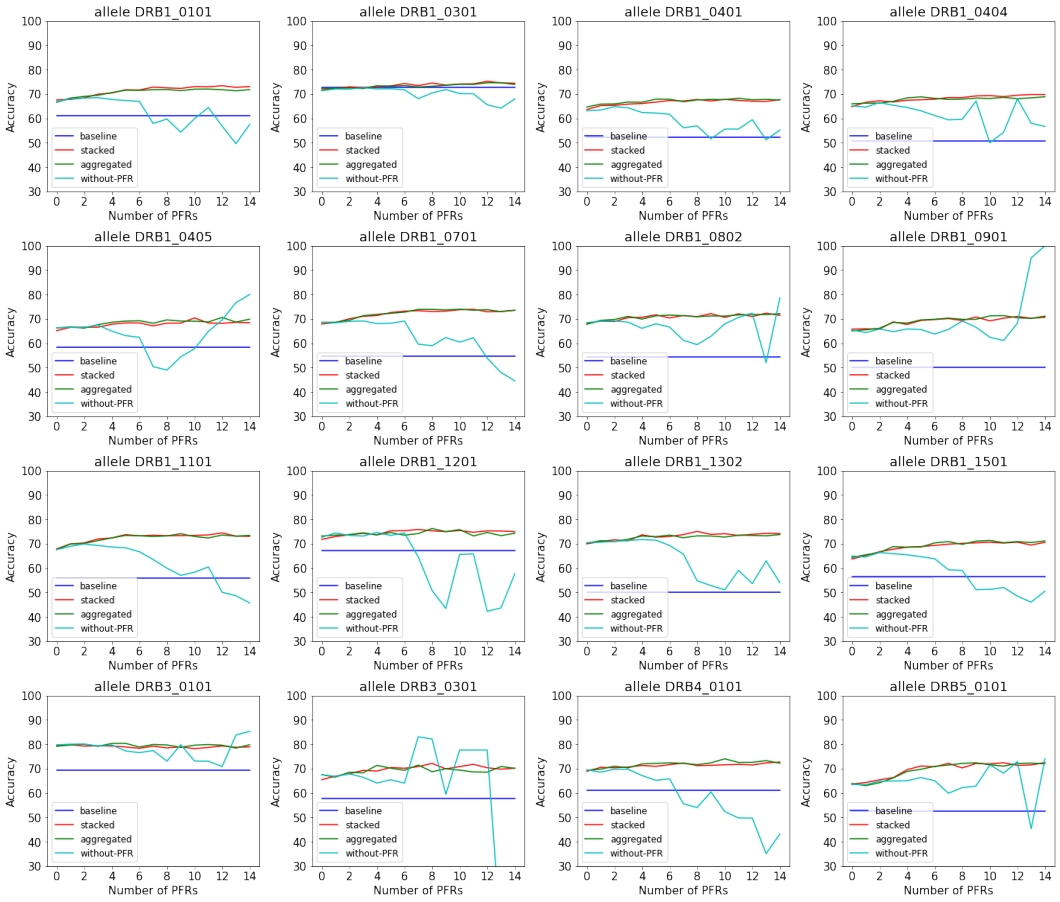


Figure 3.1: Model performance in terms of accuracy for the three input formulations for 16 MHC-II alleles considered in the study. The stacked and aggregated input formulations (red and green curves) show a steady and good performance while the without-PFR input formulation (cyan curve) shows an erratic trend with the weakest performance for most MHC-II alleles across 0-14 PFRs.

The stacked input formulation was implemented as input to our CNN model first. We noticed that the processing of the data into the stacked input formulation demanded time that could be improved upon. That is when we devised the aggregated input formulation to summarize the stacked input formulation. The aggregated input formulation sped up the process, however, it was important to determine if any significant amount of information was being lost in the aggregation. When comparing the model performance in terms of accuracy between the stacked and aggregated input formulations (Figure 3.1), a similar trend was observed across all alleles. The accuracy for these two input formulations is well above the baseline accuracy, which indicates that the model is learning when using both input formulations for all number of PFRs.

In Figure 3.2 we considered the difference in model performance in terms of AUC values for all MHC-II alleles across 0-14 PFRs. We measured the absolute difference of AUC values between the stacked and the aggregated input formulations and found it is near zero for each of the 16 MHC-II alleles, with the maximum absolute difference being under 0.025. This indicates that our CNN model architecture attains similar performance for both the stacked and the aggregated input formulations.

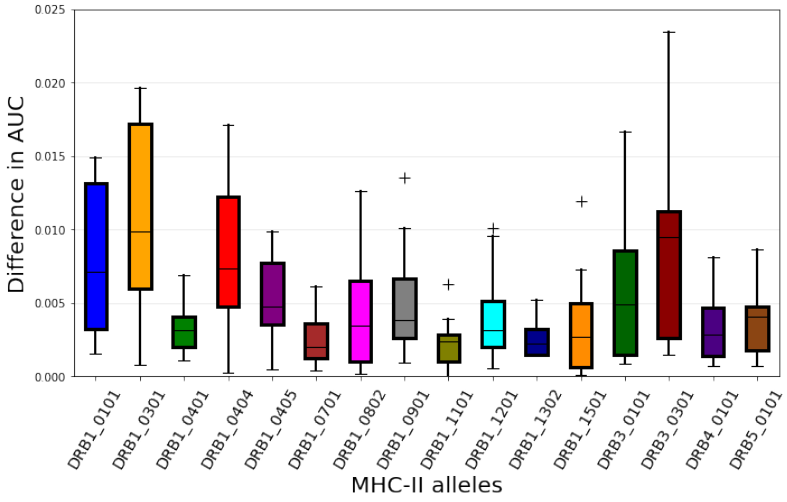


Figure 3.2: Absolute difference of AUC values between the stacked and aggregated input formulations for the 16 MHC-II alleles considered. The differences in model performance are small across MHC-II alleles and suggest that using either formulation yields similar model performance.

For more precise assessment of the difference in model performance in terms of AUC values between the stacked and the aggregated input formulations, we carried out a statistical test to find out whether the difference is significant. Table 3.1 outlines the descriptive statistics of the data along with the p-value results and their interpretation of the Wilcoxon Signed Rank test applied to the AUC value data for each of the 16 MHC-II alleles.

Table 3.1: Descriptive statistics and p-values comparing the performance, in terms of AUC values, of the stacked and aggregated input formulations for our CNN model.

Allele (DRB)	1_0101	1_0301	1_0401	1_0404	1_0405	1_0701	1_0802	1_0901	1_1101	1_1201	1_1302	1_1501	3_0101	3_0301	4_0101	5_0101
Stacked																
mean	0.771	0.741	0.757	0.755	0.766	0.819	0.803	0.774	0.814	0.824	0.835	0.774	0.812	0.768	0.789	0.784
median	0.784	0.754	0.765	0.757	0.771	0.829	0.806	0.783	0.823	0.832	0.840	0.783	0.817	0.777	0.795	0.801
stdev	0.025	0.024	0.014	0.015	0.013	0.019	0.012	0.022	0.020	0.016	0.014	0.024	0.011	0.023	0.014	0.033
SW p-value*	0.003	0.004	0.001	0.085	0.023	0.002	0.065	0.003	0.000	0.002	0.006	0.002	0.009	0.036	0.004	0.001
Aggregated																
mean	0.764	0.730	0.759	0.750	0.770	0.820	0.805	0.778	0.812	0.827	0.837	0.775	0.817	0.768	0.791	0.785
median	0.774	0.737	0.764	0.753	0.775	0.830	0.810	0.792	0.822	0.832	0.843	0.788	0.822	0.776	0.797	0.804
stdev	0.021	0.019	0.013	0.010	0.013	0.019	0.012	0.024	0.019	0.014	0.013	0.026	0.012	0.021	0.014	0.034
SW p-value*	0.001	0.008	0.003	0.014	0.005	0.002	0.003	0.001	0.000	0.003	0.007	0.001	0.003	0.001	0.001	0.001
WSR p-value**	0.002	0.000	0.973	0.015	0.991	0.884	0.970	0.999	0.020	0.987	0.996	0.833	0.999	0.715	0.990	0.695

* SW p-value: the p-value results of the Shapiro Wilk Test for normality.
** WSR p-value: the p-value results of the Wilcoxon Signed-Rank Test.

Table shows the summary statistics of the stacked and the aggregated input models. It also reports on the p-value result of the Wilcoxon Signed Rank test applied on the AUC difference values for each of the 16 MHC-II alleles to see if there is a significant difference in model performance between the two input formulations.

Each column of Table 3.1 presents each of the 16 MHC-II alleles considered in the study. The table is divided into three segments. The first and second segments have five rows each, of descriptive statistics for the stacked and aggregated input formulations, respectively. The five rows in each of the two segments list the mean, median, standard deviation, and the p-value for the Shapiro-Wilk test for normality at 95% confidence level for each input formulation. The third segment shows the results of the Wilcoxon Signed Rank test in terms of the p-value.

We compared AUC value differences between the stacked and aggregated input formulations per allele so the order of the number of PFRs had to be retained in each data sample. Furthermore, we could not assume normality for the data samples of either of the input formulations across all alleles except *DRB1_0404* and *DRB1_0802* for the stacked input formulation. Lastly, the data sample size was only 15 for each allele. Therefore, we chose the Wilcoxon Signed Rank test, because it is a paired non-parametric statistical test, therefore, does not assume any underlying distributions of the samples in the analysis, and it is also appropriate for data samples that are matched or paired and small in size (Glen, 2021).

The results of the Wilcoxon Signed Rank test (Table 3.1) are obtained at confidence level of 95%, which corresponds to the significance level of α= 0.05. We subtract the AUC values for the aggregated input formulation from the stacked input formulation. The goal is to test whether there is no statistical difference in model performance between the input formulations or if the stacked input formulation is better. Therefore we chose a one-sided test concerned with the difference in the medians of the data samples in the analysis.

The hypotheses of the one-sided Wilcoxon Signed Rank test as we implemented it are as follows:

$$H_0 : \text{The median difference is } 0,$$
$$H_a : \text{The median difference is } > 0.$$

We fail to reject the null hypothesis (H_0) for 12 out of the 16 MHC-II alleles indicated by a Wilcoxon Signed Rank p-value > 0.05. Thus, with 95% confidence, we can infer that the difference in medians is not significant for most of the alleles. This leads to the conclusion that the model utilizing the two input formulations yields similar results or better results for the aggregated input formulation in 12 out of the 16 MHC-II alleles considered.

Therefore, we favour the replacement of the stacked input formulation with the aggregated input formulation of the CNN model. The aggregated input formulation is the aggregation of the stacked input tensor along the depth; thus, it acts similarly

to the first convolutional layer on the stacked input model. The first convolution applied to the stacked input tensor can be replaced by aggregation done once in the data processing stage. By using the aggregated input formulation, we can save on computational costs and time demanded by the model training process.

3.3 The Role of PFRs in Binding Interactions

The third way we varied our input is by encoding only the peptide register corresponding to a particular number of PFRs while discounting any information of PFRs. We refer to this input variation as the *without-PFR* input formulation. The investigation of this input variation is to assess the role of PFRs on the performance of the model and to compare model performance with previous input variations that account for PFRs.

MHC-II molecules interact with peptides of various lengths. In the dataset that we use, the peptide length distribution range from 9 to 37 amino acid residues. As the number of PFRs increases, thereby taking peptide registers further and further along the peptide length, the shorter peptides fall away. This results in a loss of dataset size for the without-PFR input formulation. In the stacked input formulation, we compensate for the loss of data size by zero-padding along the depth of the input tensor for the shorter peptides. However, for the without-PFR model, this results in getting zero-matrices for peptide registers that correspond to larger numbers of PFRs. We remove these zero-matrices in the dataset for the without-PFR model.

Owing to the progressive loss of data in the without-PFR input model as the number of PFRs is increased, we can no longer use the same baseline accuracy as in Figure 3.1 to determine whether or not the model is still learning. Thus, we compute the baseline accuracy at each number of PFRs as the data size decreases and plot this against the without-PFR model accuracy for each allele in Figure 3.3. From the without-PFR baseline (in blue), we see that the data becomes highly imbalanced as we remove data points corresponding to the shorter peptides.

Figure 3.1 in Section 3.2 shows the performance of the CNN model in terms of accuracy for the without-PFR input formulation vs. the stacked and the aggregated input formulations. It is clear that the performance of the without-PFR input model rapidly and dramatically decreases compared to the other two input models across most of the alleles as the number of PFRs increases. The baseline accuracy in the plots in Figure 3.1 is that of a consistent dataset size for all numbers of PFRs. In Figure 3.3 the baseline accuracy is computed at each number of PFRs as the data size decreases with the increase of the number of PFRs.

