Machine Learning approaches for the identification and exploitation of Sequence Motifs in Immunopeptidomes

Herramientas de Aprendizaje Automático para la identificación y explotación de Motivos de Secuencia en Inmunopeptidomas

PhD Thesis

Bruno Alvarez

Advisor: Dr. Morten Nielsen

I I B I O

Instituto de Investigaciones Biotecnológicas
Universidad Nacional de San Martín
Buenos Aires, Argentina

August 2021

Contents

Prefacio

La presente tesis doctoral fue realizada en el grupo de Inmunoinformática y Machine Learning del Instituto de Investigaciones Biotecnológicas "Dr. Rodolfo A. Ugalde" (IIBIO), como requisito para optar por el título de Doctor en Biología Molecular y Biotecnología de la Universidad Nacional de San Martín (UNSAM).

El trabajo presentado en esta tesis fue realizado entre Abril de 2016 y Agosto de 2021 bajo la dirección del Dr. Morten Nielsen. Este doctorado fue financiado con una beca del Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET).

CABA, Buenos Aires, Argentina
Agosto 2021

Bruno Alvarez

Agradecimientos

Uf, que viaje! Cinco años y medio, una pandemia, algún que otro ataquecito de pánico. Café, mate, ~~alplax~~ y un doctorado. Cuantas cosas para contar. Pero mejor lo dejamos para un bar, que así en texto es medio aburrido y larguero.

Mejor agradezcamos.

En primer lugar, a mi director, Morten. Sencillamente, gracias por todo. Fue un honor y un placer haber trabajado con vos todos estos años. Tu generosidad a la hora de enseñar es infinita, y te volvería a elegir sin dudarlo. Also, looking forward for what's next.

A la gente maravillosa que iluminó mis días en el labo: Lio, Leo, Ale, Mass, Carol, Ibe, Feno, Lanza, Heli, Juli, Lean, Fernan, Emir. Gracias por los almuerzos, las charlas, las cervezas, las risas, la compañía, el aprendizaje. Me los llevo pa siempre.

To the wonderful people of Denmark I had the pleasure to meet. Vanessa, Martin, Kamilla, Birkir: skål! Thanks for all the moments, I will remember them fondly.

A mi hermosa familia. A mi viejo, por demostrarme con el ejemplo que la vida siempre sigue adelante. A mi vieja, por estar cada vez más presente dentro mío. A mi hermanita, Ari, cada año te quiero más. A mi ahijado, Joaquito, voy a estar siempre. A Guille, gracias por estar, y por cuidar. A mi tía Beatriz, siempre presente con su línea directa. A mi tío Rolo, por toda la buena onda. A Mario, Diana y Tomi, por la compañía y los buenos momentos. A Gerardito y Mara, que a pesar de que no compartimos sangre, siento lo contrario.

A mis amigos de toda la vida en todos sus días, Fer y Curi. Forever bros.

A los grandes amigos que no veo tan seguido, pero recuerdo siempre. Maurito, Cris, Roma, Agus, Flor, Solchi, Tom, Paulita, Marian, Pasca, Charly, Pablito. Gracias por ser.

A los que ya no están: gracias por haber estado. Los recuerdo a todxs, son parte de lo que soy.

Por último, a mi compañera de viaje, Coni. Gracias por tanto amor. Conocerte fue lo más hermoso que me pasó en la vida. Te elijo todos los días, y ojalá que el universo nos regale muchos años más juntos. <3.

Bueno che, nos vemos por ahí :=)

Bruno.

Πάντα ῥεῖ
Panta Rei

Resumen

Las proteínas son moléculas de primordial importancia en, virtualmente, todos los procesos celulares que sustentan la vida. Su relevancia se debe, principalmente, a su capacidad de ejercer una vasta lista de funcionalidades por medio de la unión y/o interacción selectiva con otras moléculas. Estas interacciones ocurren en interfaces proteína-molécula específicas, de naturaleza tridimensional, y se caracterizan por poseer una dinámica de tipo llave-cerradura. Más allá de que es correcto asumir que dichas interacciones son en esencia complejas, en muchos casos contienen pequeños componentes lineales, y por lo tanto pueden ser aproximadas por medio de una interacción de tipo péptido-proteína.

Un caso particular de interacción péptido-proteína es la unión de péptidos al Complejo Mayor de Histocompatibilidad (MHC en inglés). El MHC posee un rol clave en el sistema inmune adaptativo de los vertebrados, principalmente gracias a su capacidad de unión y presentación de péptidos antigénicos al espacio extracelular. Luego de dicha presentación, Linfocitos T pueden interactuar con estos MHCs y, si se satisfacen ciertas condiciones, desencadenar una respuesta inmune.

De lo antedicho, se torna evidente que existe una fuerte relación entre la inmunidad de los vertebrados y el conjunto de todos los posibles ligandos de MHC. En los últimos años, a dicho conjunto se lo ha denominado Inmunopeptidoma, y el consecuente desarrollo de herramientas científicas para su muestreo e interpretación ha dado a luz al campo de la Inmunopeptidómica.

El trabajo aquí presentado comprende el desarrollo de diversas herramientas computacionales de Inmunopeptidómica, en la forma de algoritmos y procesos de Aprendizaje Automático, capaces de ser implementados en la identificación y explotación de la información contenida en un Inmunopeptidoma de interés.

Summary

Proteins are molecules of paramount importance in virtually all cellular processes sustaining life. Their relevance rests, to a high degree, in their capacity of exerting a vast range of functionalities by means of selectively binding to (and/or interacting with) other molecules. These interactions occur in specific three dimensional protein-molecule interfaces, and are characterised by a key-lock type of mechanism. While it is safe to assume that such interactions are complex in nature, in many cases they contain a short linear component, and may be approximated by means of a protein-peptide interaction.

A particular case of protein-peptide interactions is the binding of peptides to the Major Histocompatibility Complex (MHC) protein. The MHC is a key player in the adaptive cellular immune system of vertebrates, mostly thanks to its capability of binding and presenting antigenic peptides to the extracellular space. After such presentation, T lymphocytes may interact with the loaded MHCs and, if certain conditions are met, an immune response might be fired.

From what is stated above, it becomes clear that a strong bond exists between the immunity of vertebrates and the set of all possible MHC binding peptides. In recent years, such a set has been termed the Immunopeptidome, and the consequential development of scientific tools that enable its sampling and interpretation has given birth to the field of Immunopeptidomics.

The work presented in this manuscript comprehends the development of several in-silico scientific tools for Immunopeptidomics, shaped in the form of Machine Learning algorithms and pipelines that can be readily deployed to identify and exploit the information contained within a target Immunopeptidome.

Organization of this Thesis

In the **first chapter** of this thesis, basic concepts of Immunology are briefly described. The role of MHC and the immunopeptidome in the context of the adaptive immune response are also addressed. Then, the field of immunopeptidomics and its importance in the sampling of immunopeptidomes are introduced. Different aspects of Machine Learning (ML) are afterwards discussed, with a special emphasis on Artificial Neural Networks (ANNs) training and validation. Finally, the utilization of ANNs in the task of predicting peptide binding to MHC, under the paradigm of Immunological Bioinformatics, is described.

The **second chapter** exhibits how two in-house ML algorithms (GibbsCluster and NNAlign) can be applied jointly in order to extract peptide-MHC binding motifs from immunopeptidomics datasets and to train models that enable the prediction of peptide binding to MHC.

The **third chapter** presents NNAlign_MA, a novel artificial neural network algorithm that extends the capabilities of NNAlign and drastically improves the identification of MHC binding motifs from immunopeptidomic datasets, while also boosting peptide-MHC binding predictions.

The **fourth chapter** introduces NetMHCpan-4.1 and NetMHCIIpan-4.0, two state-of-the-art ANN models trained to predict peptide-MHC binding interactions, built on top of the NNAlign_MA engine, which outperformed competitors and were released to the scientific community as public web-servers.

The **fifth chapter** dives into the Deep Learning field in order to explore possible alternative approaches for the task of MHC motif discovery. To do so, an original take on 1-dimensional convolutional neural networks is deployed and benchmarked for peptide-MHC binding data.

Taken as a whole, the present thesis intends to provide a series of Machine Learning algorithms, pipelines and insights for the analysis of immunopeptidomics data, both from the perspective of binding motifs characterization and prediction of peptide-MHC binding interactions.

List of publications

- **Publications included in this Thesis:**
  - **Computational Tools for the Identification and Interpretation of Sequence Motifs in Immunopeptidomes**. Bruno Alvarez[†], Carolina Barra[†], Morten Nielsen, Massimo Andreatta. *Proteomics (January 2018).* doi: `https://doi.org/10.1002/pmic.201700252`
  - **NNAlignMA; MHC Peptidome Deconvolution for Accurate MHC Binding Motif Characterization and Improved T-cell Epitope Predictions**. Bruno Alvarez, Birkir Reynisson, Carolina Barra, Søren Buus, Nicola Ternette, Tim Connelley, Massimo Andreatta, Morten Nielsen. *Molecular & Cellular Proteomics (December 2019).* doi: `https://doi.org/10.1074/mcp.TIR119.001658`.
  - **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**. Birkir Reynisson[†], Bruno Alvarez[†], Sinu Paul, Bjoern Peters, Morten Nielsen. *Nucleic Acids Research Web Server Issue (May 2020).* doi: `https://doi.org/10.1093/nar/gkaa379`.
- **Publications not included in this Thesis:**
  - **GibbsCluster: unsupervised clustering and alignment of peptide sequences**. Massimo Andreatta[†], Bruno Alvarez[†], Morten Nielsen. *Nucleic Acids Research Web Server Issue (April 2017).* doi: `https://doi.org/10.1093/nar/gkx248`.
  - **Footprints of antigen processing boost MHC class II natural ligand predictions**. Carolina Barra[†], Bruno Alvarez[†], Sinu Paul, Alessandro Sette, Bjoern Peters, Massimo Andreatta, Søren Buus, Morten Nielsen. *Genome Medicine (November 2018).* doi: `https://doi.org/10.1186/s13073-018-0594-6`.

[†] Joint first authorship

Abbreviations

BCR	B Cell Receptor
TCR	T Cell Receptor
MHC	Major Histocompatibility Complex
MHC-I	Major Histocompatibility Complex Class I
MHC-II	Major Histocompatibility Complex Class II
APC	Antigen Presenting Cell
HLA-I	Human Leukocyte Antigen Class I
HLA-II	Human Leukocyte Antigen Class II
BA	Binding Affinity data
EL	Eluted Ligands data
MS	Mass Spectrometry
ESI	Electrospray Ionization
HPLC	High-Performance Liquid Chromatography
FDR	False Discovery Rate
MA	Multi Allele EL data
SA	Single Allele EL data
ML	Machine Learning
AI	Artificial Intelligence
ANN	Artificial Neural Network
FFNN	Feed Forward Neural Network
CNN	Convolutional Neural Network
DL	Deep Learning
MSE	Mean Squared Error
GD	Gradient Descent
BP	Backpropagation
AD	Automatic Differentiation
CV	Cross Validation
HP	Hyperparameter
RMSE	Root Mean Squared Error
PCC	Pearson Correlation Coefficient
SCC	Spearman Correlation Coefficient
TPR	True Positive Rate
TNR	True Negative Rate
FPR	False Positive Rate
FNR	False Negative Rate
PPV	Positive Predictive Value
ACC	Accuracy
ROC	Receiver Operating Characteristic curve
PRC	Precision Recall Curve
AUC	Area Under Curve
PSSM	Position Specific Scoring Matrix
BLOSUM	Blocks Substitution Matrix
CL	1-dimensional convolutional layer
GMP	Global Max Pool

Notes

- For size concerns, large multi-page figures and tables were formatted to fit into single pages. Such objects become perfectly legible with the right amount of zoom.
- For the above, we suggest avoiding default pdf viewers (such as Ubuntu's Okular) since they are not quite optimized, and might cause hanging. A more modern and suitable viewer is, for example, WPS PDF.
- Tables which span several hundreds of lines (>300) and/or have multiple sheets were excluded from this manuscript's body of text, but can be accessed using the provided hyperlinks.

Chapter 1

Introduction

1.1 The Immune System

The immune system is a complex network of effector cells and molecules committed to the protection of the body against invading microorganisms. It is composed of two branches: innate and adaptive immune system, which differ in the specificity of their responses against invasion, their speed and their mechanisms of action.

The innate immune system represents the first line of defense against pathogens. It can be activated very rapidly on exposure to an infectious organism, and is essentially made up of non-specific, sequential barriers that aim to destroy viruses, bacteria, parasites and fungi before they are able to spread further. The first barrier is anatomical (i.e. epithelial surfaces), the second is chemical (i.e. the complement system) and the third corresponds to innate immune cells, such as macrophages, granulocytes and natural killer cells [1].