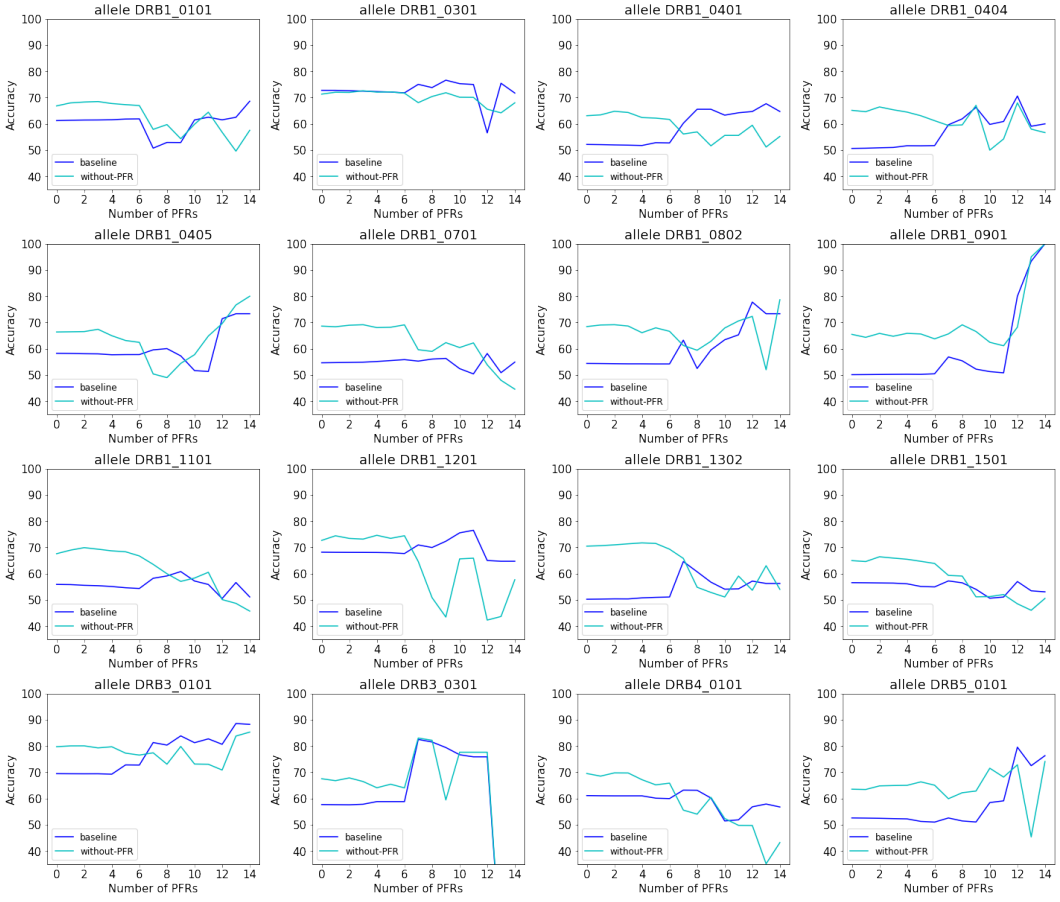


Figure 3.3: A comparison of the accuracy of the CNN model with the without-PFR input formulation and the corresponding baseline accuracy for each of the 16 MHC-II alleles. Model accuracy decreases to below baseline accuracy beyond a certain number of PFRs for most MHC-II alleles, which indicates weakness in the without-PFR input model as it loses its ability to learn within the range of PFRs in the study. This is an argument for including PFR information in the input.

The without-PFR input model's learning capacity diminishes for larger numbers of PFRs (Figure 3.3) across all alleles as the model accuracy curve dips below the baseline accuracy beyond a certain number of PFRs. This indicates that the learning capacity of the model deteriorates to the point that it is no longer learning as the number of PFRs increases.

Thus, it is evident that PFRs play an important role in determining the binding specificities of MHC-II molecules, as is indicated by the poor performance of the without-PFR input model compared to the stacked and aggregated input models which incorporate PFR information. Even with the considerations of the decrease in data size and the deterioration of the model's learning capacity beyond a certain number of PFRs for the without-PFR input formulation, we have shown the computational importance of including information of PFRs in the input in order to improve model performance.

3.4 Stopping Criterion for PFR Importance

PFRs play an important role in determining the binding specificity of MHC-II molecules. However, the degree of how much PFR information should be included in the model is not known beforehand. This section is devoted to investigating this. Figure 3.4 plots the proportion of dataset size at each number of PFRs ranging from 0 to 14 PFRs for the 16 MHC-II alleles considered. These curves shows that after 6 PFRs, which corresponds to peptide length of 15 amino acid residues or more, the data size of every considered allele decreases dramatically.

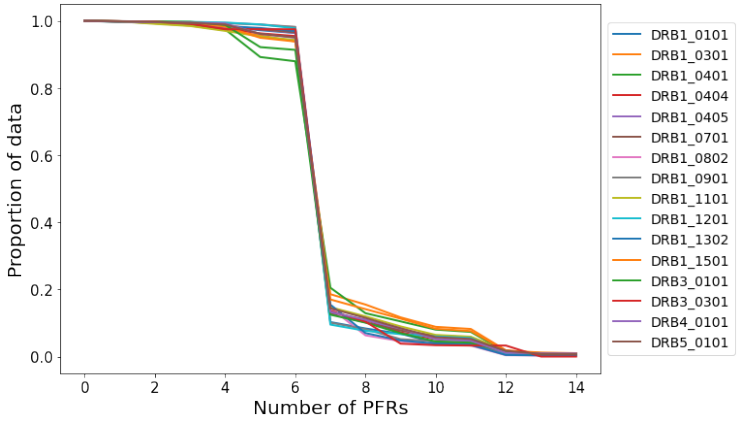


Figure 3.4: Proportion of the dataset size over various numbers of PFRs for each of the 16 MHC-II alleles. Beyond 6 PFRs, the data sample size dramatically decreases across alleles; therefore, this is can used as a stopping criterion for PFR investigation.

The point of the dramatic decrease in data size correlates with the approximate point where the stability of the baseline accuracy in Figure 3.3 also significantly deteriorates and the data becomes more imbalanced for most alleles as discussed in Section 3.3. The point of the dramatic decrease in data size also correlates with the point where there is a decline in the without-PFR input model's learning capacity. This, along with the loss of data itself leads to the indication that the data is unreliable after 6 PFRs and that 6 PFRs can be used as a stopping criterion for further model development.

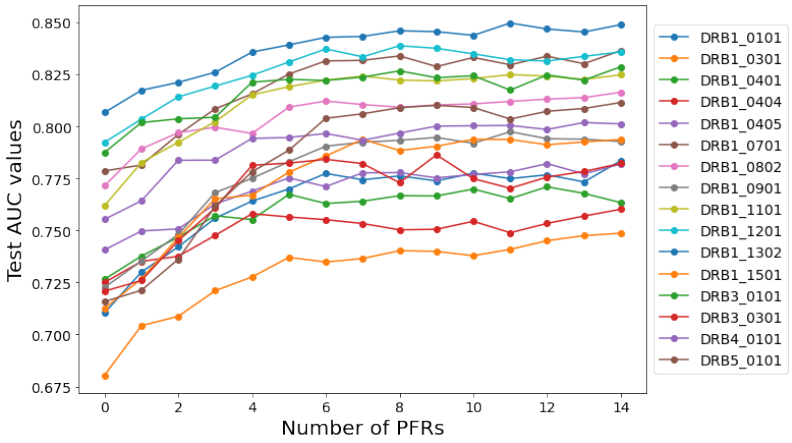


Figure 3.5: Performance results of the aggregated input model in terms of AUC curves over 0-14 PFRs for the 16 MHC-II alleles covered in the study. The model performance improves with the increase in number of PFRs across alleles and stabilizes beyond a certain number of PFRs. This point is the optimal number of PFRs to be considered in reporting model performance.

Figure 3.5 presents model performance in terms of AUC curves for the aggregate input formulation and Table 3.2 outlines the AUC values. The model performance improves with the increase in the number of PFRs, but performance improvement is more noticeable in the first few number of PFRs, up to approximately 6 PFRs for most of the alleles. While the performance continues to improve after 6 PFRs, that improvement progressively deteriorates to the point of diminishing returns for all MHC-II alleles. This further supports the recommendation to effect a stopping criterion of PFR investigation at 6 PFRs.

Table 3.2: AUC values for the stacked (S) and aggregated (A) input formulations for the 16 MHC-II alleles as obtained by the CNN model over 0 to 14 PFRs.

Allele \PFRs	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
DRB1_0101(S)	0.716	0.728	0.741	0.759	0.760	0.773	0.780	0.784	0.789	0.787	**0.791**	0.788	0.790	0.788	0.791
DRB1_0101(A)	0.710	0.730	0.742	0.756	0.764	0.770	0.777	0.774	0.776	0.774	0.778	0.775	0.777	0.773	**0.784**
DRB1_0301(S)	0.683	0.706	0.715	0.722	0.737	0.743	0.751	0.754	0.756	0.758	0.757	0.759	**0.762**	0.757	0.754
DRB1_0301(A)	0.680	0.704	0.709	0.721	0.728	0.737	0.735	0.737	0.740	0.740	0.738	0.741	0.745	0.748	**0.749**
DRB1_0401(S)	0.722	0.735	0.743	0.750	0.752	0.762	0.765	0.767	**0.769**	0.765	0.766	0.766	0.765	0.766	0.766
DRB1_0401(A)	0.727	0.738	0.747	0.757	0.755	0.767	0.763	0.764	0.767	0.767	0.770	0.765	**0.771**	0.768	0.763
DRB1_0404(S)	0.720	0.735	0.744	0.739	0.754	0.757	0.752	0.761	0.756	0.768	0.767	0.764	0.766	0.769	**0.771**
DRB1_0404(A)	0.725	0.735	0.738	0.748	0.758	0.756	0.755	0.753	0.750	0.751	0.754	0.749	0.753	0.757	**0.760**
DRB1_0405(S)	0.735	0.747	0.752	0.763	0.764	0.771	0.772	0.769	0.771	0.771	**0.785**	0.774	0.775	0.769	0.772
DRB1_0405(A)	0.741	0.750	0.751	0.763	0.769	0.775	0.771	0.778	0.778	0.775	0.777	0.778	**0.782**	0.777	0.782
DRB1_0701(S)	0.777	0.783	0.797	0.810	0.812	0.824	0.829	0.831	0.831	**0.835**	0.832	0.834	0.829	0.827	0.831
DRB1_0701(A)	0.779	0.781	0.796	0.808	0.816	0.825	0.831	0.832	0.834	0.829	0.833	0.830	0.834	0.830	**0.836**
DRB1_0802(S)	0.778	0.780	0.792	0.798	0.796	0.810	0.800	0.804	0.806	**0.816**	0.809	0.812	0.811	0.813	0.813
DRB1_0802(A)	0.771	0.789	0.797	0.800	0.797	0.809	0.812	0.811	0.809	0.810	0.811	0.812	0.813	0.814	**0.816**
DRB1_0901(S)	0.722	0.738	0.743	0.765	0.771	0.779	0.788	0.787	0.783	0.792	0.778	0.789	**0.793**	0.786	0.790
DRB1_0901(A)	0.723	0.735	0.748	0.768	0.775	0.783	0.790	0.792	0.793	0.795	0.792	**0.797**	0.794	0.794	0.793
DRB1_1101(S)	0.760	0.780	0.791	0.808	0.817	0.823	0.825	**0.827**	0.823	0.824	0.824	0.825	0.827	0.823	0.826
DRB1_1101(A)	0.762	0.783	0.792	0.802	0.815	0.819	0.822	0.824	0.822	0.822	0.823	**0.825**	0.824	0.823	0.825
DRB1_1201(S)	0.782	0.797	0.810	0.817	0.815	0.829	0.834	0.832	**0.837**	0.834	0.837	0.834	0.836	0.833	0.830
DRB1_1201(A)	0.792	0.804	0.814	0.819	0.825	0.831	0.837	0.833	**0.839**	0.838	0.835	0.832	0.831	0.834	0.836
DRB1_1302(S)	0.802	0.815	0.823	0.823	0.834	0.838	0.840	0.842	**0.848**	0.840	0.845	0.845	0.845	0.844	0.846
DRB1_1302(A)	0.807	0.817	0.821	0.826	0.836	0.839	0.843	0.843	0.846	0.846	0.844	**0.850**	0.847	0.845	0.849
DRB1_1501(S)	0.714	0.733	0.748	0.760	0.774	0.777	0.783	0.782	0.788	0.790	0.791	**0.794**	0.791	0.790	0.789
DRB1_1501(A)	0.713	0.727	0.747	0.765	0.767	0.778	0.786	**0.794**	0.788	0.790	0.794	0.794	0.791	0.793	0.794
DRB3_0101(S)	0.787	0.801	0.795	0.799	0.805	0.817	0.820	0.822	0.817	0.818	0.816	0.819	**0.823**	0.818	0.818
DRB3_0101(A)	0.787	0.802	0.804	0.804	0.821	0.823	0.822	0.824	0.827	0.823	0.824	0.817	0.825	0.822	**0.829**
DRB3_0301(S)	0.710	0.737	0.743	0.751	0.761	0.777	0.777	0.780	**0.795**	0.777	0.785	0.794	0.773	0.777	0.781
DRB3_0301(A)	0.721	0.726	0.745	0.761	0.781	0.782	0.784	0.782	0.773	**0.786**	0.775	0.770	0.776	0.778	0.782
DRB4_0101(S)	0.754	0.767	0.776	0.780	0.789	0.787	0.796	0.797	0.795	0.794	0.799	0.799	0.797	**0.801**	0.797
DRB4_0101(A)	0.755	0.764	0.784	0.784	0.794	0.795	0.797	0.793	0.797	0.800	0.800	0.800	0.799	**0.802**	0.801
DRB5_0101(S)	0.717	0.726	0.740	0.756	0.781	0.797	0.799	0.805	0.801	**0.811**	0.808	0.806	0.803	0.804	0.810
DRB5_0101(A)	0.716	0.721	0.736	0.762	0.778	0.789	0.804	0.806	0.809	0.810	0.809	0.804	0.807	0.809	**0.812**

Table showing the AUC values for the stacked and aggregated input formulations comparing the values for all 0-14 PFRs to AUC values when the stopping criterion is effected at 6 PFRs. The maximum AUC value for each allele is shown in bold while the maximum AUC value, when employing the stopping criterion, is underlined and in bold.