The adaptive immune system, also called acquired immunity, represents the second line of defense against pathogens. Unlike the innate branch (which operates based on the identification of general threats) the adaptive immunity is activated by specific exposure to pathogens, and uses immunological memory to learn about the threat and enhance the immune response against it [1]. To do this, the adaptive branch relies on adaptive immune cells called lymphocytes, that bind to antigens (molecules that stimulate the immune system) on specific sub-sections (called epitopes) using their antigen receptors. Each lymphocyte found in a particular host matures to carry an unique variant of a prototypical antigen receptor. The premise is that, among the billions of lymphocytes circulating in the body at a given time, there will always be some capable of recognizing a given foreign antigen and start a course of action against it.

Lymphocytes are categorised into B lymphocytes and T lymphocytes (see Figure 1.1), and have the capability of detecting and binding epitopes through their B Cell Receptors (BCR) and T Cell Receptors (TCR), respectively [2]. Both B cells and T cells undergo somatic rearrangements of their DNA to form clones with receptors of unique sequence and binding specificity [3]. However, most developing B and T lymphocytes are killed off in the process of positive and negative selection, where cells with TCRs or BCRs that have potential for ligand binding receive signals for survival, whereas cells reacting strongly to self antigens do not [4]. Cells that survive this process go on to survey for infection as naive lymphocytes that, upon activation, will proliferate, develop into effector cells and enforce their role in immunity. A subset of these activated cells will form long-lived memory cells, with a rapid reaction time against "memorized" antigens [5]. Generally speaking, the ultimate role of B and T cells is similar (detecting and removing offending entities), but their mode of action is not. Because of this, adaptive immunity can be further divided into two sub-branches: humoral immunity and cell-mediated immunity.

Humoral immunity is mediated by B cells, which bind to the surface of antigens in the

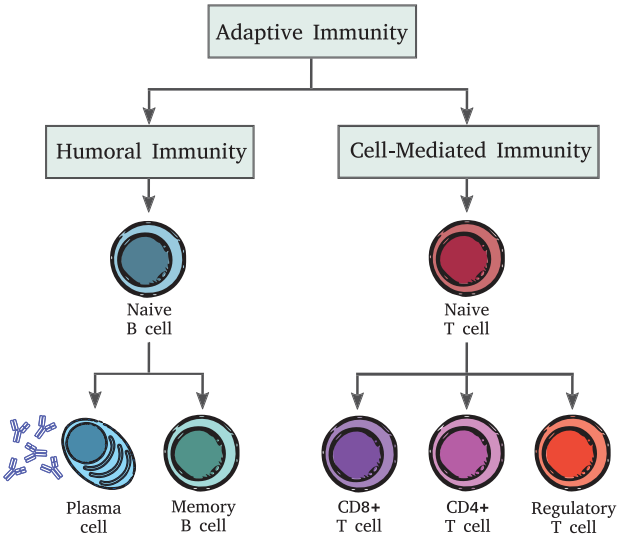

Figure 1.1. Categories of the cell lines involved in the adaptive immune response.

extracellular space using their BCRs [6]. Once a B cell becomes activated in the presence of their target antigen, it becomes a plasma cell and begins to produce and secrete large amounts of antibodies that can bind to the target antigen and neutralize it [7].

On the other hand, cell-mediated immunity is driven by T cells and their interactions with pathogens through their TCRs. T cells can be divided into two groups depending on their expression of either cell-surface CD4 or CD8 receptors. CD8+ T cells are commonly known as cytotoxic T lymphocytes, because once they strongly bind to a target cell they secrete cytotoxic granules and perforin, which penetrate the cell's membrane and induce apoptosis. Conversely, CD4+ T cells are commonly referred to as helper T cells, because after binding to target cells they play an important role in contributing to the cytokine response that stimulates either cell-mediated immunity or humoral immunity [7]. Another form of T cells are the regulatory T cells that have roles in dampening immune responses against self, which is a form of tolerance [8].

From what is shown above, it can be seen that for a T cell's effector function to become executed, it first needs to bind a target cell. In a general sense, this means that such T cell must have some kind of molecular target to bind to. Specifically, this target is the Major Histocompatibility Complex (MHC) molecule, which is in charge of presenting intracellular protein fragments (peptides) to the extracellular space [9]. MHC and its bound peptide serve as a sort of control flag in the immune system program, since they transfer information related to the internal state of cells to the space surrounding them. Because of this, the immune synapse between T lymphocytes and MHC is of paramount importance, to the point that if it is strong enough -and some other conditions are met- an immune response might be fired.

The nature of such immune synapse can be first and foremost understood based on the type of interacting T lymphocyte. CD8+ T cells bind to and recognize peptides bound to MHC class I (MHC-I) molecules. On the other hand, CD4+ T cells establish immune synapses with peptide-loaded MHC class II (MHC-II) molecules [10]. However, and independently of the type of CD expressed on the T cell's surface, it is clear that a strong relationship exists between the peptide-MHC complex, T cells and the immune response.

At the end of the day, the binding of peptides to MHC represents a necessary condition for the activation of cellular immune responses, since such binding always needs to happen prior to a possible T lymphocyte synapse. Given this intrinsic importance, the following section will introduce key concepts related to MHC and its peptide interactions.

1.2 The Major Histocompatibility Complex

The Major Histocompatibility Complex is a large genetic complex composed of multiple loci that encodes for three major classes of membrane-bound glycoproteins: class I, class II, and class III MHC molecules. Class I and II MHC molecules bind to a spectrum of antigenic peptides derived from the intracellular processing and degradation of antigen molecules and present them to the extracellular space [11]. On the other hand, class III MHC are poorly defined structurally and functionally. They are not involved in antigen presentation, and only a few of them are actually involved in immunity while many are signalling molecules in other cell communications [12]. Given this, this work will focus on the MHC class I (MHC-I) and MHC class II (MHC-II) molecules.

MHC-I is expressed by all nucleated cells, and presents peptides derived from intracellular degradation of endogenous proteins by the proteasome and other peptidases (Figure 1.1). Cytosolic peptides become naturally degraded for reutilization but some of them are transported into the Endoplasmic Reticulum by the TAP transporter [13], where empty MHC-I molecules are then loaded with such peptides and transported to the cell surface [14]. Given this, peptides presented by MHC-I molecules represent a state-of-self for CD8+ T lymphocytes [15]. On the other hand, MHC-II is expressed by Antigen Presenting Cells (APC), like dendritic cells, B-cells and macrophages [16], and presents peptides derived from the enzymatic digestion of proteins taken up from extracellular space, which helps regulating how T cells respond to an infection [17]. In particular, extracellular proteins become endocytosed into vesicles and then merged with lysosomes, resulting in antigen degradation by the acidic environment and wide variety of proteases of such lysosome [18].

All processed antigenic peptides bind to MHC by means of accommodating into its molecular binding groove. The MHC binding platform is composed of two domains, which originate from a single heavy chain (α-chain) plus a β2-microglobulin in the case of MHC class I and from two chains (α-chain and β-chain) in the case of MHC class II (Figure 1.3, panels A and B); both domains are anchored to the cell's membrane by transmembrane helices. Such MHC domains have evolved to form a slightly curved -sheet as a base and two -helices on top, which are far enough apart to fit a peptide chain in between [19]. This latter conformation is known as the MHC binding pocket, and is where peptides bind according to: (1) the formation of a set of conserved hydrogen bonds between the side-chains of the MHC molecule and the backbone of the peptide; and (2) the occupation of defined pockets by peptide side chains [20, 21]. Item (1) represents an unspecific binding interaction, and serves mainly to stabilize the peptide backbone to the binding cleft; on the other hand, (2) is a specific peptide-MHC binding interaction, and is a function of the peptide composition (amino acid side-chains and positions) and the geometry, charge distribution, and hydrophobicity of the MHC binding groove.

In MHC class I, such binding groove is closed at both ends by conserved tyrosine residues that lead to a length restriction of the bound peptides to an average of 8-10 amino acids [23–25] (Figure 1.3, panel C). MHC-I anchor positions are usually located at positions 2 (P2), P5/6 and P9 [20]. In contrast, MHC class II proteins usually accommodate peptides of 13-25 residues in length in their open binding cleft [26] (Figure 1.3, panel D), and anchor positions are generally located at P1, P4, P6, and P9 [21]. For a given MHC molecule, the interaction preferences of its binding pocket largely determines the peptide binding specificity of such MHC [27]. Also, given the limited length of the MHC pocket, such preferences can be represented as linear motifs and visualized with so-called sequence logos [28]. To generate these, one may collect a list of binding peptides, align them and convert positional amino acid frequency into an information content representation, as shown in Figure 1.4. Doing this allows to capture and display MHC binding profiles in an intuitive and clear way.

The repertoire of MHC binding preferences is vast and diverse, principally thanks to the high variation in the residues defining the binding groove between different MHCs. In essence, such variance occurs because MHC is: (1) polygenic, since it contains several different MHC-I and MHC-II genes, and this means every individual possesses a set of MHC molecules with variable ranges of peptide-binding specificities; and (2) highly polymorphic, which means there are multiple variants -or alleles- of each gene within the whole population [1]. As an example, a sample individual from the human population will have three pairs

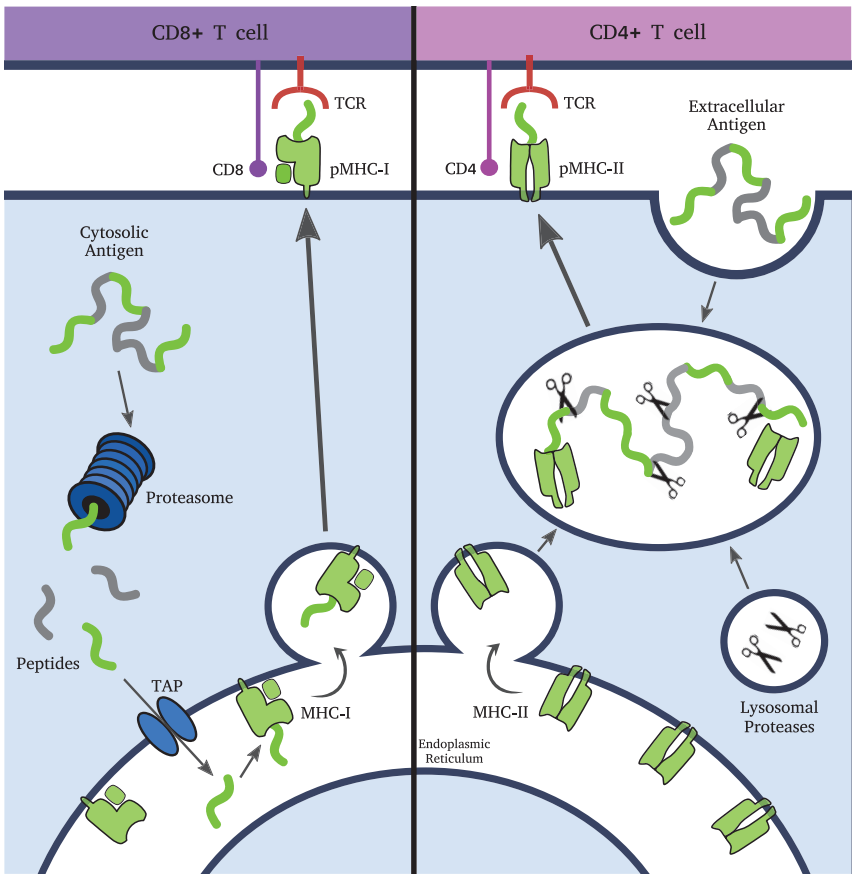

Figure 1.2. MHC antigen presentation pathways. For MHC-I (left panel), cytosolic antigens are digested by the Proteasome and transported to the Endoplasmic Reticulum (ER) through TAP. Inside the ER, MHC-I molecules are loaded with peptides -forming the pMHC-I complex- and shipped to the cell membrane for presentation to CD8+ T cells. On the other hand, MHC-II molecules (right panel) bind to peptides derived from the enzymatic digestion of engulfed extracellular antigens. After binding, the pMHC-II complex is transported to the cell membrane for CD4+ T cell presentation.

of inherited Human Leukocyte Antigen Class I (HLA-I, or human MHC-I) alleles, most commonly spanning HLA-A, HLA-B, and HLA-C; similarly, the same individual will have up to 7 pairs of inherited Human Leukocyte Antigen Class II (HLA-II, or human MHC-II) alleles, most commonly spanning HLA-DRA, HLA-DRB1, HLA-DR3,4,5, HLA-DQA, HLA-DQB, HLA-DPA, and HLA-DPB [30]. As a result, such individual will express up to 6 MHC-I and up to 12 different MHC-II allelic variants (HLA-DRA being monomorphic), depending on the level of heterozygosity. The combinatorial space from which these 6- and 12-groups are generated is extense: as of May 2021, a total of 12.995 HLA-I and 5.248 HLA-II protein sequences are available in the IPD-IMGT/HLA database [31]. This is, a total of more than 18.000 known human MHC molecules, each one contributing to an unique and specific peptide binding preference, and thus capable of scanning and selecting different subsets of natural peptides for antigen presentation to T cells.

If we think of vertebrate immunity as an evolution race between hosts (which evolve to defend) and invaders (which evolve to evade), it makes sense for this extreme MHC variability to exist as a way of dealing with the huge space of potential antigenic peptides nature has to offer [32]. As a result, any given MHC will respond to a “preferred” subspace of the aforementioned space, and for a particular MHC-expressing cell, a specific combination of these subspaces will become its particular immune signature, which we will call immunopeptidome.