Each MHC-II allele has an optimal number of PFRs, where the model performance is maximized. Table 3.2 shows the AUC values for both the stacked and aggregated input models. The maximum AUC value for each allele is shown in bold while the maximum AUC value, when employing the stopping criterion, is underlined. The variability in optimal number of PFRs across alleles is higher without the condition of the stopping criterion. By employing the choice of 6 PFRs as the stopping criterion for PFR investigation, the variability in optimal number of PFRs is decreased while model performance remains fairly preserved. Accordingly, a balance is struck between optimizing computational complexity without losing significant amounts of information. Other benefits of employing this stopping criterion are that it will improve the time demand of the model training process, save on computational expense and serve as a consideration in the foundation for the development of the CNN model and its extension of a transallelic model discussed in Chapter 4.

3.5 The Effect of Zero-padding on Model Performance

In order to compensate for the loss in data size for the *stacked* and *aggregated input formulations*, we padded the input tensor with zero matrices along its depth for shorter peptides which resulted in fewer peptide registers to be encoded into the 2D matrices as in Figure 2.1. In this section we examine the accuracy of the aggregated input model and compare its performance with and without zero-padding. The objective is to assess the effect of zero padding beyond 6 PFRs which is the point where data size dramatically decreases.

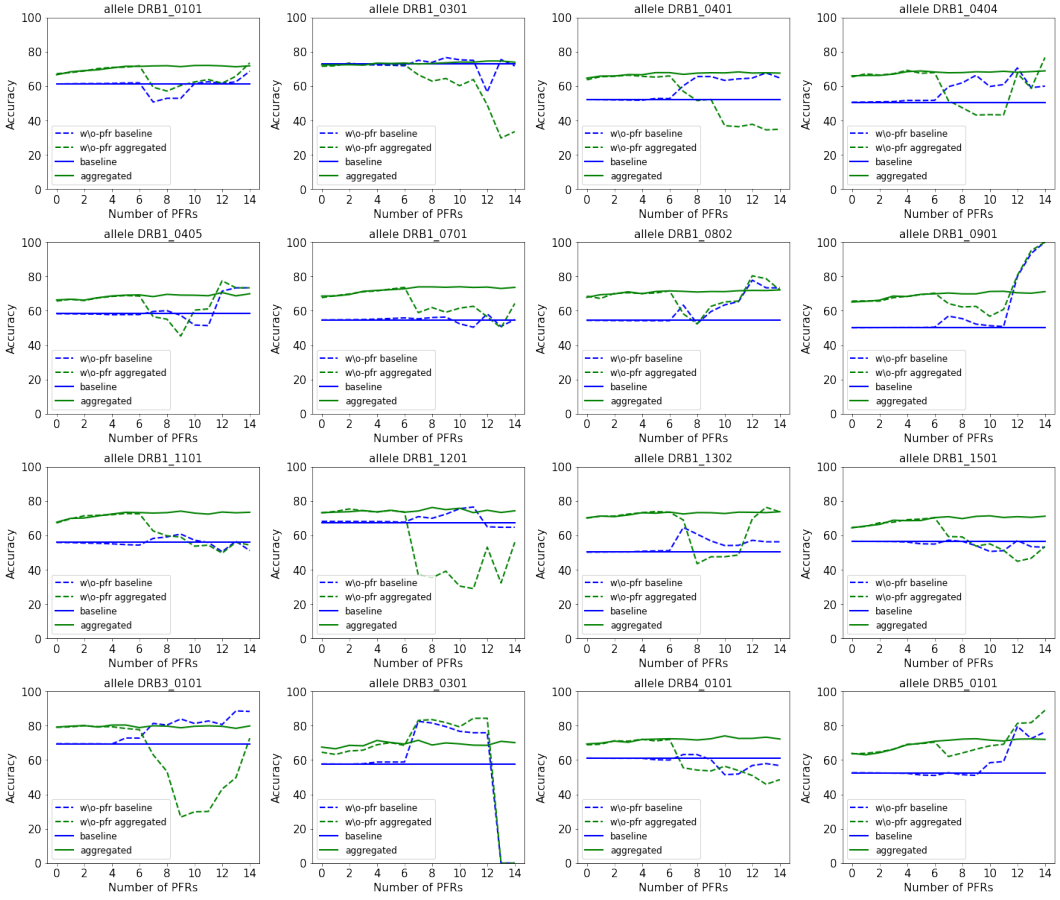


Figure 3.6: A comparison of the performance accuracy of the aggregated input model and baseline accuracy with zero-padding (solid curves) and without zero-padding (dotted curves) over 0-14 PFRs for all 16 MHC-II alleles considered. Zero-padding shows an inflation in model performance beyond 6 PFRs across alleles.

Figure 3.6 plots accuracy curves for the aggregated input model for the 16 MHC-II alleles considered on a dataset whereby shorter peptides were removed as the num-

ber of PFRs were increased, thereby exhausting the number of peptide registers that could have been encoded along the depth of the input tensor. The solid green curve shows accuracy of the aggregated input model with zero-padding, while the dotted green curve shows accuracy without zero-padding. The solid blue curve is the baseline accuracy curve, while the dotted blue curve is the baseline accuracy that takes into account the removal of zero matrices that are used to pad the input matrices. The baseline accuracy curves are included so as to assess and compare the models' learning capacity.

The performance of the aggregated input model without zero-padding (green dotted curve in Figure 3.6) deteriorates dramatically from 6 PFRs onwards. This is a similar trend to the performance of the without-PFR input model in Figure 3.3. Also, beyond 6 PFRs, the accuracy of the aggregated input model without zero-padding becomes erratic and even decreases below the baseline accuracy for most alleles, indicating that it stops learning for large numbers of PFRs. This further justifies the implementation of the stopping criterion of model performance investigation at 6 PFRs as recommended in Section 3.4.

The effect of heavily zero-padding model inputs exaggerates the performance of the model for larger numbers of PFRs (Figure 3.6), specifically beyond 6 PFRs. To mitigate these effects, we reduced and eliminated the extent to which zero padding is used in our model. One way we achieved this is by implementing the stopping criterion of training our model for up to 6 PFRs, as beyond this point, there is a huge discrepancy in model performance between zero-padded and non-zero-padded input models.

3.6 Termini Investigation

A peptide is represented by a short sequence of amino acid residues linked by hydrogen bonds. Peptides have two termini known as the N-terminus and the C-terminus, each with its own properties. The peptide sequence is conventionally written from left to right, which represents the N-terminus and the C-terminus, respectively.

When a peptide is placed into the binding groove of an MHC-II molecule, the docking of this binding process can start from either terminus of the peptide. Therefore, PFRs can be found on either side. Computationally, it may make a difference which terminus is used to process the input or to represent both termini simultaneously. Thus far only the N-terminus has been considered. In this section, we consider the C-terminus and the combined C and N termini in processing the model input and compare the model performance, review the results and discuss the advantages and disadvantages of using the different terminus configurations for input data processing.

We present the performance of our CNN model in Table 3.3 in terms of the summary statistics of the AUC values obtained from the model in the three terminus input configurations. Table 3.3 indicated further statistical tests that we used to investigate whether there is a significant difference in model performance when varying the terminus configurations. The comparison between the C-terminus and the N-terminus configurations is discussed in Section 3.6.1 and the comparison between the N-terminus configuration, and the combined C and N termini configuration is discussed in Section 3.6.2.

3.6.1 The C-terminus vs N-terminus

The aim of considering the input processed from the C-terminus is to assess if the choice of processing the input from either terminus influences the performance of the model when varying the number of PFRs. We compared model performance in terms of AUC values and accuracy between the *C-terminus* and the *N-terminus input configurations*, and employed the stopping criterion for PFR investigation as in Section 3.4 by assessing the model performance for 0 to 6 PFRs.

The model performance for the C-terminus aggregated input for 0-6 PFRs, measured in terms of AUC values is presented in Table 3.4 for the 16 MHC-II alleles covered in the study. Figure 3.7 shows the corresponding AUC curves.
Table 3.4 reports on the performance of the CNN model with input processed from the C-terminus in terms of AUC values. Each row shows the results of the model performance for each MHC-II allele considered in the study over the range of 0 to 6 PFRs. The first column lists the 16 MHC-II allele and every column after that lists AUC values of the model for each of the numbers of PFRs. The maximum AUC value that occurs at the number of PFRs for which the model performs at its optimum is highlighted in bold for each allele.

Figure 3.7 shows that there is an increasing trend in AUC values as the number of PFRs increase across all 16 MHC-II alleles. This is similar to what was observed in the N-terminus case (Figure 3.5). Furthermore, Table 3.4 shows that the C-terminus input configuration model performs best at 5 or 6 PFRs under the stopping criterion with the exception of *DRB1_0301* with optimal performance at 4 PFRs. Similar results are observed for the N-terminus aggregated input formulation under the stopping criterion (Table 3.2).

Figure 3.8 shows the absolute differences in AUC values of the C-terminus and the N-terminus input configurations over 0 to 6 PFRs for each of the 16 MHC-II alleles. The curves in (A) show the trend of the differences over the numbers of PFRs, while the boxplots in (A) summarize the distributions of the differences for each of the 16 alleles.

Table 3.3: Descriptive statistics of the model performance in terms of AUC values. The inputs are processed in three different terminus configurations for our CNN model, the N-terminus, C-terminus and combined C and N termini.

Allele (DRB)	1_0101	1_0301	1_0401	1_0404	1_0405	1_0701	1_0802	1_0901	1_1101	1_1201	1_1302	1_1501	3_0101	3_0301	4_0101	5_0101
N-terminus																
mean	0.750	0.716	0.751	0.745	0.760	0.805	0.796	0.760	0.799	0.817	0.827	0.755	0.809	0.757	0.782	0.758
median	0.756	0.721	0.755	0.748	0.763	0.808	0.797	0.768	0.802	0.819	0.826	0.765	0.804	0.761	0.784	0.762
std	0.024	0.020	0.014	0.013	0.013	0.021	0.014	0.025	0.022	0.016	0.013	0.027	0.013	0.027	0.016	0.034
C-terminus																
mean	0.746	0.717	0.737	0.727	0.731	0.802	0.783	0.758	0.788	0.808	0.828	0.754	0.812	0.724	0.761	0.766
median	0.761	0.726	0.743	0.734	0.731	0.806	0.787	0.760	0.790	0.813	0.829	0.761	0.816	0.723	0.770	0.768
stdev	0.030	0.025	0.024	0.026	0.035	0.022	0.021	0.024	0.027	0.021	0.018	0.030	0.014	0.037	0.032	0.030
Combined																
mean	0.771	0.733	0.754	0.747	0.757	0.824	0.802	0.782	0.816	0.826	0.839	0.779	0.817	0.764	0.788	0.792
median	0.776	0.740	0.761	0.746	0.768	0.825	0.804	0.786	0.820	0.825	0.844	0.785	0.818	0.769	0.789	0.796
stdev	0.010	0.013	0.016	0.007	0.021	0.009	0.011	0.010	0.013	0.005	0.010	0.012	0.011	0.015	0.009	0.013

Table shows the summary statistics of the model performance in terms of AUC values for the input varied in processing from the N-terminus, C-terminus and combined C and N termini for each of the 16 MHC-II alleles.

The curves in Figure 3.8(A) show that the larger differences occur at the smaller numbers of PFRs and decrease as the number of PFRs are increased across all alleles. The largest differences across alleles are less than 0.07 at 0 PFRs and less than 0.025 at 6 PFRs. These are small differences; however, they are further analysed

Table 3.4: AUC values of the CNN model of the aggregated input formulation from the C-terminus for 16 MHC-II alleles covered in the study over 0 to 6 PFRs.