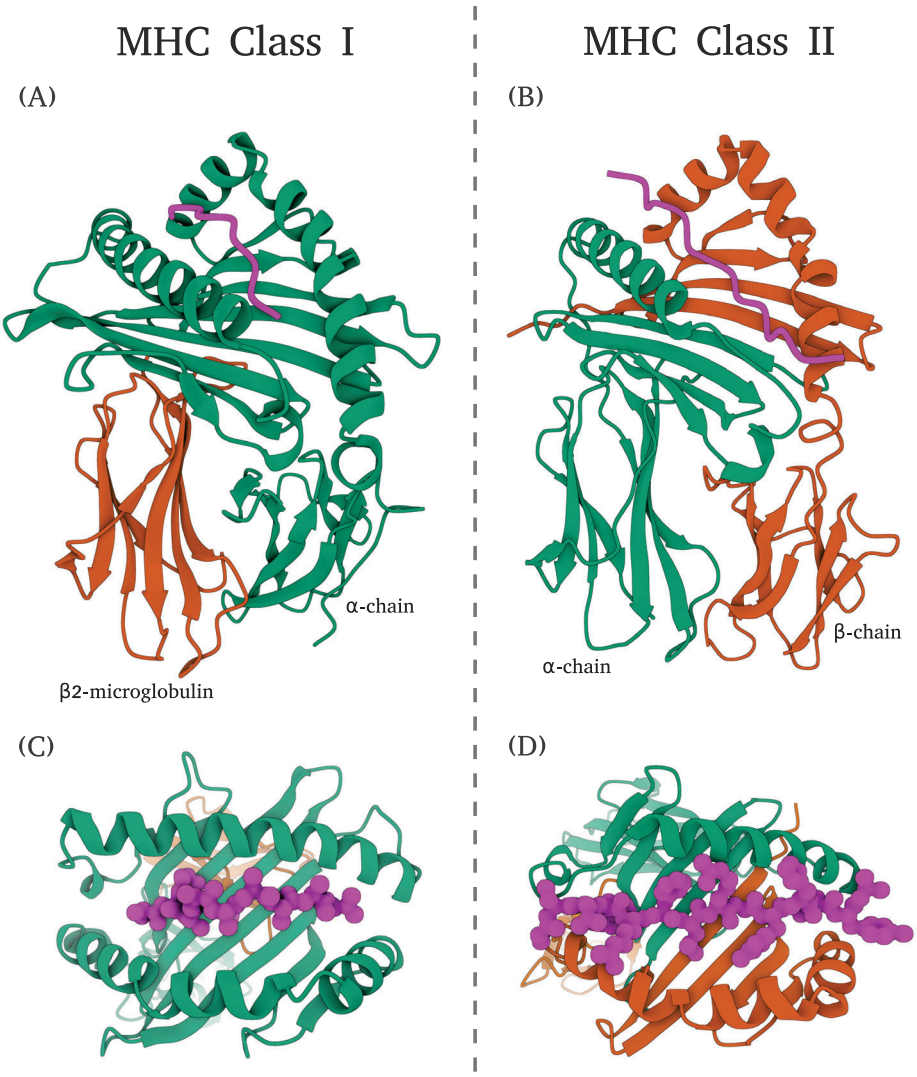

Figure 1.3. Structural representations of MHC-I and MHC-II proteins in complex with peptides (in pink). (A) Structure of an MHC-I molecule, with α-chain colored green and β2-microglobulin colored orange. (B) Structure of an MHC-II molecule, with α-chain colored green and β-chain colored orange. (C) Top view of MHC-I binding groove, with anchored ligand shown in volumetric representation. (D) Top view of MHC-II binding cleft, with bound peptide shown in volumetric representation. All 3D representations were generated using Mol* Viewer [22].

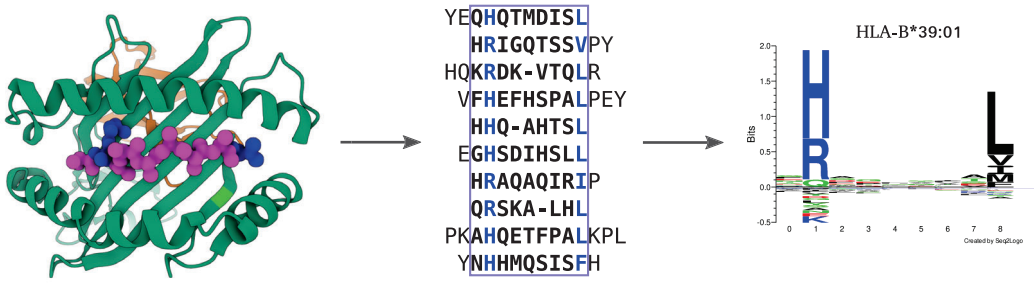

Figure 1.4. MHC binding motif visualization. In the left panel, the top view of an MHC-I binding groove accomodating a peptide (in pink) is shown; anchor residues are displayed in blue. Sequencing and aligning several ligands (middle panel) can be exploited to extract a binding core. Then, the position-specific information content from such core is used to generate a sequence logo (right panel) representing the aminoacidic binding preferences of the corresponding MHC (in this example, the human MHC-I HLA-B*39:01 is displayed). The MHC structural representation was generated using Mol* Viewer [22]; the sequence logo was created using Seq2Logo [29].

1.3 The Immunopeptidome

The set of all peptides presented by a cell via its MHC molecules is termed its immunopeptidome [33], and represents a unique fingerprint of the health of such cell. An immunopeptidome can be sampled by means of wet lab procedures that aim to identify peptides bound to MHC. Historically, two main techniques have been developed to do this: in vitro Binding Affinity (BA) assays and Eluted Ligands (EL) experiments, which quantify binding constants of peptide-MHC complexes and a cell's naturally presented peptides, respectively.

From a timeline perspective, BA assays were the first wetlab attempts to quantify MHC binding preferences [34, 35]. An example of these are binding competition experiments, in which the concentration of a query peptide that leads to 50% inhibition of a reference binding peptide (IC50) is measured [36]. If a low concentration of the target peptide is needed to displace the reference peptide, it means that the query peptide has great affinity for the MHC molecule under scrutiny, and vice versa. The result of such experiments is a set of quantitative binding affinities for different MHC-peptide combinations. However, a weak spot of such assays is that query peptides must be synthesized a priori, which makes the overall procedure costly, hard to scale and prone to selection bias. BA experiments also ignore in vivo characteristics of the MHC antigen presentation pathway, whose steps are related to selection and processing of relevant binding peptides.

As time passed, novel and more sophisticated approaches were developed in order to acquire peptide-MHC binding data. In particular, Mass Spectrometry (MS) proteomics has revolutionized biology with its ability to sequence and quantify proteins on the proteome scale [37]. In a really broad sense, the idea of such proteomic pipelines is to first isolate some protein sample one wants to characterize and then feed it to a mass spectrometer in order to obtain its aminoacidic sequence. In recent years, technology has enabled better isolation techniques for MHC-bound peptides for their posterior usage as input for mass spectrometry [38], setting in motion the field of MS immunopeptidomics and facilitating the high-throughput extraction of EL data from these kind of experiments [39].

As stated above, in the case of immunopeptidomics, the first step prior to applying any MS technique is to prepare a target biological sample consisting of peptides that were previously bound to MHC [40] (refer to Figure 1.5 for a general overview). This is done by first selecting a type of cell expressing the MHC variants of interest, usually by genotyping the MHC of such cell. Then, the sample is cultured and mixed with detergent in order to lyse its cells by rupturing their membranes. From this lysate, peptide-MHC complexes (previously bound to the lysed cellular membranes and also present inside such cells) become isolated via immunoprecipitation using antibodies specific to the MHC class of interest. Afterwards, peptide-MHC bounds are ruptured using acid (i.e. acetic), and peptides are further separated from other remaining acid-digested MHC components (-chains, -chains and 2-microglobulins) by means of liquid chromatography. At this stage, the sample is mostly composed of MHC binders, and thus is in place to be loaded onto a mass spectrometer for sequencing [39].

A mass spectrometer is a device used to measure the mass to charge (m/z) of ions. In a typical MS experiment, a solid, liquid or gaseous sample is ionized, for instance by bombarding it with a beam of electrons [41]. This may cause some of the sample's molecules to break up into positively charged fragments or simply become positively charged without fragmentation. These positively ionized products are then separated according to their m/z ratio, for instance, by accelerating them under a deflecting electric or magnetic field. Since ions of the same m/z composition are (according to the Lorentz force [42]) deflected equally, they will always hit the same spots on a collision sensor, enabling a reproducible detection and characterization.

In a similar way, mass spectrometry of proteins requires that proteins in solution or solid state be turned into an ionized gas form before they are accelerated, detected and characterized. One of the most used methods for protein ionization is electrospray ionization [43] (ESI), where small and highly charged droplets (ions) are created from nebulizing a solution containing proteins. This technique allows for these fragile molecules to be ionized without fragmentation, and sometimes even preserves non-covalent interactions (this is why ESI is

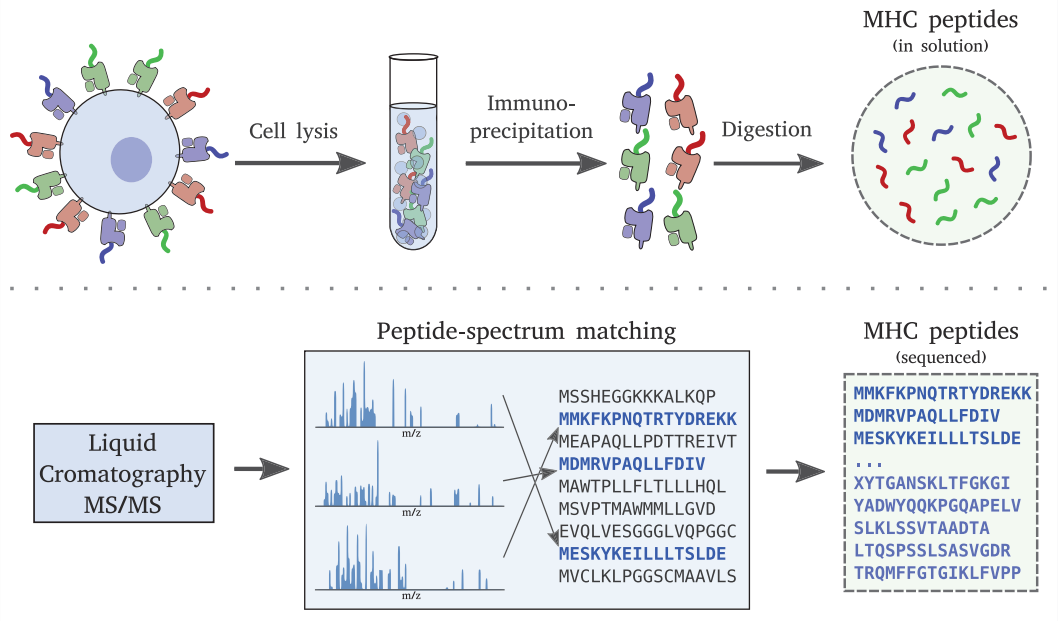

Figure 1.5. General overview of a Mass Spectrometry pipeline for Immunopeptidomics. In the top panel, the sample preparation step is shown. First, a target cell culture expressing MHCs is lysed. Then, immunoprecipitation is applied to separate peptide-MHC complexes from the rest of cellular debris. A digestion step is afterwards ensued to recover only bound peptides, and these are fed to a Liquid Chromatography-coupled tandem MS (bottom panel). After peptide-spectrum matching, the associated list of MHC peptides for the cell line under study is finally assembled.

often called soft ionization). Also, the requirement of having to work with a liquid protein solution makes it ideal to be readily coupled to the last step of a sample's preparation. To help with this, mass spectrometers can be integrated with high-performance liquid chromatography (HPLC) columns [44], allowing for the analysis of complex peptide mixtures by introducing peptides into the instrument at a controlled rate, ideally one at a time. So, after HPLC-assisted electrospray ionization, generated droplets evaporate under Coulomb fission [45] and resulting protonated peptides enter a first mass spectrometer, where an m/z spectrum is recorded (MS spectrum). Then, peptide-ions of a particular m/z coming from the first mass spectrometer are selected and fragmented into smaller ions (for instance, by means of energetic collision with a gas). These fragment-ions are then injected into a second mass spectrometer, which produces -thanks to the prior fragmentation- a more specific m/z spectrum (MS/MS spectrum). This technique of using multiple mass spectrometers connected in series is called tandem mass spectrometry, and is able to achieve specificities and sensitivities equivalent to other competing methods while performing analyses in much shorter times [46]. Finally, after the MS and MS/MS spectra are acquired, they become stored for further processing.