Allele \PFRs	0	1	2	3	4	5	6
DRB1_0101	0.701	0.715	0.729	0.761	0.767	0.774	**0.775**
DRB1_0301	0.674	0.698	0.710	0.726	**0.740**	0.734	0.739
DRB1_0401	0.697	0.718	0.725	0.743	0.751	0.760	**0.762**
DRB1_0404	0.683	0.707	0.717	0.734	0.743	0.749	**0.756**
DRB1_0405	0.672	0.705	0.720	0.731	0.752	0.765	**0.769**
DRB1_0701	0.764	0.786	0.798	0.806	0.814	0.819	**0.829**
DRB1_0802	0.749	0.770	0.774	0.787	0.795	0.802	**0.808**
DRB1_0901	0.718	0.740	0.749	0.760	0.773	0.781	**0.785**
DRB1_1101	0.740	0.768	0.780	0.790	0.803	0.811	**0.820**
DRB1_1201	0.768	0.797	0.808	0.813	0.816	0.824	**0.831**
DRB1_1302	0.796	0.815	0.824	0.829	0.840	0.844	**0.847**
DRB1_1501	0.706	0.725	0.745	0.761	0.777	0.779	**0.783**
DRB3_0101	0.788	0.800	0.816	0.811	0.821	**0.826**	0.823
DRB3_0301	0.673	0.701	0.693	0.723	0.747	**0.770**	0.761
DRB4_0101	0.709	0.735	0.747	0.770	0.782	**0.792**	0.791
DRB5_0101	0.721	0.740	0.753	0.768	0.785	0.792	**0.804**

Table shows the maximum AUC value, which is the point where the model performs at its optimum, highlighted in bold for each of the 16 MHC-II alleles covered in the study. The maximum AUC value occurs at 6 PFRs for 12 of the 16 alleles. This is a similar result to the N-terminus case (See Table 3.2).

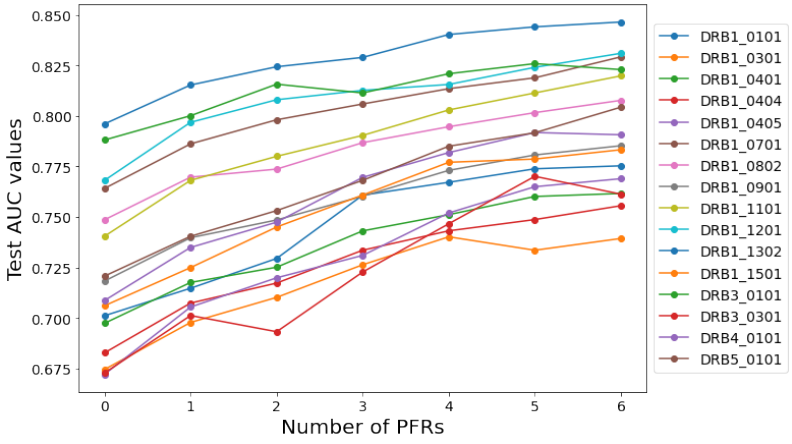
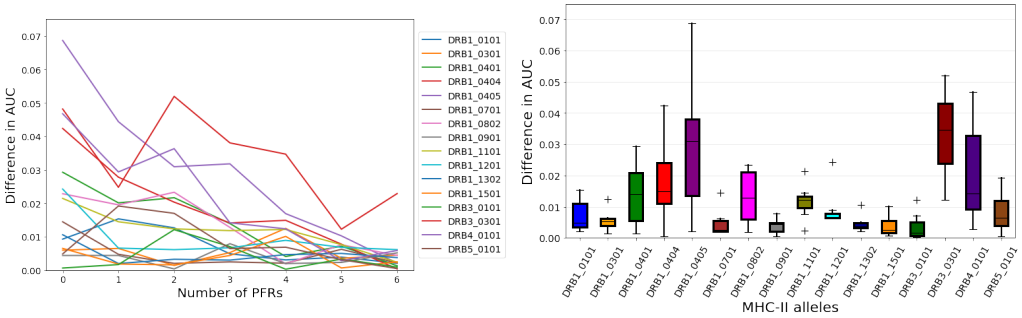


Figure 3.7: AUC curves representing the performance of the model for input processed from the C-terminus over varying numbers of PFRs for the 16 MHC-II alleles covered in the study. Model performance improves with the increase in number of PFRs as in the N-terminus configuration.

statistically to determine whether the differences are significant to change the input formulation for improvement of model performance as they relate to varying numbers of PFRs. The distribution of the absolute differences in AUC values for each allele are summarized in the boxplots in Figure 3.8(B). The boxplots show that the



(A) (B)

Figure 3.8: Curves (A) shows that with the increase in number of PFRs that the difference in model performance between the N and the C-terminus configurations decreases over the range of 0 to 6 PFRs, and boxplots (B) of the absolute difference in AUC values of the C and N-terminus input configurations are small across alleles for the 16 MHC-II alleles considered.

distributions of the differences are skewed for most alleles which is a good indication that normality cannot be assumed, which informs an adequate statistical test.

Figure 3.9 shows the correlation between the N-terminus and the C-terminus AUC values for 0 to 6 PFRs across the 16 MHC-II alleles. Table 3.5 presents the Pearson's correlation coefficient (PCC) and p-value results of the statistical test of the differences in AUC values for the 16 MHC-II alleles over 0 to 6 PFRs.

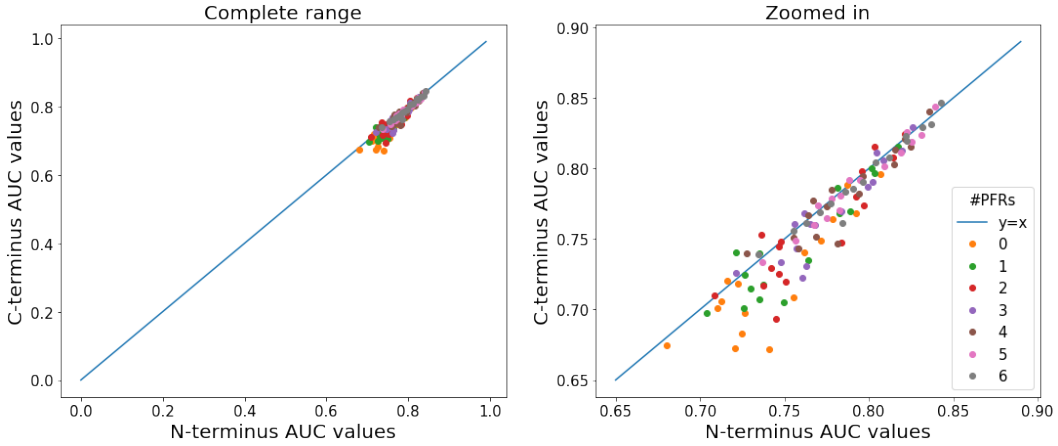


Figure 3.9: Correlation scatterplots of AUC values for the N-terminus vs C-terminus input configurations: The left shows the whole range of the AUC values and the right is zoomed in to the datapoints. A strong positive correlation is evident with a slight bias towards the N-terminus, indicating that if the difference in model performance is statistically significant, then the N-terminus input model is superior.

The plots in Figure 3.9 show a strong positive correlation between the AUC values of the model with inputs processed from the C-terminus and the N-terminus for each allele. The PCC values in Table 3.5 support this with values above 0.8 over the range of 0 to 6 PFRs. These results indicate that the model behaves similarly with inputs processed from either terminus. While the model is allele specific, the results thus far along with the trend observed in Figure 3.8(A) of the absolute differences in AUC values over the number of PFRs show that it is important to investigate the model performance relating to the number of PFRs. So, for the comparison of the model performance for N-terminus and C-terminus input configurations we performed a statistical significance test for varying numbers of PFRs.

Table 3.5: Pearson's correlation coefficients (PCC) results and p-values for the one-sided Wilcoxon Signed Rank test over the AUC values of the model with input processed from the C-terminus vs. the N-terminus.

PFRs	0	1	2	3	4	5	6
PCC	0.865	0.914	0.890	0.923	0.924	0.985	0.981
p-value	1.000	0.994	0.978	0.987	0.902	0.987	0.981

We carried out the one-sided Wilcoxon Signed Rank test with confidence level of 95% on the differences in AUC values of the C-terminus and N-terminus configurations as the data samples. Furthermore, we assessed the statistical significance of the difference in model performance for each number of PFRs. The results in terms of p-values are reported in Table 3.5. The order of the alleles had to be retained in each data sample. We are using AUC value data from the aggregated input formulation for the N-terminus as in Section 3.2; so, we cannot assume normality for the distribution of the data samples. This assumption is further supported by the boxplots in Figure 3.8(B). The sample size is 16 which is the number of MHC-II alleles considered, this sample size is considered small. Therefore, for all the aforementioned characteristics of the analysis, we required a paired non-parametric statistical test; hence, the Wilcoxon Signed Rank test.

The goal was to analyse whether it is warranted to replace the input processed from the N-terminus with the input processed from the C-terminus for significant improvement to the CNN model. We achieved this by assessing whether the difference in model performance is statistically and sufficiently significant. We subtract the AUC values of the N-terminus configuration from the AUC values of the C-terminus configuration and assess the difference in medians for each number of PFRs.

The hypotheses of the test are as follows:

$$H_0 \text{ :The median difference is 0,}$$
$$H_a \text{ :The median difference is } > 0,$$

where the null hypothesis (H_0) represents that the model performance is either better with input processed from the N-terminus or similar when the input is processed from either terminus. The alternative hypothesis (H_a) represents that the model performance is improved with input processed from the C-terminus.

The p-value results in Table 3.5 are greater than the significance level of $\alpha = 0.05$ for all numbers of PFRs; therefore, we fail to reject the null hypothesis for the whole range of 0 to 6 PFRs. This leads to the conclusion that the difference in model performance is not statistically significant to warrant replacing the N-terminus input configuration with the C-terminus input configuration as the model yields either better performance for the N-terminus input configuration or similar performance for either terminus over the whole range of 0 to 6 PFRs.

3.6.2 The Combined C and N termini vs. N-terminus

Thus far we have assessed the model with an input from only a single terminus, either the N or the C-terminus. In this section we aim to analyse the model with input that represents an equal likelihood that the peptide binds from either terminus by combining information of both termini simultaneously. The performance of the model was assessed using AUC values and accuracy and was compared to the single terminus cases. The stopping criterion as discussed in Section 3.4 was employed; so, the range of PFRs we varied is 0 to 6.

The model performance in terms of AUC values for the combined C and N termini input configuration from 0 to 6 PFRs for the 16 MHC-II alleles covered in the study is presented in Table 3.6.

Table 3.6: AUC values of the CNN model of the aggregated input formulation from the combined C and N termini for 16 MHC-II alleles covered in the study with varying numbers of PFRs.

Allele \PFRs	0	1	2	3	4	5	6
DRB1_0101	0.752	0.764	0.772	0.778	0.776	0.776	**0.781**
DRB1_0301	0.712	0.721	0.728	0.741	0.742	**0.746**	0.740
DRB1_0401	0.722	0.743	0.754	0.761	0.766	**0.767**	0.763
DRB1_0404	0.734	0.746	0.745	0.753	0.753	0.746	**0.755**
DRB1_0405	0.715	0.746	0.754	0.771	0.768	**0.776**	0.771
DRB1_0701	0.813	0.812	0.821	0.825	**0.833**	0.830	0.831
DRB1_0802	0.780	0.797	0.802	0.808	0.804	**0.816**	0.806
DRB1_0901	0.767	0.776	0.775	0.786	0.787	0.787	**0.797**
DRB1_1101	0.787	0.810	0.820	0.820	**0.826**	0.825	0.821
DRB1_1201	0.818	0.822	0.828	0.825	0.828	0.824	**0.834**
DRB1_1302	0.826	0.824	0.840	0.845	0.846	0.844	**0.850**
DRB1_1501	0.763	0.766	0.772	0.787	0.787	0.785	**0.796**
DRB3_0101	0.803	0.806	0.811	0.818	**0.829**	0.827	0.824
DRB3_0301	0.738	0.751	0.770	**0.780**	0.769	0.763	0.776
DRB4_0101	0.773	0.782	0.786	0.790	**0.800**	0.789	0.795
DRB5_0101	0.768	0.783	0.790	0.796	0.801	0.800	**0.804**

Table shows the maximum AUC value, which is the point where the model performs at its optimum, highlighted in bold for each of the 16 MHC-II alleles covered in the study. The maximum AUC value occurs at 6 PFRs for 7 of the 16 alleles.

The first column in Table 3.6 lists the 16 MHC-II alleles and each column after that represents the AUC values for 0 to 6 PFRs. The maximum AUC value indicating the number of PFRs at which the model performs optimally for each MHC-II allele is highlighted in bold in each row. The maximum AUC values occur at 6 PFRs for only 7 out of the 16 MHC-II alleles (Table 3.6). With the maximum occurring earlier in the combined termini case for more alleles than in the single terminus cases (Tables 3.2 and 3.4), and as early as 3 PFRs for allele DRB3_0301, it shows that less work needs to be done for optimal model performance for input processed from both termini combined.