Since peptide-ions fragments create patterns characteristic of a specific amino acid sequence [47], the peak pattern of the MS and MS/MS spectra provides information about the peptide sequence. Considering this, sequencing can be directly done from the recorded spectra by means of "de novo" peptide-spectrum matching [48], or by implementing a database approach where measured spectra is compared against theoretical spectra of peptides we expect to find in our sample [47]. For this latter approach, bioinformatic algorithms score a query MS/MS spectrum against predicted fragmentation spectra of sequences from a target database, returning a list of high-scoring amino acid sequences. While these algorithms are very powerful, the problem is that there is substantial overlap between the scores for correct and incorrect peptide hits [49]. So, a priori, one does not know which reported matches are correct and which are wrong, and this limits the proper identification of true positives. To manage this far from ideal situation, a False Discovery Rate (FDR) filtering is usually introduced. First, a decoy database is constructed by reversing [50], shuffling [51] or constructing random peptides [52] from the target database. Then, for a given MS/MS spectrum, the reported matching scores distribution for the target database and the reported matching

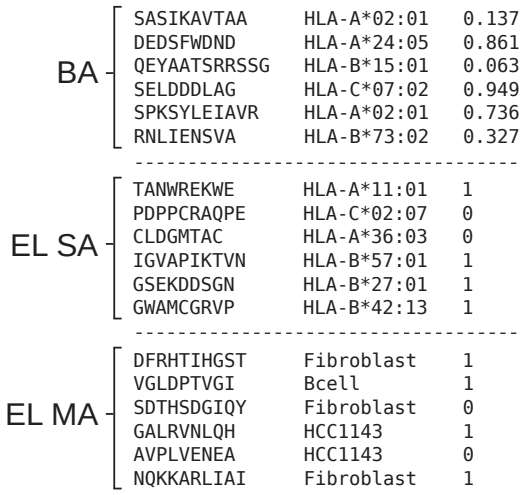

Figure 1.6. Comparison between three BA, EL SA and EL MA datasets. For all data types, the first column corresponds to experimentally acquired peptide sequences. The second column contains the restriction elements: for BA and EL SA, such restrictions are HLA molecules; on the other hand, for EL MA, specific identifiers are present. It is common for such identifiers to be named after the cultured cell line, whose expressed MHCs are known. Finally, the third column contains the target value for the corresponding peptide-restriction pairs. Notice how BA data have real-valued target values, whereas for EL SA and EL MA these are boolean-valued.

scores for the decoy database (which will serve as a background distribution or null hypothesis) are generated. Lastly, for a given score threshold, a simple FDR can be obtained by counting the number of decoy matches (DM) above the threshold and the number of target matches (TM) above the threshold and computing the ratio DM/TM. With this, one can play with the score thresholding in order to increase or decrease FDR to a value that suits the needs [49].

Independently of the approach used for peptide-spectrum matching, the final result of an immunopeptidomics MS assay is a list of sequenced peptides that bind to the MHC molecules expressed by the input cell culture. However, since the sample preparation step decouples peptides from their associated MHCs, such output list will have a mixed MHC specificity, and thus each peptide-MHC mapping will be multiple. The mixture complexity will depend on the cell's genotype and the target MHC class (in the context of human immune system, cells are multi-allelic, and up to 6 MHCs can be expressed for class I and up to 12 MHCs for class II). We term this type of EL data Multi-Allele (MA) data. On the side, genetically modified cell lines expressing a single MHC molecule (mono-allelic) circumvent this problem [53]. Since such cell lines express only one MHC type, they generate what we call EL data of the Single-Allele (SA) type, which has a much more simplified analysis, but such data usually makes up only the minority of MS experimental setups. Finally, a crucial characteristic of EL data (both SA and MA) in comparison to BA data is that it is not affected by selection bias imposed by hypothesis-driven peptide selection, since by definition every EL peptide is a natural MHC ligand. For a comparison between all these data types, refer to Figure 1.6.

Sampling the immunopeptidome of a target cell is, for sure, a highly technical and skilled endeavour. Moreover, the gigantic amount of data it generates is highly enriched in valuable immunological information. Since this type of data essentially captures MHC binding preferences towards specific peptide subsets, a question arises: could it be possible to transfer such preferences to a computational model? This is, in essence, to construct an algorithm capable of learning the rules that govern peptide-MHC interactions and render them into some human-interpretable form. If so, such algorithm could be exploited to predict the binding of new peptides towards a target MHC, and thus to infer a potential immune response. Different approaches can be used to push forward the development of such a model, but given the data-driven nature of our problem, employing Machine Learning is a great starting point.

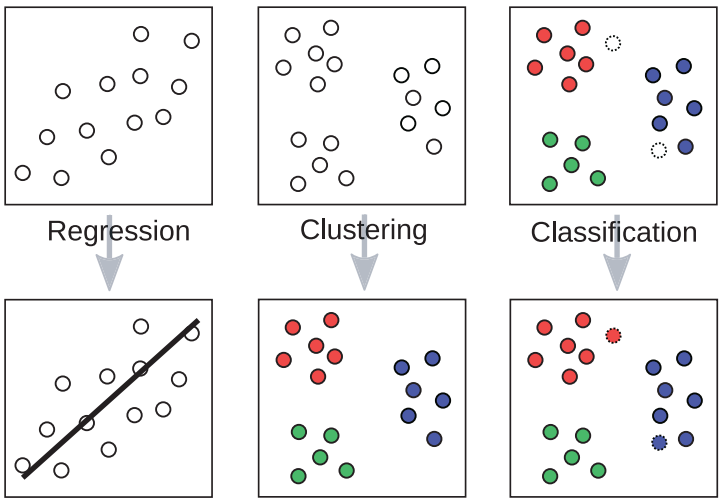

Figure 1.7. Toy illustration of regression, clustering and classification tasks. For regression, a set of points can be used to construct a function approximating such a set, in order to make new inferences (in this example, a linear model is shown). In clustering, one can start from unlabeled information and exploit different properties of the data space (i.e. distance metrics) to construct groups of similar points. Finally, classification algorithms enable labeling of new data points (dotted circumferences) according to some learnt classification function (i.e. closeness criteria). Adapted from [65].

1.4 Machine Learning

Machine Learning is a subfield of Artificial Intelligence (AI) devoted to the research, development and application of computer algorithms to find meaningful patterns in data. While AI approaches may include operator-hardcoded rules, one of the main characteristics of ML methods is that they improve automatically by means of "looking" at data, in a process called learning [54]. This leads to an algorithm capable of discovering how to make predictions or decisions without being explicitly programmed to do so.

The first published usage of the term Machine Learning dates from 1959 by Arthur Samuel, a North American electrical engineer working at IBM. In his paper "Some Studies in Machine Learning Using the Game of Checkers" [55], Samuel proposes an algorithm capable of accumulating experience by playing successive games of checkers against a human opponent. Such algorithm is based on the construction of directed graphs, where nodes encode checkerboard states and edges represent valid moves. A backwards graph traversing strategy is used to decide the optimal play at a given game state, with this optimum depending on certain weights that are updated move after move. Finally, and after 8-10 hours of playing, Samuel's algorithm outplayed him [55].

Besides the morbid fun of dominating its creators, Machine Learning algorithms serve a whole list of meaningful purposes. Common tasks include: data regression, data classification and data clustering (Figure 1.7). Generally speaking, the solution to a regression problem is a function able to map a given input space to a real-valued (or continuous) output space; examples of regression algorithms are: linear regression [56], polynomial regression [57] and logistic regression [58]. In a similar fashion, solutions to classification problems are able to map input spaces to integer-valued (or categorical) output spaces; some examples of classification algorithms are: perceptron [59], linear discriminant analysis [60] and k-nearest neighbours [61]. Finally, clustering solutions are able to separate the input space in regions such that elements in a given region (or cluster) are more similar (according to some metric) to each other than those in other clusters; examples of clustering algorithms are: k-means clustering [62], self-organizing maps [63] and DBSCAN [64]. It is important to mention that the final "shape" of a mapping or clustering will depend on the applied algorithm.

The above mentioned ML problems are usually solved under some type of learning

paradigm. Up to date, the most developed ones are: supervised learning, unsupervised learning, semi-supervised learning and reinforcement learning. Supervised learning refers to the task of learning a function that maps an input to an output based on known input-output pairs [66]; in other words, It makes possible to infer such function from labeled training examples [67]. On the other hand, unsupervised learning enables learning patterns from unlabeled training examples through some type of self-organization [68]. The third category, semi-supervised learning, refers to the middle point between the first two, in the sense it leverages using both labeled data and unlabeled data during training [69]. Regarding reinforcement learning, it is the most youthful of the four paradigms, and has gained a lot of attention in recent years mostly thanks to groundbreaking work [70–74] done by OpenAI [75], DeepMind [76], and others. Essentially, it focus on the idea of learning by interaction, and deals with how AI agents map situations to actions in a given environment in order to maximize some type of reward function [77] (refer to this video [78] for a quick and amazing demonstration of the capabilities of this paradigm).

If we now think about peptide-MHC interactions in ML terms, the task of predicting if a given peptide binds to a target MHC can be a regression problem (if BA data is employed), a binary classification problem (if EL data is employed), or hybrid (if using mixtures of BA and EL data). Furthermore, given these types of immunopeptidomics datasets, the learning process can be supervised (if we use BA and/or EL SA data), unsupervised (if we only use EL MA data), or semi-supervised (if we mix BA and/or EL SA with EL MA data). Luckily, all these requirements can be jointly addressed if Artificial Neural Networks are employed as the training algorithm for peptide-MHC binding predictors.

1.5 Artificial Neural Networks

Introduction

Artificial Neural Networks are a ML algorithm loosely inspired by the wiring of biological neural networks [59, 79]. In essence, ANNs are weighted directed graphs composed of nodes/neurons and vertices that perform computations and propagate such computations, respectively, throughout the graph topology. A particular way of grouping and connecting nodes together is referred to as network architecture. Several architectures have been developed in the past [80], ranging in complexity and applications. One of the most simple and widely used network configurations is called Feed Forward Neural Network (FFNN).

FFNNs are acyclic directed weighted graphs whose neurons are grouped in layers connected one after the other in a sequential fashion (refer to Figure 1.8 to see an example architecture). Generally speaking, in FFNNs each neuron inside a layer is connected to all the neurons inside the previous layer, and receives their outputs as input. Then, they compute a weighted sum of such inputs from the previous layer and apply a nonlinear activation function to it. Activation functions enable ANNs to solve complex, nontrivial mapping problems by means of bending their inner representations [81] (moreover, it has been proven that a multi-layer FFNN with non polynomial activation functions can indeed approximate any function [82]). Such activation functions come in different shapes and flavours, ranging from a step function, sigmoid or ReLU, between others [83].

Specifically, for the graph shown in Figure 1.8, each node/neuron is connected to all downstream neurons through weighted vertices. Neurons are grouped together into layers; specifically, from left to right, an input layer, hidden layer and output layer. Two particular units are shown in the bottom of the illustration, called bias neurons. To spread information through the topology, an input vector $\vec{x}$ of length M and componentes x_i is fed to the input layer. This layer propagates such vector to the hidden layer, where a neuron h with associated weights $w_{i,h}$ will compute the operation:

$$f(\Sigma) = f\left(\sum_{i=0}^{M} w_{i,h} \cdot x_i + b_h\right) \tag{1.1}$$

where f represents an activation function. The output of such operation (for each hidden neuron) is then passed onto the last layer. Here, the only available neuron o will repeat the computation of Equation 1.1, but using the previous layer's output as input, its weights

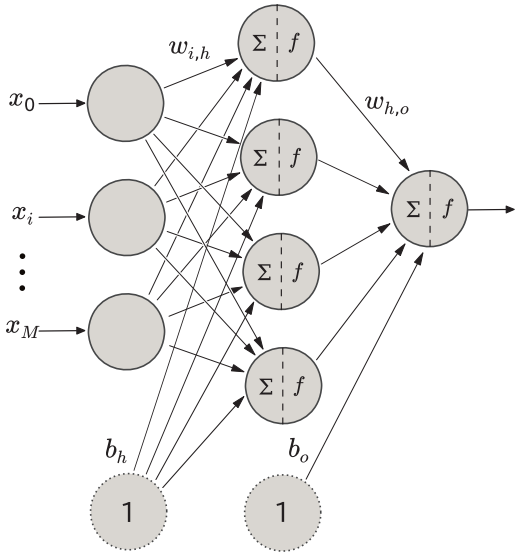

Figure 1.8. Example architecture of a FFNN.

$w_{h,o}$ and bias b_o. This last operation finally generates the output of the network for input $\vec{x}$. As can be observed, ANNs with multiple neurons can become computationally taxing pretty easily. Historically, this has always been a problem, and narrowed the applicability scope of neural networks. However, in recent years, the acceleration of hardware capabilities has overcome this, and led to the blooming of more complex ANNs capable of dealing with higher order problems (thanks to denser graphs), popularly referred to as Deep Neural Networks (DNN).

Deep Neural Networks are the object of study of Deep Learning (DL), a subfield of Machine Learning focused on the research and development of multi-layer ANN architectures capable of learning representations of data with multiple levels of abstraction [84]. As an example, a FFNN with two or more hidden layers can be considered a deep network. If we, for instance, train such network for object detection in images, the learned features in the first layer will typically represent the presence or absence of edges of particular angles, a second layer will mostly detect arrangements of edges, a third layer may group such arrangements into larger combinations that will correspond to parts of familiar objects, and subsequent layers will detect objects as further combinations of these previous representations [85]. This compounding abstraction leads to strange, yet very interesting observations such as the presence of specific floppy and pointy ear detectors in dog vs. cat image classification models [86], and more recently the existence of in-silico multimodal neurons [87], first described in neuroscience experiments by Quiroga et al. [88], which are capable of selectively firing to specific individuals, landmarks or objects.

The abstraction power offered by deep learning has boosted the problem solving capabilities of several fields. In recent years, DL approaches have beaten image recognition [89–92], speech recognition [93–95], and protein folding [96, 97] prediction records, and have been successfully applied at predicting the activity of potential drug molecules [98], analysing particle accelerator data [99, 100], reconstructing mice brain circuits [101], segmenting cell membranes in electron microscopy images [102], detecting mitosis in breast cancer histology images [103], predicting protein contact maps [104], playing atari, go, chess [105] and Dota 2 [72], between others.

Many of the aforementioned milestones have been achieved using a particular DNN architecture called Convolutional Neural Networks (CNN), which are a specialized kind of feed-forward networks. One of the fundamental differences with FFNNs is that they can learn local patterns through the use of convolutional filters over the input data [106]; also, CNNs are able to operate upon variable-length inputs, while FFNNs are not. The convolution is a mathematical operation broadly used in signal processing, and is capable of transforming certain target elements of an input signal and producing a desired filtered out-

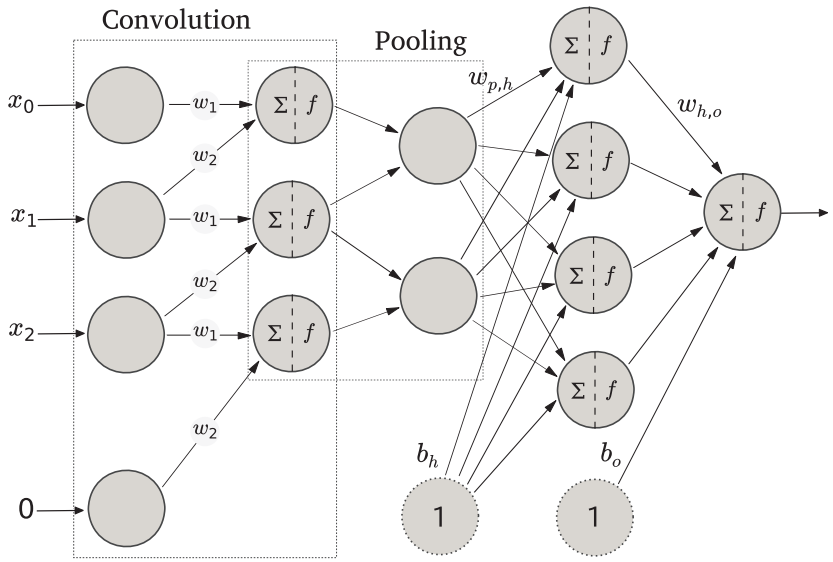

Figure 1.9. Example schematic of a CNN architecture. For simplicity, a short input $\vec{x}$ of only three components is fed to the input layer, which propagates the information to the convolutional layer. In this example, a single two-weight convolution $[w_1, w_2]$ is applied to a zero-padded input (this preserves dimensionality). Neurons in the convolutional layer apply the transformation shown in Equation 1.1, and pass the output to pooling neurons p. These units are in charge of performing some downsampling operation, such as selecting the maximum value of their inputs. As a result, the length of the convolutional output is reduced by one. Next, the output of the pooling layer is fed to a FFNN through the weights $w_{p,h}$ to produce the final network output. In practice, CNNs tend to implement several convolution layers with variable quantity of weights and different types of convolutions. Note: the biases of the convolution neurons are omitted from the drawing in order to simplify it.

put [107]. In the context of CNNs, convolutions are applied by means of adaptive filters of different sizes and strides, whose weights are adjusted through the training cycle of the network. Upon convolving their input, convolutional layers transfer their output to the next layer, which is usually a pooling layer that summarizes its input by computing, for example, its maximum or average [108]. Afterwards, a concatenation of FFNN layers tends to be used to produce the final output (refer to Figure 1.9 for an illustrative example). Two main advantages of CNNs are that learned patterns are translation-invariant (this is, independent of the position they occupy in the input sample) and, as with any other DL technique, a sequential hierarchy of abstraction between such patterns can be established. Although CNNs were originally designed to operate upon image and video inputs [89, 92, 109–112], they have also been applied to biology-related data [113–119], proving their potential use for sequence-based, biological pattern recognition challenges such as the one proposed in this thesis.

Independently of the shallow- or deepness of the ANN architecture of choice, all neural networks are composed of interconnected weights that resemble a graph topology. Such weights represent the trainable parameters of these networks, and indicate how much influence each neuron's output will have on downstream neurons. This means that for an ANN to learn a correct input-output mapping representation, such weights need to be adjusted to a specific value range. This is generally referred to as training or fitting the network, and is done by means of applying some kind of optimization strategy.

Optimization Approaches

As mentioned above, a neural network must be trained in order to learn meaningful data representations. In this context, training consists of making small, consecutive changes to the network weights until the model correctly maps its inputs to its corresponding outputs. Such changes do not happen randomly, but following an optimization criteria that quantifies the mapping quality using some distance metric between the predicted and measured outputs.

A valid approach is to implement a loss function L to measure the predictive error of the model being trained; the overall shape of L will depend (among other things) on the type of problem trying to be solved. For instance, in a supervised regression task, a common loss metric is the Mean Squared Error (MSE) function:

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(\theta, x_i))^2 \tag{1.2}$$

where N is the total samples in the training set, y_i is the target (or measured) value of sample x_i, f represents the input-output mapping (in this case, an ANN) and θ are the trainable parameters of such mapping. Ideally, the goal is to find θ_M, or the specific subset of θ that minimizes L globally.

Because of the large quantity of parameters in ANNs models, calculating such minimization by means of an analytical (closed-form) solution, and in a sensible time, is just not possible. For instance, the weight count of LeNet-5 [120] (a neural network initially built for handwritten digit recognition) is 431.000 [121]; and this can be considered small, since for other networks such as AlexNet [89] it is in the order of hundreds of millions [121]. This mesmerizing amount of trainable parameters means that a different approach needs to be used in order to minimize L. If we think about it, choosing differentiable activation functions (such as sigmoids) make f differentiable over θ, and thus L becomes differentiable over θ as well. This satisfies the necessary and sufficient condition for ∇L to always point towards the direction of maximum growth of L from any given θ [122]. Now, if following ∇L from a given domain point would mean to "walk" towards a maximum error, walking towards $-\nabla L$ would mean to walk away from this maximum error direction or, in other words, walk towards the minimum error direction. This clever calculus strategy is exploited by the algorithm of Gradient Descent (GD) [123], which has the following form:

$$\theta^{t+1} = \theta^t - \eta \cdot \nabla L(\theta^t) \tag{1.3}$$

where θ^t represents the current state of the model (the "present values" of θ) and η is a scalar known as learning rate, which is used to modulate the contribution of ∇L to the weight updating schema. Several flavours of GD are used in Machine Learning [124], all equally in charge of exploring the hills and cliffs of a given loss function in order to minimize it (Figure 1.10). It is important to notice that, on an arbitrary neural network, L has no guarantees of behaving as a convex function of θ (weights can be positive or negative, there are nonlinearities -activation functions-, etc.). Because of this, GD will most likely not converge to the global minimum θ_M of the error landscape, but a local minimum θ_m instead. These local minima are however usually sufficient to generate overall good predictors in many practical problems [84]; moreover, avoiding θ_M may be beneficial, since it prevents over-training a neural network.

So, in order to compute θ^{t+1} in Equation 1.3, the gradient of Equation 1.2 at a given iteration t needs to be calculated, such that:

$$\nabla L(\theta^t) = -\frac{2}{N}\sum_{i=1}^{N}(y_i - f(\theta^t, x_i)) \cdot \nabla f(\theta^t, x_i) \tag{1.4}$$

The hard task now is to calculate ∇f, since the complexity of f increases with the quantity of network layers (from a calculus perspective, f represents an extensive series of function compositions), and again this cannot be done analytically in a sensible time. To tackle this issue, the Backpropagation (BP) algorithm was introduced in the early beginnings of neural networks development [125,126]. BP works by computing the derivatives of f with respect to each network weight using the chain rule, calculating intermediate gradientes in the function composition one layer at a time, starting from the output layer and finishing in the input layer. Backpropagation is a particular case of Automatic Differentiation (AD) [127], a set of techniques to evaluate the derivative of any function specified by a computer program. AD exploits the fact that computer programs can be represented as a graph whose nodes represent elementary arithmetic operations (addition, subtraction, multiplication, division, etc.) and elementary functions (exp, log, sin, cos, etc.). Then, by applying the chain rule

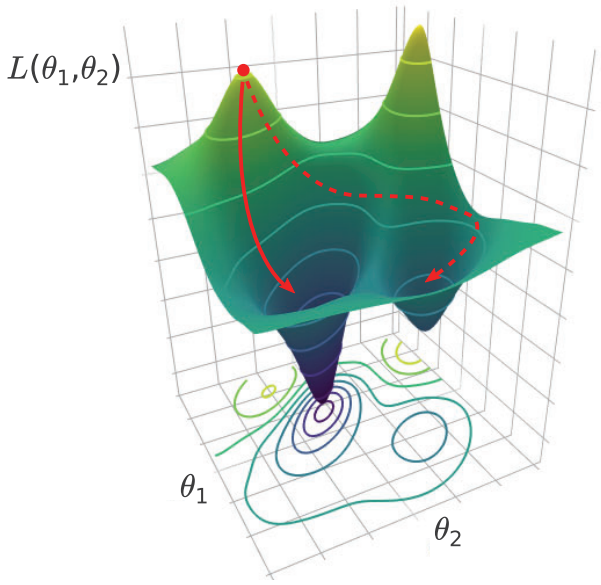

Figure 1.10. Visualization of the gradient descent algorithm. Here, a segment of the error landscape of a theoretical non-convex loss function L, depending on model parameters (θ_1, θ_2), is shown. The level curves of L are projected on the parameters plane. Starting at the red dot, and depending on the heuristics of execution, GD may converge to the leftmost well (solid red line), or the rightmost well (dashed red line). Red dot's final resting place will determine the local minimum of L, and therefore the set of parameter values to be used.

repeatedly, derivatives of arbitrary order can be computed from the graph by calculating node gradients. At the end of the day, BP makes the task of obtaining ∇f feasible, which in turn enables heuristic minimization by GD and makes it possible to train a neural network.

Training

In general, neural networks are trained on data extracted from a training set in a pretty straightforward way: a training instance is fed to the network, loss is calculated, graph gradients are computed, BP is applied and weights become updated. This procedure is repeated until all input data becomes used, or, in other words, a training epoch is concluded. Then, the cycle starts all over again, and usually ends when all the desired epochs finish executing. After this, the model can be challenged to make predictions on some previously unseen data to measure its ability to generalize, or to maximize its predictive performance on such data. So, during the training phase, it is desirable to estimate -in some way- how the generalization process is unfolding. A well documented way of doing this is to apply a K-fold Cross-Validation (CV) schema. In general, CV is a model validation and selection technique used for assessing how the results of training will generalize to a previously unseen, independent data set [128]. To do this, all N available training points are split into K partitions of N_k elements each. In general, N_k can be chosen at will for each partition, but usually $N_k = N/K$ is used. Then, the learning algorithm (in our case an ANN) is fitted on a training set composed of $K-1$ partitions and its performance is evaluated on a validation set consisting of the left out partition. Afterwards, this process is repeated K times, cycling partitions until all partitions are used as validation set [129]. The process results in K trained models, that can be used to estimate the power of the implemented ANN architecture to predict unseen data and/or for model selection (i.e. by concatenating the K validation folds predictions and calculating associated performances).