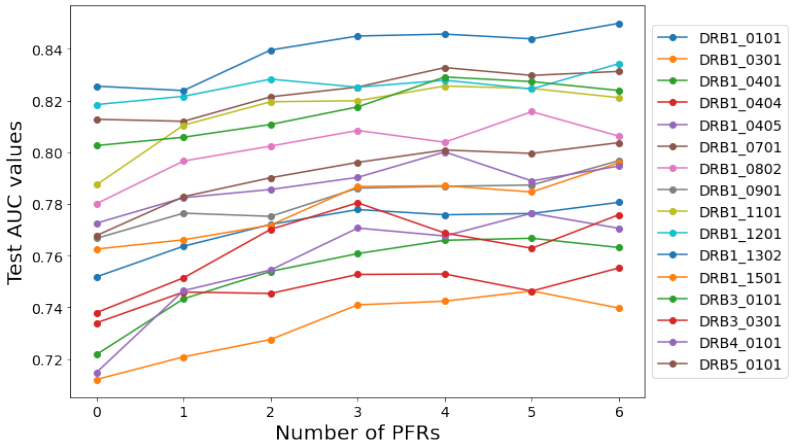


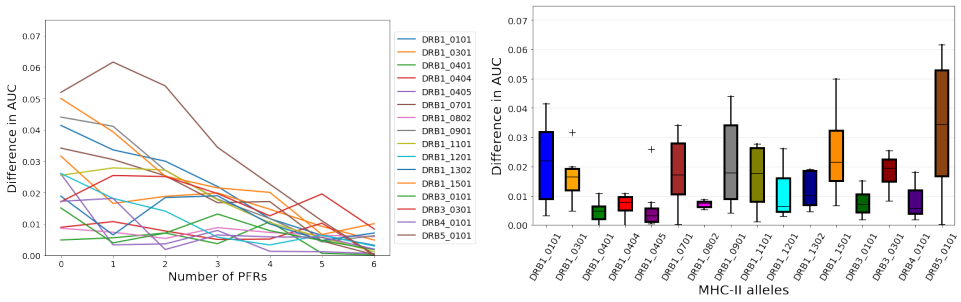
Figure 3.10: Curves of AUC values representing the performance of the model for input processed from the combined C and N termini over varying numbers of PFRs in the range of 0 to 6 for the 16 MHC-II alleles covered in the study. The performance of the model remains consistent in improving with the increase in the number of PFRs; however, appear to be flatter than the AUC curves for the N-terminus and C-terminus configurations.

The curves in Figure 3.10 show an increasing trend in AUC values but are flatter than those in the single terminus input configuration (Figures 3.5 and 3.7) across all 16 MHC-II alleles. This corresponds to the results observed in Table 3.6 where the maximum AUC values occur at smaller numbers of PFRs than in the case of the single terminus configurations.

Figure 3.11 shows the absolute difference in AUC values of the combined C and N termini and the N-terminus input configurations over the range of 0 to 6 PFRs for each of the 16 MHC-II alleles. The curves in (A) show the trend of the differences over the range of PFRs while the boxplots in (B) summarize the distributions of the differences for each of the alleles.

Similarly to the single terminus case in Section 3.6.1, the difference in AUC curves in Figure 3.11(A) show a decreasing trend along the varying numbers of PFRs. The largest difference across alleles is less than 0.065 at 0 PFRs and less than 0.01 at 6 PFRs. These are small differences; hence, we further tested the significance of the differences statistically. The distributions of the differences in AUC values for each allele are summarized in the boxplots in Figure 3.11(B).

Figure 3.12 shows the correlation between the model with input processed from the combined C and N termini and the input processed from the N-terminus for varying numbers of PFRs ranging from 0 - 6 PFRs for each of the 16 MHC-II alleles. Table 3.7 reports the PCC and p-value results of the statistical test of the differences in



(A) (B)

Figure 3.11: Curves in (A) show smaller differences the higher the number of PFRs as in the C-terminus configuration over the range of 0 to 6 PFRs, and boxplots in (B) of the absolute differences in AUC values of the combined C and N termini and the N-terminus input configurations show even smaller differences in model performances as in the N- vs. C-terminus comparison for the 16 MHC-II alleles considered.

AUC values for the 16 MHC-II alleles over 0 to 6 PFRs.

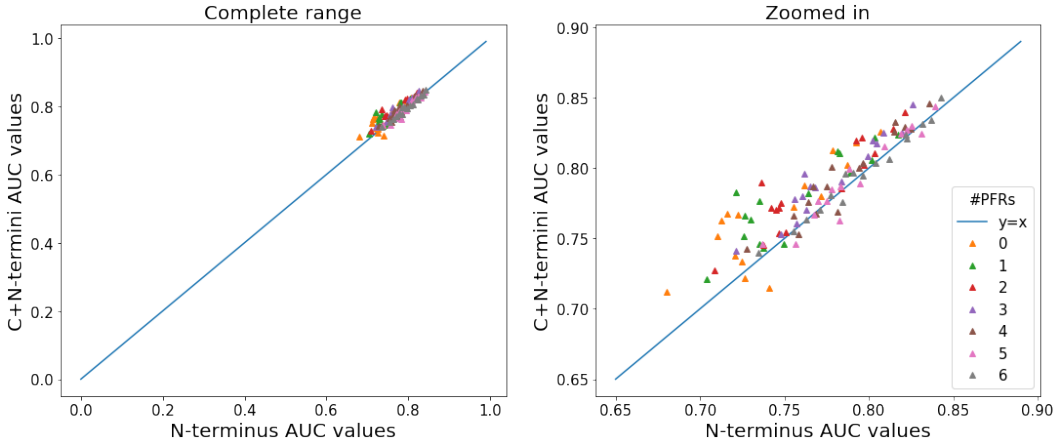


Figure 3.12: Correlation scatterplots of AUC values for the N-terminus vs. the combined C and N termini input configurations: The left shows the whole range of AUC values and the right is zoomed in to the datapoints. These plots show a strong positive correlation with a slight bias towards the combined C and N termini configuration indicating it to have superior model performance in the event the difference is significant.

Figure 3.12 shows a strong positive correlation between the AUC values of the model with inputs processed from the combined C and N termini and the N-terminus. This is supported by the PCC values in Table 3.7 with all values greater than 0.8 for all numbers of PFRs. This result indicates that the model behaves similarly with either input configuration over the range of number of PFRs; however, does not in-

form the significance of the difference in model performance.

We compared the difference in model performance between the combined C and N termini and the N-terminus input configurations as it relates to varying the number of PFRs statistically. The results of the statistical test in terms of p-values are outlined in Table 3.7.

Table 3.7: Pearson's correlation coefficients and p-values for AUC values of the model with input processed from the combined C and N termini vs. N-terminus.

PFRs	0	1	2	3	4	5	6
PCC	0.840	0.873	0.917	0.960	0.957	0.965	0.988
p-value	0.0014	0.0003	0.0002	0.0002	0.0036	0.1760	0.3782

We carried out the Wilcoxon Signed Rank test with confidence level of 95% on the differences in AUC values of the model with inputs processed from the combined C and N termini and from the N-terminus as the data samples. The aim of the test was to determine whether the difference in model performance is sufficiently significant to favour the use of the combined C and N termini input configuration over the N-terminus input configuration. The Wilcoxon Signed Rank Test is used in this case as the conditions are the same as in Section 3.6.1.

We subtracted the AUC values of N-terminus configuration from the combined C and N termini configuration, and assess the difference in medians for each number of PFRs. In this case, the null hypothesis represents that the model performance is either better with input processed from the N-terminus or the performance is similar in either configuration. The alternative hypothesis represents that the model performance is improved with input processed from the combined C and N termini.

The p-values in Table 3.7 are less than the significance level of $\alpha = 0.05$; therefore, we reject the null hypothesis for the alternative hypothesis for 0 to 4 PFRs. Otherwise, we fail to reject the null hypothesis for 5 and 6 PFRs. This leads to the conclusion that model performance for the combined C and N termini input configuration is better for smaller numbers of PFRs while the difference in model performance is not significantly different for larger numbers of PFRs. This is a mixed result but indicates an advantage of using the combined C and N termini input configuration as it produces improved results earlier in the varying of the number of PFRs.

3.6.3 Conclusion of the Termini Investigation

The similarity in model performance means that we can mostly use either model without losing information, even though it is indicated that the combined C and N

termini input configuration is better. To definitively determine which model to use, we further compared the maximum AUC values over 0 - 6 PFRs from the three ways the input was processed, namely, the N-terminus, the C-terminus and the combined C and N termini. These maximum AUC values are reported in Table 3.8.

Table 3.8: Maximum AUC values for model inputs from the N-terminus, C-terminus and combined C and N termini and the number of PFRs at which they occur.

Allele	N-terminus	C-terminus	Combined
DRB1_0101	0.777 (6)	0.775 (6)	**0.781** (6)
DRB1_0301	0.737 (5)	0.740 (4)	**0.746** (5)
DRB1_0401	0.767 (5)	0.762 (6)	**0.767** (5)
DRB1_0404	**0.756** (5)	0.756 (6)	0.755 (6)
DRB1_0405	0.775 (5)	0.769 (6)	**0.776** (5)
DRB1_0701	0.831 (6)	0.829 (6)	**0.833** (4)
DRB1_0802	0.812 (6)	0.808 (6)	**0.816** (5)
DRB1_0901	0.790 (6)	0.785 (6)	**0.797** (6)
DRB1_1101	0.822 (6)	0.820 (6)	**0.826** (4)
DRB1_1201	**0.837** (6)	0.831 (6)	0.834 (6)
DRB1_1302	0.843 (6)	0.847 (6)	**0.850** (6)
DRB1_1501	0.786 (6)	0.783 (6)	**0.796** (6)
DRB3_0101	0.823 (5)	0.826 (5)	**0.829** (4)
DRB3_0301	**0.784** (6)	0.770 (5)	0.780 (3)
DRB4_0101	0.797 (6)	0.792 (5)	**0.800** (4)
DRB5_0101	0.804 (6)	**0.808** (6)	0.804 (6)

Table showing the best AUC results for each of the inputs processed from the N-terminus, C-terminus, and combined C and N termini for each of the 16 MHC-II alleles and the number of PFRs at which they occur shown in parentheses. The highest AUC value of the 3 input configurations for each MHC-II allele is highlighted in bold and occurs for 12 out of 16 alleles in the combined C and N termini input configuration.

The first column of Table 3.8 lists the 16 MHC-II alleles considered in the study. The second, third and forth columns are the maximum AUC values for the N-terminus, the C-terminus and the combined C and N termini input configurations, respectively. The corresponding number of PFRs at which the maximum AUC occurs is indicated in parentheses. For each allele, the highest value is highlighted in bold indicating at which input configuration the model best performs and the number of PFRs at which this occurs.

The model with inputs processed from the combined C and N termini outperforms the other two input configurations for 12 out of the 16 MHC-II alleles. Note that the maximum AUC values occur at smaller numbers of PFRs for the combined termini input configuration than the single terminus input configurations for most alleles.

Figure 3.13 shows accuracy curves for the model with inputs from the N-terminus, C-terminus and the combined C and N termini for 16 MHC-II alleles and varying

numbers of PFRs ranging from 0 to 6.