As mentioned above, one of the critical goals of cross-validation is to estimate the model's ability to generalize. If the network fails to do so, there is a risk of tightly fitting the training data by learning small, specific and/or noisy statistical variations instead of a generalized predictive rule [130]. This phenomenon is usually called overfitting and is really important to keep it under control when training ANNs, since an overfitted network will lose any ability to work on new, real world data, becoming rather useless. One of the easiest ways to

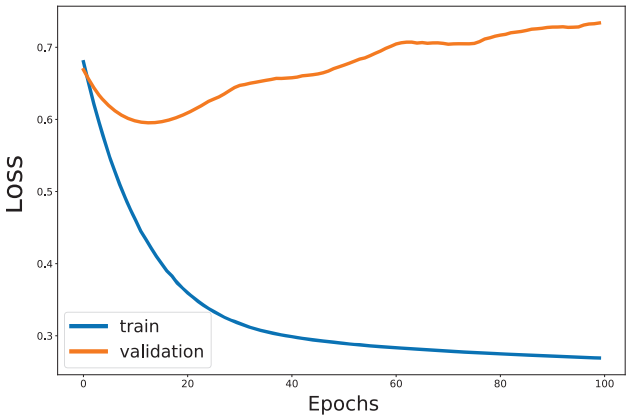

Figure 1.11. Example of an overfitting measurement in machine learning. Here, the train set loss (blue) decreases monotonically, but starting at epoch 20 (approximately) the validation loss (orange) starts to increase. This particular example is the result of training a FFNN with 500 hidden layer neurons on a toy dataset composed of only 100 points of 2 features each, using a $K = 2$ cross validation. As a result, it depicts how an over-complex network (for the dataset at hand) is able to fit information to an extreme -and detrimental- detail.

address overfitting is to monitor the model training during cross validation [131]. To do this, some prediction metric is extracted, epoch after epoch, for both the training set and the validation set. Such a metric can be, for instance, the value of the loss function L (Equation 1.2). As epochs complete (and assuming our ANN architecture is sound) one would expect for both training and validation losses to go down and then into a plateau. However, this is usually not the case. As shown in Figure 1.11, beginning at some epoch, the validation loss may start to go up, whereas the train loss may continue to decrease. This is a classic, telltale sign of a model starting to lose its generalization capabilities, and applying some kind of counter-strategy is key to avoiding it.

Several techniques exist to deal with overfitting, known collectively as regularization. In the above scenario, an initial counter-strategy would be to apply early stopping, which consists of just stopping the training when the validation metric gets continuously worse after a predefined quantity of epochs (something commonly denoted as patience) [132]. Afterwards, the model weights for the best epoch are generally saved as the output of the training. There are several other regularization recipes available, such as L1 and L2 regularizations [133], elasticnet regularization [134], data augmentation [135] and dropout layers [136], each one with its strengths and weaknesses. As an example, dropout layers randomly set a fraction of neurons in a given layer to zero during training, preventing units from co-adapting too much to the data, and thus increasing overall generalization [137]. The fraction of neurons dropped is commonly known as dropout rate, and represents what is called a network hyperparameter (HP).

HPs are a "special" kind of parameters, in the sense they do not get updated by the BP algorithm during training iterations [138]. Activation functions, quantity of neurons and layers, early stopping's triggering epoch, and even the learning rate of eq. 2 are all forms of hyperparameters. For a neural network to perform optimally, its hyperparameters need to be fitted too, in a process called hyperparameter tuning or optimization. Pragmatically, this means to select some target subset of HP configurations, extract their CV performances and select the best performing configuration as the winner. In order to do this, different strategies have been proposed to explore the usually gigantic hyperparameter space, from painful manual tweaking to smarter algorithms such as grid search [139], random search [140], and bayesian optimization [141], between others [142].

Independently of the chosen HP optimization criteria, a new concern surfaces in the context of model validation. From what was shown above, K-folds Cross-validation is performed to validate and select statistically sound models; however, optimal HPs are also selected during cross validation (i.e. early stopping's epoch). This would mean to use the validation set for both fitting hyperparameters and reporting performance, resulting in a biased estimation

(usually an overestimation) of the model's true predictive power [143]. Because of this, an "extended" version of CV is often employed when training neural networks, called Nested Cross-validation. Different ways of approaching such nested CV exist, but commonly a first chunk of N_t data points (the test set) is separated from the N original instances, and then the remaining $N - N_t$ points are partitioned following the classic CV schema. In this scenario, the test set serves as a true independent evaluation for model selection, since it is not used for fitting parameters nor hyperparameters.

Finally, independently of using vanilla CV or nested CV for generalization estimation and/or model selection, a pristine left out dataset (never used during training whatsoever) should be utilized to assess the true predictive power of our model, and to compare it against competitors. In the case of peptide-MHC binding predictors, a commnly used type of left-out data are epitopes (MHC ligands that elicit immune responses), since they share the property of being binders (category used for training), but also represent real true positives from the immune system's perspective.

From all the exposed above, it can be observed that rigorous practices are crucial for correctly training, validating and testing neural networks. From a ML pipeline perspective, this is tightly bound to conducting proper measurements during the different phases of such a pipeline (measuring train and validation loss to detect overfitting, assessing independent predictive power for model selection, etc). In order to do this, different performance metrics are used.

Performance Metrics

To quantitatively guide the development of neural networks, several performance metrics are employed. Either to assess a model's performance or compare it with competing algorithms, the chosen metrics will depend on the type of problem being addressed. For instance, regression models rely on some distance criteria between predicted values $\hat{y}$ and measured values y, like the root mean squared error (RMSE) [144]:

$$\mathrm{RMSE}(\hat{y}, y) = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(\hat{y}_i - y_i)^2} \tag{1.5}$$

with M being the quantity of samples in the dataset that is going to be predicted. Also, correlation measurements are broadly used. Assuming that $\hat{y}$ and y have a linear relationship, one can use the Pearson Correlation Coefficient (PCC) [145]:

$$\mathrm{PCC}(\hat{y}, y) = \frac{\sum_{i=1}^{M}(y_i - \bar{y})(\hat{y}_i - \bar{\hat{y}})}{\sqrt{\sum_{i=1}^{M}(y_i - \bar{y})^2}\sqrt{\sum_{i=1}^{M}(\hat{y}_i - \bar{\hat{y}})^2}} \tag{1.6}$$

where variables with overlines represent the mean values of such variables. PCC lies in the [-1,1] range, with -1 indicating perfect linear anticorrelation, 0 no linear correlation and 1 perfect linear correlation. This means the higher the PCC, the higher the correspondence between predictions and measurements. On the other hand, quantification of non-linear correlations between $\hat{y}$ and y can be achieved using the Spearman's Correlation Coefficient (SCC) [146], which is computed similarly to PCC but assuming a monotonic relationship between predictions and measurements.

On the other hand, classifiers use metrics related to their ability to predict the proper measured classes. In particular, binary classifiers deal with the task of predicting two possible output classes, usually referred to as the positive class P and the negative class N. This creates four possible outcomes: instances belonging to P can be correctly classified as belonging to P (true positives, or TP), or wrongly classified as belonging to N (false negatives, or FN); on the other hand, elements of N can be correctly classified as belonging to N (true negatives, or TN), or wrongly classified as belonging to P (false positives, or FP). These four categories are often used to compute the Confusion Matrix (see Figure 1.12) of the classification task, which is a quick intuitive way of visualizing the classifier's performance.

		Predicted Class	
		Positive	Negative
Measured Class	Positive	True Positive **TP**	False Negative **FN**
	Negative	False Positive **FP**	True Negative **TN**

Figure 1.12. Confusion matrix showing the four categories of binary classification. Rows correspond to measurements, columns correspond to predictions.

Consequently, the entries of the confusion matrix can be used to derive many useful performance criterions. In principle, one may want to measure the classifier's ability to correctly and wrongly assign a class label to its inputs. If we start with elements of P, we could use the True Positive Rate (TPR) and False Negative Rate (FNR):

$$\text{TPR} = \frac{TP}{TP + FN} \tag{1.7}$$

$$\text{FNR} = \frac{FN}{TP + FN} \tag{1.8}$$

TPR (also known as Sensitivity or Recall) measures the amount of correctly identified positive instances over all the positive population; on the other hand, FNR denotes the ratio of wrongly identified positive instances over the positive population. In a similar fashion, related metrics can be employed for elements of N, such as the True Negative Rate (TNR) and False Positive Rate (FPR):

$$\text{TNR} = \frac{TN}{TN + FP} \tag{1.9}$$

$$\text{FPR} = \frac{FP}{TN + FP} \tag{1.10}$$

TNR (also referred to as Specificity or Selectivity) is the ratio of all the correctly identified negative instances over the negative population; on the side, FPR (also referred to as Fallout) computes the ratio of wrongly identified negative instances over the negative population. Since TPR + FNR = 1 and TNR + FPR = 1, it follows that TPR + FNR = TNR + FPR, showing how binary classification is a literal interplay between being correct and being wrong for both P and N. Other commonly used performance metrics are Accuracy (ACC) and Positive Predictive Value (PPV):

$$\text{ACC} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1.11}$$

$$\text{PPV} = \frac{TP}{TP + FP} \tag{1.12}$$

ACC computes the total number of correct classifications over the whole $P + N$ population, while PPV (which is also known as precision) evaluates the amount of correct positive calls over the total positive calls (both correct and wrong). In particular, when dealing with models trained to predict peptide-MHC binding, the F-Rank (abbreviated FRANK) becomes a useful metric to quantify independent test set performances:

$$\text{FRANK} = \frac{FP}{N} \tag{1.13}$$

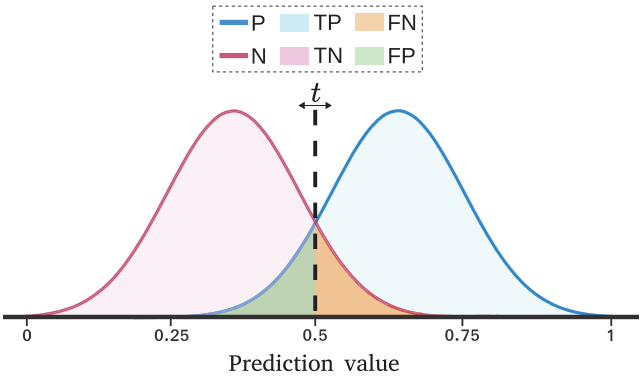

Figure 1.13. Visualization of TP, FN, TN and FP as a function of prediction threshold t. Curves correspond to the probability density functions of classes (P and N) as a function of prediction value. Prediction values above t are classified as positives, while values below as negatives (for this example, $t = 0.5$). Overlapping areas (in green and orange) correspond to misclassifications.

When evaluating the capacity of an MHC binding predictor, true measured epitopes are conventionally employed. A pretty much standard testing set will consist of a unique positive instance (the epitope) and multiple negative instances. These negative instances are obtained from the protein where such epitope was found, by means of extracting all the possible sub-sequences of a certain length. With this, FRANK tells us the model's ability of identifying a singleton hit, in a way "emulating" how an MHC molecule would scan an antigenic protein, and providing a useful measure of peptide prioritisation in the context of epitope discovery. This metric outputs a score of 0 for a perfect predictor (epitope was ranked first in the ordered prediction output) and a score of 0.5 for a random predictor.

Besides classification models that directly output class assignment probabilities (such as naive Bayes [147]), certain regression models can be transformed into classification algorithms. In particular, an ANN with a sigmoid activation function in its output layer is an example of such a model. Basically, since output values fall in the range [0,1], they can be interpreted as estimations of conditional probabilities over the input feature space. For binary classifiers, after getting a prediction value p, one can use a threshold t to assign a class to a prediction, such that if $p > t$ it belongs to P, or to N otherwise (see Figure 1.13). A question then arises: what is a proper value of t? The short answer is that $t = 0.5$ is usually employed, but nonetheless one could want to pick t such that it optimizes the classifier's operation, according some criteria.

Moreover, implementing a metric that shows the overall performance of a classifier for all possible threshold values (and not only the chosen one) is also pretty desirable. To do so, the Receiver Operating Characteristic (ROC) curve is usually deployed. The ROC space was originally introduced during World War II in order to calibrate the operating point of radars (receivers); concluded the war, one of the first formal works on the topic was published by Peterson et al. in the early 50s [148]. The ROC curve is defined as the plot of TPR as a function of FPR parameterized for $t \in [0, 1]$, and depicts how the probability of detection is affected by the probability of false alarm for decreasing decision thresholds. The curve always starts at (0,0) ($t = 1$) and finishes in (1,1) ($t = 0$), with the best shape being a step function (ideal classifier), and the worst shape being an unitary slope rect (meaning that using the classifier is the same as random guessing). Analogously, the Precision Recall Curve (PRC) [149] is another useful way of measuring the predictive performance of a classification model. It is constructed by plotting PPV as a function of TPR also for $t \in [0, 1]$, and illustrates the algorithm's capacity of correctly retrieving positive instances from both the true positive and predicted positive populations, for decreasing values of t. Such curve always starts at (0,1) ($t = 1$) and finishes at $(1, P/(P + N))$ ($t = 0$), with the best shape being an x-mirrored step function and worst shape a horizontal line located at $PPV = P/(P + N)$ (Figure 1.14). Finally, to crunch the ROC and PRC information into a single meaningful number, an Area Under the Curve (AUC) [150] is usually computed

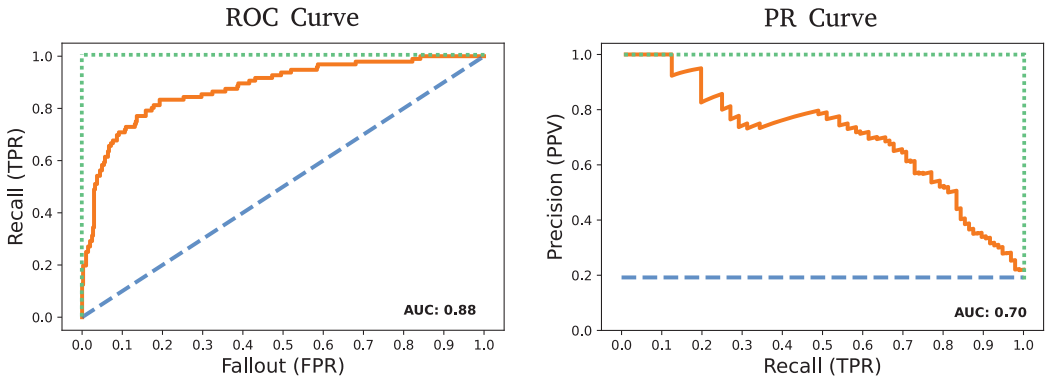

Figure 1.14. ROC and PR curves. Left panel: ROC curve (orange), random guess boundary (dashed blue line) and best possible classifier (dotted green lines); the corresponding AUC is displayed in the bottom-right corner. Right panel: PR curve (orange), random guess boundary (dashed blue line) and best possible classifier (dotted green lines); the AUC for this curve is shown in the bottom-right corner as well.

for both curves by means of integrating them. For ROC, an AUC of 0.5 means random guessing, while for PRC a random classifier corresponds to an AUC of $P/(P+N)$ (Figure 1.14). In some cases where certain FPR regions are not relevant and where the focus is centered on a high specificity rather than sensitivity, the usage of a partial AUC may be desirable. In this context, AUC0.1, defined as the integral of the ROC curve up to FPR = 0.1, has proven highly useful for comparing performance of epitope discovery pipelines

Besides all the aforementioned metrics being useful to assess an individual model's performance on independent data, statistics must be applied to prove that such model consistently outperforms its competitors. For binary classification problems such as peptide-MHC prediction, the binomial test is a good statistical test candidate. If, for example, we predict tests sets using models M_1 and M_2, we end up with vectors of associated metrics $\vec{m}_1$ and $\vec{m}_2$, each one of length T. Then, the null hypothesis H_0 will be that both models have the same probability (0.5) of outperforming the other for each set, and by comparing $\vec{m}_1$ and $\vec{m}_2$ element-wise we can accept or reject such H_0. This strategy will be thoroughly used in the publications presented in this thesis.

Up to this point, we have firstly introduced the primordial necessity of peptide-MHC binding to occur for the onset of immune responses. Then, a more in-depth examination of MHC's polymorphic nature revealed highly variant binding grooves, which in turn result in a vast universe of binding motifs. Moving on, advances in current technologies have enabled a heretofore unthinkable high-throughput sampling of immunopeptidomes, increasing the overall quantity and quality of published binding data. Finally, the ever-growing availability of such data has generated an increased necessity of interpreting and exploiting it, and for this cause machine learning (in particular, artificial neural networks) can be readily deployed. Finally, this complex connective tissue between immunology and algorithms represents the leitmotif of this thesis, and it can be fully interpreted as a direct application of techniques derived from the Immunological Bioinformatics domain.

1.6 Immunological Bioinformatics

Immunological Bioinformatics is a research field that applies computational techniques to generate a systems-level view of the immune system. This may be achieved in a stepwise fashion, where models are developed for the different components of the immune system, and then combined in order to understand and develop therapies, vaccines, and diagnostic tools for different diseases such as AIDS, malaria, and cancer. The long term goal of this field is to establish an in silico immune system, with the ultimate capacity of predicting B- and T-cell epitopes [151]. Then, and since the assembly of peptide-MHC complexes is a necessary condition for T-cell activation, capturing the binding preferences of MHC molecules is an obligatory step towards the ultimate goal of predicting T-cell epitopes.

Essentially, capturing the receptor preferences of an MHC molecule can be done by exploiting a set of peptide sequences that binds it. Naturally, the generated binding signals will have a high correlation with the binding groove of the scrutinized MHC, and thus will exhibit specific anchor positions and characteristic binding lengths. As shown earlier, MHC-I binders tend to have binding cores of 8-10 amino acids long and their anchor positions are often located at P2, P5/6 and PΩ; on the other hand, MHC-II usually binds peptides of length 13-25, and anchor positions tend to be located at P1, P4, P6, and P9. A first approach to quantify the enrichment of an amino acid at a position is to construct a Position Specific Scoring Matrix (PSSM) [152] from binding peptide alignments, using the following formula:

$$M_{a,i} = \log_2 \left(\frac{f_{a,i}}{q_a} \right) \tag{1.14}$$

where $f_{a,i}$ is the frequency of amino acid a at position i, and q_a represents the frequency at which amino acid a appears in nature (background frequency). Following Shannon's information axioms [153], a base 2 logarithm is then applied to the frequencies quotient in order to extract the information content of amino acid a at position i. PSSMs are the essence of Sequence Logos, but can also be used as MHC binding likelihood estimators by position-wise scoring the amino acids of a query peptide using the entries of $M_{a,i}$. Approaches like this are simple and require little amounts of data, but on the side they rely on linear assumptions such that peptide binding can be represented as the sum of individual amino acid contributions. Such assumptions narrow the applicability scope of these methods as predictors, mainly because they are unable to capture higher order correlations that may be present in binding data. Moreover, for MHC-binding, it has been shown that signals of higher order exist in amino acids located between the anchor positions [154]. Because of this, neural networks have become a major step forward in seizing more detailed peptide-MHC binding rules.

In order to train neural networks using sequence data, some type of encoding needs to be implemented. This means to transform peptides from the string space to a numerical representation, so the network is able to use them for loss computation and backpropagation. A simple approach is to use one-hot encoding, where each amino acid is represented as a vector of the alphabet length (20 in the case of proteins) that has a 1 in the position corresponding to the amino acid's alphabet position, and 0 elsewhere. It follows that for one-hot encoding every vector is orthogonal to each other, meaning that all amino acids are also assumed to be orthogonal. This, of course, is not what is observed in nature, where different amino acid residues can have similar chemical properties and/or biological conservation patterns. To capture these relationships, one could opt to use a Blocks Substitution Matrix (BLOSUM) [155] encoding. Essentially, BLOSUM matrices are constructed by extracting sequence alignments of conserved regions from protein families and calculating similarity scores. This results in an encoding where amino acids of similar side-chain properties are represented with similar numerical values, and this helps improve peptide-MHC predictions [154]. As a final note, it is important to mention that an additional letter may be added to any encoding schema to identify the wildcard amino acid; usually, "X" is used, and in most cases is just a vector of zeros.

To finally fit an ANN with biological sequencing data, one may fall in the temptation of randomly splitting the data into K partitions, encode them and then perform a Cross Validation training. This, however, cannot always be done so straightforwardly. The problem with biological sequencing data is that it tends to be highly redundant, in the sense that certain sequences may overlap other sequences in the string space, directly leading to information leakage during training. As an example, let's assume a $K = 2$ split of the training data; so, for any CV fold, we end up with a single partition for training, and a single partition for validation. Imagine now that inside the training partition the sequence YPLKYNHQYLYDV is present with a target value of 1 for a particular MHC; on the side, the validation partition has the sequence HKVKYNHQYLYPL, with a target value 1 for the same MHC. Examining carefully, one can see that both sequences share the common sub-sequence KYNHQYLY, and this has huge implications. During the training loop, the network is going to be fitted using the training sequence, and validated using the validation sequence. Since such overlapping exists, CV performance is going to be overconfident, because the sub-sequence is used to both update network weights and report validation metrics. Because of this, our network will surely end up overfitted, becoming worthless when faced with the real task of predicting

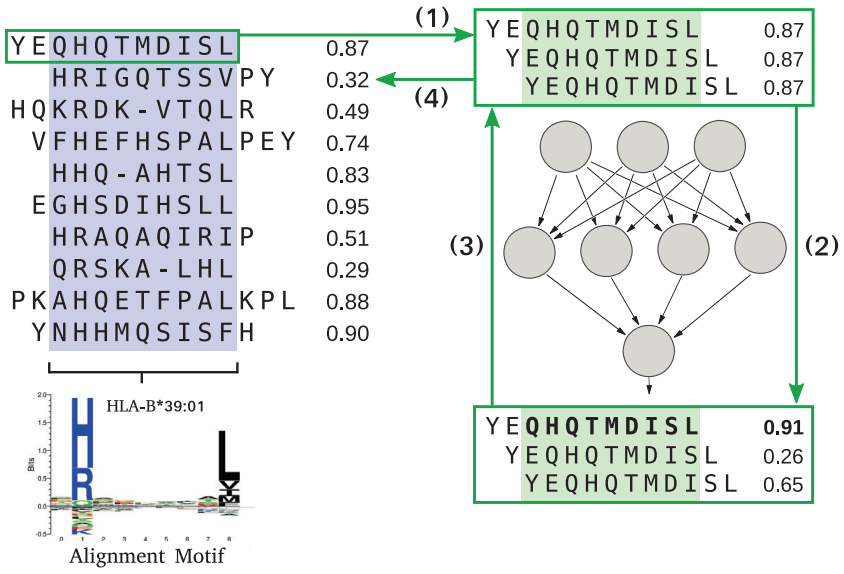

Figure 1.15. The NNAlign framework. In this example, MHC-I binding data (top left), consisting of the peptides of Figure 1.4 and their associated affinity values, is being fed to the algorithm. (1) An input sequence is selected and a sliding window of length 9 (in green) is applied to it. (2) Windowed sub-sequences are fed to a FFNN, resulting in different prediction values (for more details on this architecture, refer to Figure 1.8). (3) The sub-sequence whose predicted affinity is closest to the measured affinity (in bold) is annotated as a member of the final alignment core, and its corresponding prediction value is used for backpropagation. (4) The next input sequence is selected, and the process starts all over again. After completing all training epochs, the resulting alignment core (in blue) will correspond to the binding core of the MHC under study (in this example, the alignment motif for HLA-B*39:01 is displayed). Sequence logo was generated using Seq2Logo [29].

epitopes. To address this problem and regularize the network, train, validation and test data must be thoroughly curated to limit their sequence similarities. This is done by applying the Common Motif redundancy removal procedure [156], which is a customized version of the Hobohm1 algorithm [157]. In Common Motif, sequences with overlapping sub-sequences of length l are clustered together in the same partitions, mitigating inter-partition overlapping and enabling proper independent validation. As a result, the final output of the common motif algorithm are K orthogonal partitions ready to be used as training input for cross validation.

Apart from proper encoding and redundancy removal procedures, variable length peptides need to be aligned in order to be used as training data for FFNNs. This occurs because such network architecture has a fixed input size, and thus a fixed-size alignment core becomes a fit candidate. However, this means that vanilla FFNNs will depend on a previous data alignment step in order to be trained, imposing a major bias for representation learning. To overcome this, a self-contained approach was introduced with the NNAlign [158] algorithm, consisting of a neural network capable of aligning peptide sequences derived from BA assays while simultaneously identifying MHC binding core motifs (Figure 1.15). NNAlign is essentially a feed forward network that updates its weights by means of selecting the top scoring sub-peptide within a target input peptide. In general, the training procedure consists in picking a peptide, feeding all the sub-peptides of length l (i.e. for MHC-I peptides, $l = 9$ is usually a good choice), selecting the top scoring sub-peptide and then applying backpropagation using the corresponding loss value. This is repeated for all peptides in the training set, for all epochs. After training, the collection of all the selected sub-peptides will correspond to a heuristic alignment core of the training data, enabling the posterior extraction of the corresponding MHC binding motif. Later, a second version of NNAlign [159] introduced the capability of applying insertions and deletions to the input peptides, critically expanding the alignment space of the algorithm. Thanks to this, and besides improving general performances, NNAlign-2.0 helped interpreting modes of binding for MHC-I [160] and identifying non-canonical binding modes for MHC-II [161].

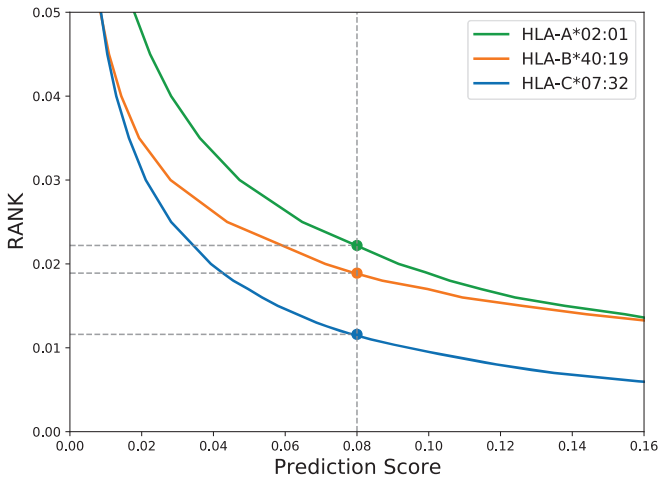

Figure 1.16. Example of RANK curves for three different HLA-I alleles. In this zoomed-in region, the relationship between prediction scores and RANK values can be easily observed, with an unique prediction score of 0.08 resulting in three different RANK values (one for each allele). This type of score transformation is crucial for pan-specific algorithms to work properly.