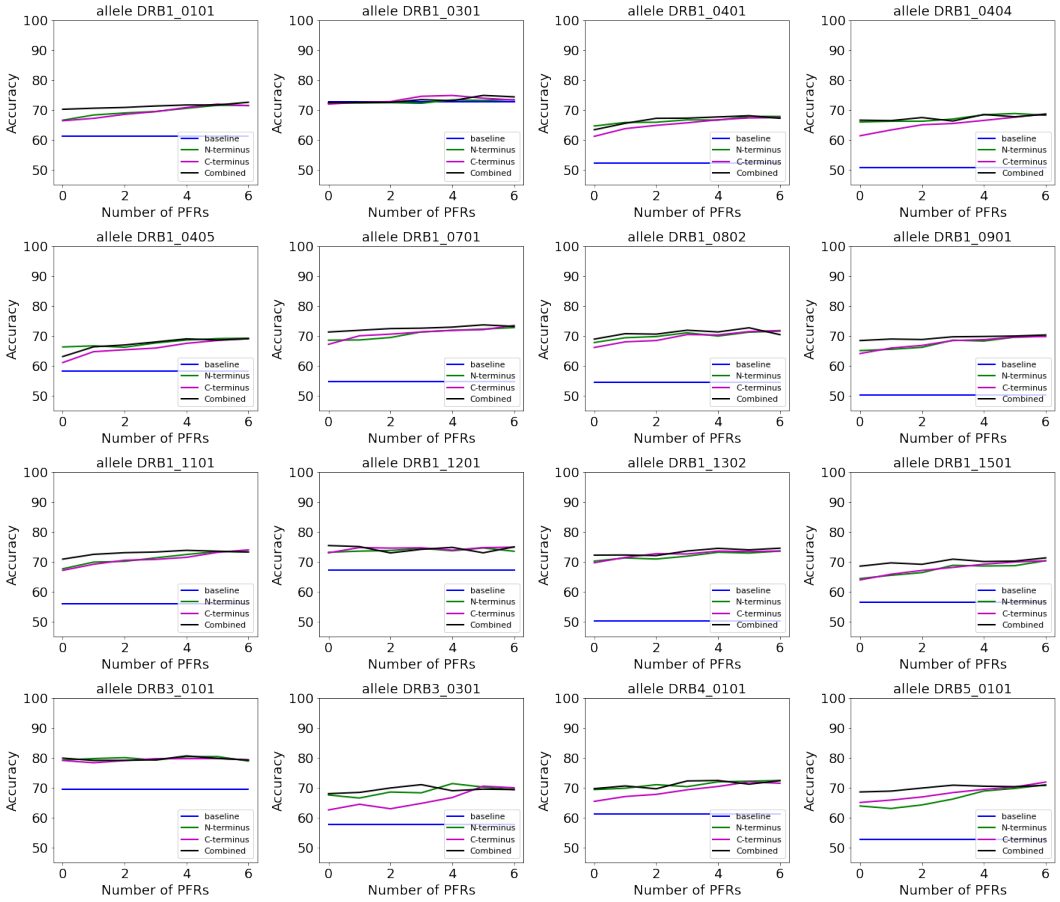


Figure 3.13: A comparison of the performance of the CNN model in terms of accuracy for aggregated input processed from the N-terminus (in green), C-terminus (in magenta) and combined C and N termini (in black) over 0-6 PFRs for the 16 MHC-II alleles considered in the study. All three terminus configuration input models learn well over the entire range of PFRs and have a steady increasing trend in model performance. However, across most alleles the combined C and N termini configuration yields better performance than the other two configurations and moreover, for even smaller numbers of PFRs.

The accuracy curves in Figure 3.13 show that the model is learning in all three terminus configurations over the range of 0 to 6 PFRs for all the 16 MHC-II alleles considered in the study. The accuracy for the combined C and N termini input configuration is higher than or similar to the accuracy for the single terminus input configurations over the range of 0 to 6 PFRs for all alleles. These accuracy curves further indicate that the model can be trained for smaller numbers of PFRs to achieve optimal performance for the combined C and N termini input configura-

tion, which decreases computational expense.

Concluding the experiments of the input formulations and the terminus input configurations, the results of our CNN model from investigations in terms of AUC values and accuracy strongly indicate that the best model input to use is the aggregated formulation processed in the combined C and N termini configuration also employing the stopping criterion of investigating PFRs. This way optimal performance of the CNN model is ensured across all alleles over the range of 0 to 6 PFRs with input that includes the most information possible about the peptide-MHC-II binding interactions while requiring less work. The result is a comprehensively representative and computationally optimized CNN model for predicting peptide-MHC-II binding interactions.

3.7 Performance Comparison with NetMHCII-2.3

Allele-specific models are restrictive in nature as they treat data for each MHC-II allele as independent. This may be the reason behind the lack of such models in literature as they also form the foundation of their more generalizable transallelic counterpart models, e.g. the allele-specific NetMHCII and the transallelic NetMHCIIpan. Our CNN model is primarily an allele-specific model which factors in the influence of PFRs in its predictive performance. One model that is known to have these characteristics is the NetMHCII model of Jensen *et al.* (2018). It was first published in 2009 and has since been updated to include the NN-Align, an ANN based alignment algorithm Nielsen and Lund (2009) applied to it for improved performance. NetMHCII along with its transallelic counterpart are considered the state-of-the-art models in the area of peptide-MHC-II binding interaction predictions (Liu *et al.*, 2021; Degoot *et al.*, 2018).

We compared the performance of our CNN model to the performance of the latest version of NetMHCII, version 2.3 published in 2018. Two datasets were used in reporting the results of NetMHCII-2.3, the peptide-MHC-II interaction dataset we used to train our CNN model from 2016 obtained from IEDB and the 2013 dataset from which the 2016 dataset was extended. Our CNN model was only trained and tested on HLA-DR data of the 2016 dataset, while NetMHCII-2.3 was trained on all three loci of the human MHC-II alleles, HLA-DP, HLA-DQ and HLA-DR and on the H2 locus of the mouse MHC-II alleles. Of the HLA-DR locus data, the results of NetMHCII-2.3 are reported on 14 MHC-II alleles in Jensen *et al.* (2018). We report the results of our CNN model as trained on the 2016 dataset and tested on the 2013 dataset for the 16 MHC-II alleles covered in our study over 0 to 6 PFRs in Table 3.9 and compare its best results with the 14 MHC-II alleles as reported for NetMHCII-2.3 in Table 3.10.

Table 3.9: Predictive performance of our CNN model trained on the 2016 dataset and tested on the 2013 dataset for 16 MHC-II allele considered in the study and for 0 to 6 PFRs.

Allele PFRs	0	1	2	3	4	5	6
DRB1_0101	0.919	0.920	0.919	0.921	0.920	**0.922**	0.920
DRB1_0301	0.867	0.855	0.880	0.879	0.881	0.884	**0.891**
DRB1_0401	0.862	0.869	0.875	0.882	**0.890**	0.884	0.880
DRB1_0404	**0.961**	0.960	0.955	0.948	0.946	0.938	0.951
DRB1_0405	0.963	0.973	0.972	**0.975**	0.969	0.970	0.973
DRB1_0701	0.973	0.973	**0.977**	0.967	0.970	0.966	0.966
DRB1_0802	0.961	0.961	**0.962**	0.961	0.957	0.944	0.958
DRB1_0901	**0.969**	0.967	0.966	0.962	0.946	0.952	0.961
DRB1_1101	0.965	0.967	0.971	0.971	0.968	0.966	**0.973**
DRB1_1201	0.690	0.734	0.727	0.700	**0.740**	0.722	0.723
DRB1_1302	0.962	0.959	**0.967**	0.960	0.964	0.960	0.963
DRB1_1501	0.971	0.970	**0.972**	0.964	0.961	0.955	0.960
DRB3_0101	0.959	0.965	0.970	**0.971**	0.971	0.966	0.971
DRB3_0301	0.726	0.732	0.721	0.715	0.700	0.724	**0.749**
DRB4_0101	**0.977**	0.970	0.977	0.977	0.974	0.974	0.977
DRB5_0101	0.868	0.870	0.875	0.879	0.896	0.895	**0.901**

Table showing the AUC values for our CNN model as trained on the 2016 dataset and tested on the 2013 dataset. It shows the AUC values for the 16 MHC-II alleles considered in our study across 0 to 6 PFRs. The bold values are the highest AUC values for each MHC-II allele indicating the number of PFRs for which our CNN model performs at its optimum. The underlined values are the second highest AUC values for the MHC-II alleles whose optimum occurs at 4 or more PFRs.

The first column in Table 3.9 lists the 16 MHC-II alleles for which the resultant model, i.e., the CNN model with aggregated input formulation processed in the combined C and N termini configuration, was trained and tested. The rest of the columns show the AUC values achieved by the CNN model for each number of PFRs from 0 to 6, as trained on the 2016 dataset and tested on the 2013 dataset. The maximum AUC values for each MHC-II allele are highlighted in bold under the optimal number of PFRs; this indicates the number of PFRs at which the CNN model performs best for each MHC-II allele. The AUC values in bold for each MHC-II allele are used in the model performance comparison with NetMHCII-2.3 and presented in Table 3.10. Lastly, the second highest AUC values for each MHC-II allele, where the maximum AUC values occur at 4 PFRs or more, are underlined and will be considered in future developments of the CNN model and further analysed for the stopping criterion of PFR investigation.

The first column of Table 3.10 shows the 14 HLA-DR MHC-II alleles for which the two methods were tested. The second and third columns show the profile of the 2013 dataset; particularly, they respectively outline the number of peptides that are present in the dataset and how many of them bind for each of the 14 alleles. The third column shows the AUC value results of our CNN model and the number of

Table 3.10: Comparison of predictive performance, in terms of AUC values, of our CNN model and NetMHCII trained using five fold cross-validation on the 2016 dataset and tested on the 2013 dataset.

Allele	#Peptides	#Binders	CNN model	NetMHCII-3.2
DRB1_0101	2754	2635	**0.922** (5)	0.822
DRB1_0301	1403	379	**0.891** (6)	0.826
DRB1_0401	1639	695	**0.890** (4)	0.791
DRB1_0404	542	331	**0.961** (0)	0.768
DRB1_0405	1438	595	**0.975** (3)	0.860
DRB1_0701	1619	806	**0.977** (2)	0.857
DRB1_0802	1310	400	**0.962** (2)	0.767
DRB1_0901	841	560	**0.969** (0)	0.761
DRB1_1101	1604	730	**0.973** (6)	0.876
DRB1_1302	1351	463	**0.967** (2)	0.823
DRB1_1501	1601	672	**0.972** (2)	0.820
DRB3_0101	1266	267	**0.971** (3)	0.846
DRB4_0101	1329	467	**0.977** (0)	0.841
DRB5_0101	1606	765	**0.901** (6)	0.847

Table showing the profile of the 2013 dataset and the performance of our CNN model and of NetMHCII-2.3. The table outlines how many peptide-MHC-II interaction datapoints there are, along with how many of the peptides bind for 14 of the HLA-DR MHC-II alleles. It also shows AUC values of our CNN model and the number of PFRs at which they occur in parentheses and AUC values of NetMHCII-2.3. The bold values are the best performance scores for each MHC-II allele.

PFRs at which our CNN model performs best for each allele, and the forth column shows the AUC value results of NetMHCII-2.3. The values in bold are the higher of the AUC values when comparing the two methods.

In summary, the comparison of our CNN model and the state-of-the-art allele-specific model that predicts peptide-MHC-II binding interactions, NetMHCII-2.3, shows that our CNN model consistently outperforms the NetMHCII-2.3 model for every MHC-II allele of the HLA-DR locus on which it was tested (Table 3.10). Furthermore, the number of PFRs where our CNN model performs optimally, for 9 out of the 16 MHC-II alleles as shown in Table 3.9, is at most 3 PFRs. Moreover, for the remaining 7 of the 16 MHC-II alleles for which the maximum AUC values occur at 4 or more PFRs, the second highest AUC value occurs at 3 or fewer PFRs for 4 MHC-II alleles. This presents an opportunity to further optimize our model by training for half the number of PFRs as is our stopping criterion. This would significantly save on computational costs without losing much information while speeding up our model's predictive power.

Chapter 4

Transallelic Model

4.1 Introduction

The CNN-based model that has been detailed thus far is primarily an allele-specific model, as it is trained and evaluated on data from each MHC-II allele independently. In Chapter 3, the model performed well in terms of prediction. Furthermore, when compared to the most recent version of the cutting-edge allele-specific model that predicts peptide-MHC-II interactions, the NetMHCII-2.3 (Jensen *et al.*, 2018), our CNN model outperformed the NetMHCII-2.3 for all the 16 MHC-II alleles studied (see Table 3.10). The construction of a transallelic model based on the allele-specific CNN model is described in this chapter. The transallelic model's goal is to predict peptide-MHC-II binding affinity for previously unseen MHC-II alleles.

The transallelic version of the CNN model uses data from all known and accessible MHC-II alleles as a single dataset to train the model. The data used in training and testing the transallelic model is described in Section 4.2. The resultant model takes information about a peptide and an unknown MHC-II molecule as input and predicts binding interactions based on weights learned from peptide-MHC-II binding data of known MHC-II alleles. Because MHC-II alleles are highly polymorphic, developing a separate model for every conceivable MHC-II allele is not a scalable solution. There are currently very few MHC-II alleles with enough peptide-MHC-II binding data available to build a model with sufficient predictive ability. The size of the training dataset, as is widely known, is one of the elements that influences the predictive capacity of machine learning algorithms. Section 4.3 reports the performance achieved by the transallelic CNN model and compares this performance with that of the allele-specific CNN-based model.

4.2 Data

We interrogate the data profile for the transallelic CNN-based model. According to Table 4.1, the minimum and maximum peptide lengths are 9 and 37 amino acid residues long, respectively, for all of the 16 MHC-II alleles considered. We discovered that more than 80% of the peptides are 15 amino acid residues long; the distribution of peptide lengths is shown in the histogram in Figure 4.1. This corresponds to 6 PFRs, which is consistent with the results obtained in Section 3.4 to effect the stopping criterion for PFR inquiry at 6 PFRs. Thus, we do this for the transallelic model.

Table 4.1: Descriptive statistics of the peptide length data from all 16 MHC-II alleles that make up the dataset for training the transallelic model.

Allele	#Peptides	mean	stdev	min	median	max
DRB1_0101	10412	15.28	1.50	9	15	37
DRB1_0301	5352	15.54	1.99	9	15	35
DRB1_0401	6317	15.38	2.00	9	15	37
DRB1_0404	3657	15.32	1.55	9	15	31
DRB1_0405	3962	15.35	1.49	9	15	37
DRB1_0701	6325	15.41	1.74	9	15	37
DRB1_0802	4465	15.30	1.21	9	15	31
DRB1_0901	4318	15.30	1.32	9	15	31
DRB1_1101	6045	15.39	1.87	9	15	37
DRB1_1201	2384	15.35	1.51	9	15	33
DRB1_1302	4477	15.29	1.33	9	15	37
DRB1_1501	4850	15.52	1.98	9	15	37
DRB3_0101	4633	15.26	1.71	9	15	35
DRB3_0301	884	15.29	1.33	9	15	21
DRB4_0101	3961	15.39	1.77	9	15	35
DRB5_0101	5125	15.41	1.77	9	15	37
All alleles	77167	15.36	1.68	9	15	37

Table showing the profile of the entire dataset of 16 MHC-II alleles covered in the study. The first column lists the alleles, the second column the number of peptides belonging to each allele and the third to last column, the summary statistics for the peptide lengths.

Furthermore, it is important to analyse the extent to which MHC-II molecules share peptides. This is known as the allele-specificity and promiscuity of peptides -that is, a single peptide can bind several MHC-II molecules. For the model to predict, with any substantial success, the binding interactions between a peptide and an unseen MHC-II molecule, it must learn patterns from the known MHC-II alleles to which the peptide binds. The pie chart in Figure 4.2 depicts the extent to which peptides

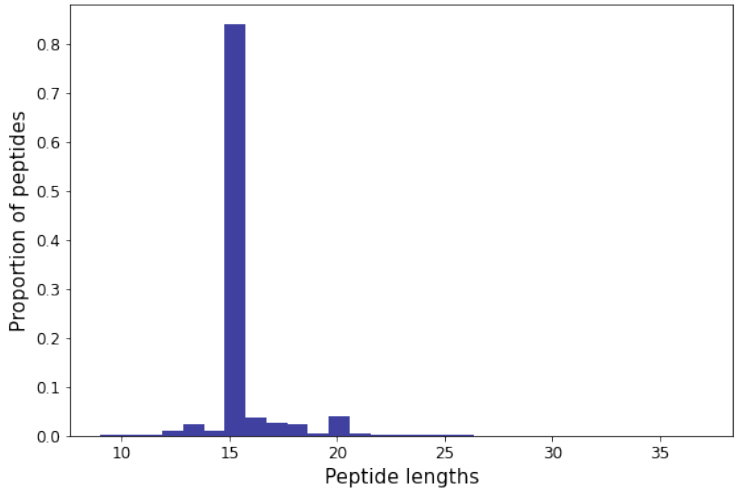


Figure 4.1: The histogram depicts the peptide length distributions for the dataset that was used to develop the transallelic version of the CNN-based model.

are shared across the 16 known MHC-II alleles in the training data for the transallelic model. We observe that unique peptides (i.e., peptides "shared" by 1 MHC-II allele) make up only 45.91% of the dataset, meaning 54.09% which is more than half of the peptides in the dataset are shared between 2 to 15 MHC-II alleles. Note that there are no peptides that are shared by all 16 alleles. This indicates a good dataset on which to develop the transallelic model.

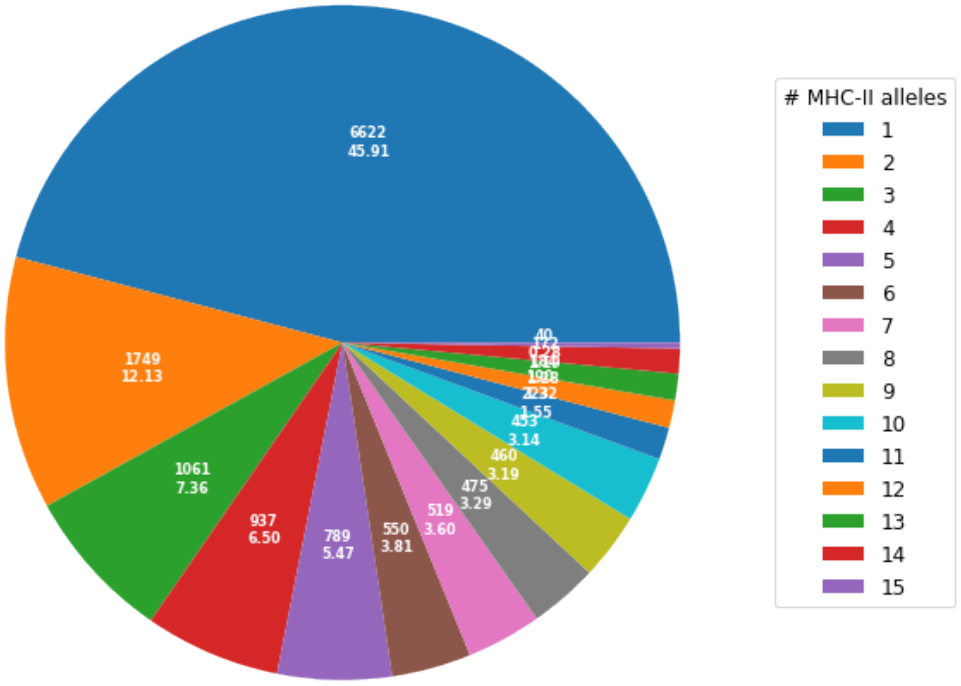


Figure 4.2: Pie chart showing the numbers and percentages of the unique ("shared" by 1 allele) and shared peptides (shared by 2-15 alleles, there are no peptides that are shared by all 16 alleles) across the 16 MHC-II alleles. The legend colour codes the number of MHC-II alleles that share peptides.

4.3 Results

The transallelic CNN model is implemented in three ways: nearest allelic neighbour (NN), leave-one-molecule-out (LOMO) and five-fold cross validation (FFCV). The predictive performance of the three models are compared with each other and results are reported in terms of AUC values in Table 4.3. The nearest allelic neighbour and the leave-one-molecule-out is reported for each allele and averaged out to compare overall performance with the five-fold cross validation model. The dataset for transallelic model consists of 77167 data points covering 16 human MHC-II alleles of the HLA-DR locus with binder peptide percentage of 46.26% (Table 2.3) with the minimum and maximum peptide lengths at 9 and 37 amino acid residues, respectively, and with 54.09% of peptides shared across 2-15 MHC-II of the 16 MHC-II alleles covered in the study.

The nearest allelic neighbour model requires that the similarity between all pairs of MHC-II alleles present in the available dataset be measured using the Hamming distance metric applied on sequences of the two MHC-II alleles of interest. The nearest neighbour model is tested on data for each MHC-II allele, which we refer to as the focal allele, and trained on data from the MHC-II allele most similar to the focal allele (Degoot *et al.*, 2018). The resulting model is then used on unknown MHC-II alleles by choosing the model trained on MHC-II alleles most similar to the unknown allele from the data for the available MHC-II alleles. Table 4.2 shows the similarity scores for every pair of 16 available MHC-II alleles covered in this study.

The similarity scores in Table 4.2 were computed using the Hamming distance metric. The first column lists the 16 MHC-II alleles that we use as focal alleles for which the nearest neighbour is tested. Every column after that lists the similarity score of the focal allele and all the other 15 available alleles. The values in bold indicate the highest similarity scores and occur in the column of the most similar allele for each focal allele. The last column lists the most similar allele for each of the 16 MHC-II alleles.

The leave-one-molecule-out model is tested on each of the 16 MHC-II alleles and trained on the concatenated data from the remaining 15 MHC-II alleles. The five fold cross validation is done in the same manner as in the allele-specific model. The five folds for each of the 16 MHC-II alleles are retained and each fold is concatenated over all the alleles to create the folds for the training data for the FFCV implementation of the transallelic model. The results of the comparison of the predictive performance of the NN and LOMO implementations of the transallelic model against the allele-specific model are reported for each allele in Table 4.3. The FFCV implementation of the transallelic model is also compared with the average performance of the allele-specific model for all 16 MHC-II alleles.

FFCV achieves an AUC value of 0.787 at 6 PFRs. The average AUC value for the allele-specific model over all 16 MHC-II alleles is 0.799. It may not be a fair comparison to make as the average AUC result for allele-specific model is the average over the AUC values reported for all 16 alleles and not the AUC value for the average model. Nonetheless, the average AUC value for the allele-specific model performance is better than the FFCV implementation of the transallelic model; however, FFCV still yields an encouraging model performance which is close to the average performance for the allele-specific model.

LOMO outperforms the allele-specific CNN model for 6 out of 16 MHC-II alleles with AUC values ranging from 0.684 for DRB1_0301 to 0.863 for DRB1_0701. NN performs with AUC values ranging from 0.468 for DRB3_0101 to 0.808 for DRB1_0901. NN yields good performance with AUC values above 0.7 for 12 out of 16 alleles. For allele DRB3_0101, NN yields an AUC value less than 0.5, mean-

ing that it performs no better than a random classifier for this allele. This is a cause for concern. Otherwise, while the allele-specific model outperforms the transallelic model overall, the transallelic model still yields encouraging performance as a foundation from which to keep developing and improving its performance.

An interesting observation is that the PFRs at which the transallelic model of the NN and LOMO implementations performs best is much smaller than for the allele-specific model for most alleles. This can indicate that with the increase in dataset size, the model reaches optimal performance earlier than the case for the allele-specific model. This can be an added consideration for PFR investigations for transallelic models. NN reports optimal performance at 3 PFRs and less for 8 out of 16 alleles and LOMO for 13 out of 16 alleles. What is surprising is that this is not the case for the FFCV. This may indicate a further examination of the factors that influence the performance of the FFCV implementation of the transallelic model.

The transallellic CNN model shows the most encouraging results for the LOMO implementation. This indicates that LOMO may be the best implementation of the transallelic model, which is good as a starting point for the development of the model. In any event the transallelic CNN model has yielded promising results as a foundation. The model is intended to be improved upon in time in order to be competitive in the community as was demonstrated by its allele-specific counterpart.

Table 4.2: Measurements of closeness between all sequence pairs of the 16 MHC-II alleles considered in this study using the Hamming Distance metric.

Focal allele	1_0101	1_0301	1_0401	1_0404	1_0405	1_0701	1_0802	1_0901	1_1101	1_1201	1_1302	1_1501	3_0101	3_0301	4_0101	5_0101	Closest allele
DRB1_0101	1.000	0.910	0.932	0.932	0.932	0.921	0.921	0.932	0.921	0.906	0.917	**0.947**	0.902	0.906	0.887	0.914	DRB1_1501
DRB1_0301	0.910	1.000	0.914	0.914	0.906	0.883	0.944	0.891	0.959	0.940	**0.970**	0.914	0.921	0.929	0.853	0.861	DRB1_1302
DRB1_0401	0.932	0.914	1.000	0.992	**0.992**	0.910	0.925	0.914	0.925	0.887	0.921	0.925	0.910	0.914	0.880	0.898	DRB1_0405
DRB1_0404	0.932	0.914	0.992	1.000	**0.992**	0.910	0.925	0.914	0.925	0.895	0.917	0.929	0.902	0.914	0.887	0.898	DRB1_0405
DRB1_0405	0.932	0.906	0.992	**0.992**	1.000	0.914	0.925	0.917	0.925	0.891	0.917	0.921	0.906	0.910	0.880	0.898	DRB1_0404
DRB1_0701	0.921	0.883	0.910	0.910	0.914	1.000	0.902	**0.947**	0.898	0.895	0.898	0.906	0.906	0.914	0.865	0.887	DRB1_0901
DRB1_0802	0.921	0.944	0.925	0.925	0.925	0.902	1.000	0.906	**0.974**	0.944	0.962	0.914	0.891	0.902	0.872	0.891	DRB1_1101
DRB1_0901	0.932	0.891	0.914	0.914	0.917	**0.947**	0.906	1.000	0.902	0.891	0.898	0.902	0.906	0.898	0.883	0.902	DRB1_0701
DRB1_1101	0.921	0.959	0.925	0.925	0.925	0.898	0.974	0.902	1.000	0.947	**0.981**	0.921	0.891	0.902	0.861	0.891	DRB1_1302
DRB1_1201	0.906	0.940	0.887	0.895	0.891	0.895	0.944	0.891	0.947	1.000	**0.955**	0.898	0.887	0.902	0.842	0.868	DRB1_1302
DRB1_1302	0.917	0.970	0.921	0.917	0.917	0.898	0.962	0.898	**0.981**	0.955	1.000	0.925	0.898	0.910	0.853	0.883	DRB1_1101
DRB1_1501	**0.947**	0.914	0.925	0.929	0.921	0.906	0.914	0.902	0.921	0.898	0.925	1.000	0.891	0.895	0.872	0.887	DRB1_0101
DRB3_0101	0.902	0.921	0.910	0.902	0.906	0.906	0.891	0.906	0.891	0.887	0.898	0.891	1.000	**0.966**	0.842	0.868	DRB3_0301
DRB3_0301	0.906	0.929	0.914	0.914	0.910	0.914	0.902	0.898	0.902	0.902	0.910	0.895	**0.966**	1.000	0.842	0.861	DRB3_0101
DRB4_0101	0.887	0.853	0.880	**0.887**	0.880	0.865	0.872	0.883	0.861	0.842	0.853	0.872	0.842	0.842	1.000	0.865	DRB1_0404
DRB5_0101	**0.914**	0.861	0.898	0.898	0.898	0.887	0.891	0.902	0.891	0.868	0.883	0.887	0.868	0.861	0.865	1.000	DRB1_0101

Table 4.3: Predictive performance of the transallelic CNN model in terms of AUC values for the three implementations, nearest allelic neighbour, leave-one-molecule-out and five fold cross validation.

Allele	#Peptides	Allele-specific	NN	LOMO
DRB1_0101	10412	0.781 (6)	0.727 (2)	**0.786** (2)
DRB1_0301	5352	**0.746** (5)	0.621 (2)	0.684 (1)
DRB1_0401	6317	**0.767** (5)	0.711 (5)	0.757 (0)
DRB1_0404	3657	0.755 (6)	0.711 (5)	**0.836** (0)
DRB1_0405	3962	0.776 (5)	0.760 (6)	**0.814** (2)
DRB1_0701	6325	0.833 (4)	0.781 (5)	**0.862** (1)
DRB1_0802	4465	**0.816** (5)	0.774 (2)	0.756 (1)
DRB1_0901	4318	0.797 (6)	0.808 (6)	**0.829** (2)
DRB1_1101	6045	**0.826** (4)	0.694 (2)	0.805 (0)
DRB1_1201	2384	**0.834** (6)	0.791 (1)	0.812 (0)
DRB1_1302	4477	**0.850** (6)	0.700 (4)	0.787 (0)
DRB1_1501	4850	0.796 (6)	0.759 (1)	**0.804** (0)
DRB3_0101	4633	**0.829** (4)	0.468 (3)	0.752 (6)
DRB3_0301	884	**0.780** (3)	0.573 (6)	0.738 (6)
DRB4_0101	3961	**0.800** (4)	0.744 (4)	0.761 (1)
DRB5_0101	5125	**0.804** (6)	0.744 (4)	0.788 (4)

The table shows the best performance for up to 6 PFRs for the transallelic model as tested on each of the 16 MHC-II alleles. The first column lists the 16 MHC-II alleles the model was tested on, and the second column shows the number of peptides belonging to each allele. The third, forth and fifth columns show results for the allele-specific, nearest allelic neighbour and leave-one-molecule-out models along with the PFR at which the maximum AUC values occur in parentheses. The values in bold are the best across the two transallelic model implementations and the allele-specific model for each allele.

Chapter 5

Conclusion

In this thesis, we successfully developed a novel CNN-based model to predict the binding interactions between peptides and MHC-II molecules belonging to the HLA-DR locus, the first of its type in the literature for predicting peptide-MHC-II binding interactions. The CNN-based model was created and continuously refined by experimentally adjusting neural network hyperparameters, investigating optimal input representations, and incorporating information from PFRs into the model's input. We discovered that the optimum approach to handle the input is to consider PFR information from both peptide termini (Table 3.8); moreover, to aggregate the stacked input tensor that encodes the entire peptide (Figure 2.2) along its depth for each number of PFRs.

The CNN-based model was evaluated to determine how processing the input that incorporates PFRs impacts the model's predictive performance. Including PFR information in the input translates into a more comprehensive model for peptide-MHC-II interactions. We discovered that PFRs play, in line with prior results, an important role in establishing a high-performing and efficient peptide-MHC-II predictive model.

The CNN-based model outperformed the latest version of the state-of-the-art allele-specific peptide-MHC-II interaction model, NetMHCII-2.3, across all MHC-II alleles considered. The results suggest that the allele-specific CNN-based model is competitive and suited to be extended to developing a transallelic model that allows for the prediction of binding interactions for unknown MHC-II alleles. The preliminary findings that resulted from implementations of a transallelic CNN-based model are encouraging (Table 4.3) and will serve as the foundation for future work. The CNN-based model, was only trained on one of the three loci of human MHC-II

molecules, HLA-DR. Alleles from the two additional loci of the human MHC-II molecules, HLA-DP and HLA-DQ, will be incorporated into future work to further expand the model's capacity. While the model incorporates information from PFRs and provides insights into the impact PFRs have on model performance, nothing is learned from the model's output about the biological reasoning behind the process by which MHC-II molecules choose which peptides to bind. More biophysical aspects may be included in future research to allow for conclusions that might lead to a better understanding of these binding interactions.

The performance of the allele-specific CNN-based model demonstrated remarkable predictive ability of the model. The model's results were also encouraging in motivating further development to predict for MHC-II alleles of the loci that were not included in the training of our CNN model. Furthermore, model's results highlighted the effect that including PFR information in the input has on optimizing model performance. Our findings indicate that PFRs should be incorporated in modelling these peptide-MHC-II binding interactions in order to result in a more comprehensive model with substantial predictive ability. Lastly, a transallelic model developed on the foundations of the success of the allele-specific CNN-based model has been demonstrated to be achievable and will be a crucial next step to widening the scope of the CNN model's task.

List of References

Bahrami, A.A., Payandeh, Z., Khalili, S., Zakeri, A. and Bandehpour, M. (2019). Immunoinformatics: In silico approaches and computational design of a multi-epitope, immunogenic protein. *International Reviews of Immunology*, vol. 38, no. 6, pp. 307–322.

Bordner, A.J. and Mittelmann, H.D. (2010). Multirta: A simple yet reliable method for predicting peptide binding affinities for multiple class ii mhc allotypes. *BMC bioinformatics*, vol. 11, no. 1, pp. 1–12.

Degoot, A.M., Chirove, F. and Ndifon, W. (2018). Trans-allelic model for prediction of peptide: Mhc-ii interactions. *Frontiers in immunology*, vol. 9, p. 1410.

DTU-Bioinformatics (). Supplementary material. http://www.cbs.dtu.dk/services/NetMHCIIpan-3.2/. Accessed: September 2019.

Fawcett, T. (2006). An introduction to roc analysis. *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874.

Géron, A. (2017). *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. 1st edn. O'Reilly Media, Inc. ISBN 1491962291, 9781491962299.

Gleichmann, N. (2020). Innate vs adaptive immunity. https://www.technologynetworks.com/immunology/articles/innate-vs-adaptive-immunity-335116. Accessed: November 2021.

Glen, S. (2021). Wilcoxon signed rank test: Definition, how to run, spss. https://www.statisticshowto.com/wilcoxon-signed-rank-test/. Accessed: September 2021.

Glorot, X., Bordes, A. and Bengio, Y. (2011 11–13 Apr). Deep sparse rectifier neural networks. In: Gordon, G., Dunson, D. and DudÃk, M. (eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, vol. 15 of *Proceedings of Machine Learning Research*, pp. 315–323. PMLR, Fort Lauderdale, FL, USA. Available at: https://proceedings.mlr.press/v15/glorot11a.html

Goodfellow, I., Bengio, Y. and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Google (). Google colaboratory notebooks. https://colab.research.google.com/.

Han, Y. and Kim, D. (2017). Deep convolutional neural networks for pan-specific peptide-MHC class I binding prediction. *BMC bioinformatics*, vol. 18, no. 1, p. 585.

Holland, C.J., Cole, D. and Godkin, A. (2013). Re-directing cd4+ t cell responses with the flanking residues of mhc class ii-bound peptides: The core is not enough. *Frontiers in Immunology*, vol. 4, p. 172. ISSN 1664-3224.
Available at: https://www.frontiersin.org/article/10.3389/fimmu.2013.00172

Hu, J. and Liu, Z. (2017). Deepmhc: Deep Convolutional Neural Networks for High-performance peptide-MHC Binding Affinity Prediction. *bioRxiv*, p. 239236.

IBM-Cloud-Education (2020). Convolutional neural networks. https://www.ibm.com/cloud/learn/convolutional-neural-networks. Accessed: July 2021.

IEDB (). Immuno epitope database. http://tools.iedb.org/main/datasets/. Accessed: September 2019.

Jensen, K.K., Andreatta, M., Marcatili, P., Buus, S., Greenbaum, J.A., Yan, Z., Sette, A., Peters, B. and Nielsen, M. (2018 01). Improved methods for predicting peptide binding affinity to MHC class II molecules. *Immunology*, vol. 154, no. 3, pp. 394–406.

Karosiene, E., Rasmussen, M., Blicher, T., Lund, O., Buus, S. and Nielsen, M. (2013 10). Netmhciipan-3.0, a common pan-specific mhc class ii prediction method including all three human mhc class ii isotypes, hla-dr, hla-dp and hla-dq. *Immunogenetics*, vol. 65, no. 10, pp. 711–724.

Kumar, V. and McNerney, M.E. (2005). A new self: Mhc-class-i-independent natural-killer-cell self-tolerance. *Nature Reviews Immunology*, vol. 5, no. 5, pp. 363–374.

Lin, H., Zhang, G., Tongchusak, S., Reinherz, E. and Brusic, V. (2008 12). Evaluation of mhc-ii peptide binding prediction servers: Applications for vaccine research. *BMC bioinformatics*, vol. 9 Suppl 12, p. S22.

Liu, Z., Cui, Y., Xiong, Z., Nasiri, A., Zhang, A. and Hu, J. (2019). Deepseqpan, a novel deep convolutional neural network model for pan-specific class i hla-peptide binding affinity prediction. *Scientific reports*, vol. 9, no. 1, pp. 1–10.

Liu, Z., Jin, J., Cui, Y., Xiong, Z., Nasiri, A., Zhao, Y. and Hu, J. (2021). Deepseqpanii: an interpretable recurrent neural network model with attention mechanism for peptide-hla class ii binding prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*.

Murphy, K., Travers, P., Walport, M. and Janeway, C. (2008). *Janeway's Immunobiology*. New York: Garland Science.

Nielsen, M. and Lund, O. (2009). NN-align. An artificial neural network-based alignment algorithm for MHC class II peptide binding prediction. *BMC bioinformatics*, vol. 10, no. 1, p. 296.

Nielsen, M., Lund, O., Buus, S. and Lundegaard, C. (2010). Mhc class ii epitope predictive algorithms. *Immunology*, vol. 130, no. 3, pp. 319–328.

Nielsen, M.A. (2015). *Neural Networks and Deep Learning*. Determination Press. Available at: http://neuralnetworksanddeeplearning.com/

Reynisson, B., Alvarez, B., Paul, S., Peters, B. and Nielsen, M. (2020 05). NetMHCpan-4.1 and NetMHCIIpan-4.0: improved predictions of MHC antigen presentation by concurrent motif deconvolution and integration of MS MHC eluted ligand data. *Nucleic Acids Research*, vol. 48, no. W1, pp. W449–W454. ISSN 0305-1048. https://academic.oup.com/nar/article-pdf/48/W1/W449/33433048/gkaa379.pdf.
Available at: https://doi.org/10.1093/nar/gkaa379

Robinson, J., Halliwell, J.A., McWilliam, H., Lopez, R. and Marsh, S.G.E. (2012 11). Ipd: the immuno polymorphism database. *Nucleic Acids Research*, vol. 41, no. D1, pp. D1234–D1240. ISSN 0305-1048. https://academic.oup.com/nar/article-pdf/41/D1/D1234/3641596/gks1140.pdf.
Available at: https://doi.org/10.1093/nar/gks1140

Rudolph, M.G., Stanfield, R.L. and Wilson, I.A. (2006). How tcrs bind mhcs, peptides, and coreceptors. *Annual Review of Immunology*, vol. 24, no. 1, pp. 419–466. https://doi.org/10.1146/annurev.immunol.23.021704.115658.
Available at: https://doi.org/10.1146/annurev.immunol.23.021704.115658

Shao, X.M., Bhattacharya, R., Huang, J., Sivakumar, I.A., Tokheim, C., Zheng, L., Hirsch, D., Kaminow, B., Omdahl, A., Bonsack, M. *et al.* (2020). High-throughput prediction of mhc class i and ii neoantigens with mhcnuggets. *Cancer immunology research*, vol. 8, no. 3, pp. 396–408.

Tomar, N. (2020). *Immunoinformatics*. Methods in Molecular Biology. Springer US. ISBN 9781071603888.

Traherne, J. (2008). Human mhc architecture and evolution: implications for disease association studies. *International journal of immunogenetics*, vol. 35, no. 3, pp. 179–192.

Yanminsun, Wong, A. and Kamel, M.S. (2011 11). Classification of imbalanced data: a review. *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 23.

Zhang, L., Udaka, K., Mamitsuka, H. and Zhu, S. (2012). Toward more accurate pan-specific mhc-peptide binding prediction: a review of current methods and tools. *Briefings in bioinformatics*, vol. 13, no. 3, pp. 350–364.