Besides creating peptide binding predictors for specific MHCs, NNAlign is also used as the training engine for so-called pan-specific models. The major difference between an allele-specific model and a pan-specific one is that the latter incorporates MHC information into the training loop. In particular, such information encodes MHC sequences based on polymorphic residues in close vicinity to the peptide in the binding groove (we term this subsequence as pseudosequence). Two examples of pan-specific methods are NetMHCpan-4.0 [162] and NetMHCIIpan-3.2 [163], which have been shown to successfully extrapolate peptide binding prediction to MHCs with limited data coverage [164], enabling the potential to predict binding to any MHC molecule of known sequence. An important take on pan-specific methods is that, since different MHCs bind to peptides with different strengths, a normalization schema is needed in order to correctly interpret prediction scores. Usually, a ranking approach is employed, where predictions of random peptides against each available MHC are ordered from high to low, and then relative positions are associated to predictive outcomes. This results in the so-called RANK curves (see Figure 1.16), where the ranking of a given peptide can be obtained using its associated predicted value. With this transformation, prediction scores across different MHCs can be compared; also, on a side note, the more intuitive percentual rank score, defined as %RANK = RANK * 100, is usually used.

As time passed and technology improved, EL SA data became increasingly abundant, and this pushed forward the development of new ways of incorporating such type of data in combination with classic BA datasets. An essential contribution towards integrating these data types was the two output neuron architecture introduced in Jurtz et al. [162]. As shown in Figure 1.17, in this architecture BA and EL data get a dedicated output neuron each. Then, during training, the network only backpropagates from the output neuron matching the data type that was fed to the input. Since weights between the input and hidden layers are shared between both data types, learned representations become shared across EL and BA data as well, leveraging the best of both worlds.

As everything else in life, time and technology continued to move on, and EL MA data started to become abundant as well. However, since training a model on MHC peptide data is inherently a supervised task, unambiguous MHC annotation of EL MA data became a major limitation. Because of this, prior deconvolution of immunopeptidomes was needed in order to "convert" EL MA data into EL SA. Such step relies on external unsupervised clustering algorithms such as GibbsCluster [165, 166], which is able to simultaneously align and cluster an input peptide collection spanned from mixed MHC receptor specificities. Each possible cluster is represented by a PSSM, and the method aims at maximizing the

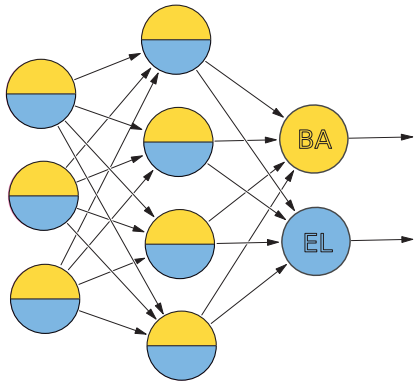

Figure 1.17. Two output neuron FFNN architecture. A color code is used to represent BA data (yellow) and EL data (blue). Notice how the input and hidden layers of the neural network share both data types, whereas the two output neurons do not.

information content of individual matrices while minimizing the overlap between distinct clusters following a Monte Carlo sampling approach. As a result, GibbsCluster reports the optimal amount of clusters for the provided input and the corresponding alignment of each cluster. Then, after peptide clustering, an intervention is needed to annotate each cluster to an MHC restriction. Usually, annotation can be done with prediction models [167], or visual inspection against known MHC binding motifs, but this often leads to biased and inaccurate annotations. Also, the integrity of training data will rely on the accuracy of the clustering algorithm, adding an extra limitation. After clustering, neural networks can be trained using the NNAlign algorithm in order to generate predictive models for peptide-MHC binding. To dig further into the approach of prior external deconvolution for posterior neural network training, refer to the second chapter of this thesis.

To tackle the above mentioned limitations, further development of current algorithms was needed. In particular, NNAlign was promoted to NNAlign_MA, a new, pan-specific version capable of single-handedly dealing with EL MA data deconvolution, training and annotation. To do this, such an algorithm exploits the co-occurrence of MHCs across datasets together with the exclusion principle, under a semi-supervised learning schema. For a given cell line, NNAlign_MA annotates each one of its measured peptides to the top-scoring MHC from the list of MHCs expressed by the cell line. It achieves this by (1) kick-starting the network training using only BA + EL SA data for a short quantity of epochs, (2) annotating MA data by means of assigning each peptide to its top-scoring MHC prediction, thus mapping MA data to the SA datatype, and (3) using this newly mapped data to perform backpropagation over the network weights (Figure 1.18). MA data annotation is repeated after every epoch, allowing the model to update peptide-MHC assignment beliefs on the go. Also, the algorithm introduces a customized prediction rescaling technique, capable of leveling out differences in data availability across MHCs, improving motif annotation. With this training strategy, NNAlign_MA is able to fully leverage immunopeptidomics data (up to date consisting of BA, EL SA and EL MA datasets), expanding the learning capacities of its previous version and improving identification of T-cell epitopes. To see a more in-depth analysis on how NNAlign_MA was designed, trained and validated, refer to the third chapter of this thesis.

Given the pan-specific nature and improved performance of NNAlign_MA, new upgraded versions of NetMHCpan and NetMHCIIpan were generated using such a framework. The new NetMHCpan suite was trained on rather comprehensive data -spanning BA, EL SA and EL MA datasets-, and tested on independent ligands and epitopes, exhibiting an overall increased predictive power. Finally, both NetMHCpan and NetMHCIIpan were uploaded to the internet as web-servers of free access to the public. To read more about how these upgrades were implemented and deployed, refer to the fourth chapter of this thesis.

Finally, after having completed the task of improving current MHC motif deconvolution, annotation and peptide binding prediction methodologies, we ventured into the field of deep

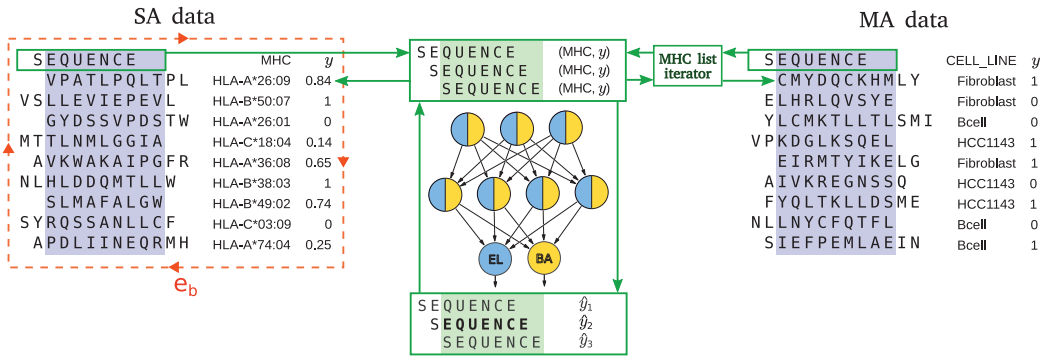

Figure 1.18. The NNAlign_MA framework. In this example, toy SA (leftmost panel) and MA (rightmost panel) data are used to train the two-output FFNN architecture from Figure 1.17 (middle panel). First, only the SA dataset (composed of sequences, single MHC restrictions, and real- and boolean-valued target values) is fed to the network for a short burn-in period lasting e_b epochs (dashed orange block and arrows). Usually, $e_b = 20$ is used. During burn-in, unambiguous information from single-restriction SA instances is imprinted into the network weights as a means of "initial ground truth". This is done following the NNAlign procedure explained in Figure 1.15 (windowed sequences with target value y are fed, and closest predicted value $\hat{y}$ is used for backpropagation and alignment core conformation), with the aggregation of MHC information (NNAlign_MA is a pan-specific method). After burn-in concludes, the network is in place to start assigning single MHC restrictions to multiple-labeled MA dataset (consisting of sequences, cell line identifiers and boolean target values). To do so, an "MHC list iterator" block is annexed in between the neural network and the MA data. Such a block is in charge of cycling and feeding, one by one, all MHCs expressed by the current peptide's cell line. In this way, not only a proper binding core is selected for the input peptide, but also its best matching MHC allele. Afterwards, the corresponding predicted value is used for backpropagation. The MA data training and annotation proccess is alternated with SA data training, and repeated until all training epochs are concluded.

learning with a more exploratory spirit. Taking advantage of the astounding capabilities of 1-dimensional convolutional neural networks, we defined a particular architecture that enabled us to formulate mathematical "projections" of input data onto the network's weight space. This led to interesting results on how CNN topologies like the one proposed here construct internal representations of MHC binding motifs, suggesting how such representations could be extracted from the CNN with the purpose of learning the rules of MHC-peptide interaction. To take a closer look on how this exploration was done, refer to the fifth chapter of this thesis.

Chapter 2

A first approach to motif discovery in immunopeptidomics data

2.1 Summary

This chapter presents the article "Computational Tools for the Identification and Interpretation of Sequence Motifs in Immunopeptidomes", in which a first generation approach is introduced to characterize and exploit MHC binding motifs from immunopeptidomics data.

The aforementioned approach consists in the combined application of GibbsCluster-2.0 and NNAlign-2.0 (two publicly and freely available web-servers) to deconvolute experimentally-acquired immunopeptidomes and train peptide-MHC binding predictors using such deconvolution output, respectively. The datasets employed correspond to Mass Spectrometry assays of HLA-I and HLA-II ligands, where multi- and mono-allelic cell lines (respectively expressing multiple and single HLA proteins) were utilized.

Due to their simplicity, HLA-I mono-allelic cell lines are first analyzed. For such data, GibbsCluster is applied as a filter of spurious sequences, which we hypothesize are generated during the experimental wet lab procedure. It follows the analysis of HLA-I EL MA data, where GibbsCluster is implemented as both deconvolution and filtering algorithm. Then, HLA-II mono- and multi-allelic cells are filtered and clustered using the same approach followed for HLA-I datasets. After deconvolution and assignment of HLA restrictions, peptide-HLA binding prediction models are generated for both HLA-I and HLA-II. To do this, the NNAlign neural network framework is used as a training algorithm.

Results show that GibbsCluster enables immunopeptidomics motif deconvolution and NNAlign is able to effectively capture deconvoluted HLA's binding preferences. Jointly, these two algorithms serve scientists as a first-hand approach to identify motifs contained in immunopeptidomes and train prediction models to conduct epitope discovery.

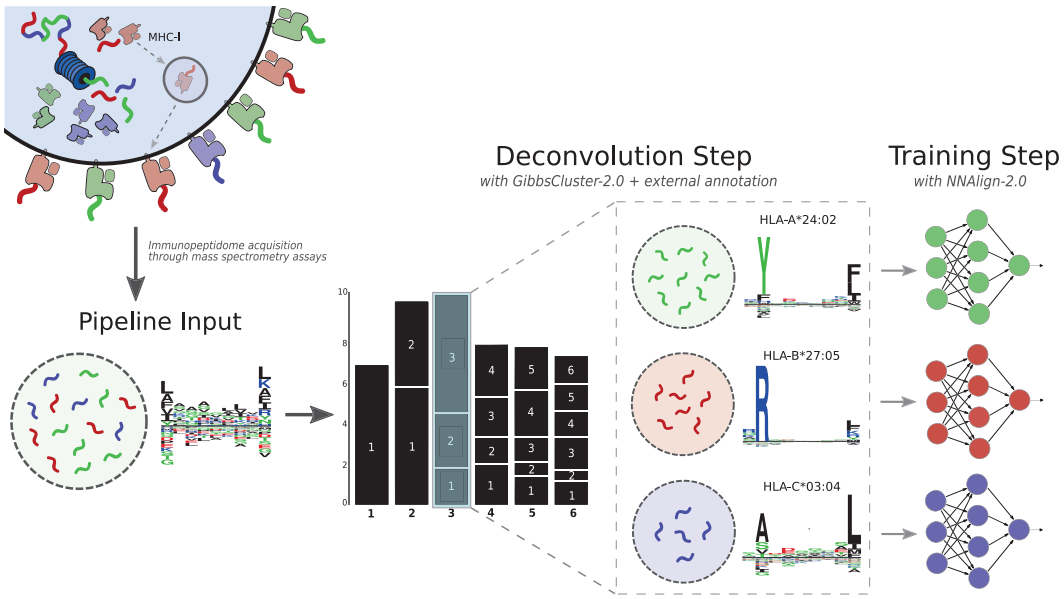

Figure 2.1. Graphical abstract of chapter two. On the top left, a theoretical cell line expressing three types of MHC-I (green, red and purple) is being characterized. After applying the corresponding MS/MS procedures, a collection of peptide sequences with mixed MHC restrictions is assembled in order to feed the pipeline. Notice how, prior to the deconvolution step, the sequence logo of such a collection is just noise. GibbsCluster-2.0 is afterwards applied to the input, and the top scoring clustering solution (tallest stack of vertical bars, corresponding to three groups) is selected. Cluster logos are also reported, and their binding motif annotation is assigned externally, by means of visual inspection or prediction using external tools. Afterwards, three peptide-MHC binding predictors are trained using the NNAlign-2.0 software, each one modeling the preferences of the original MHCs expressed by the cell line under study.

2.2 Paper I

Computational Tools for the Identification and Interpretation of Sequence Motifs in Immunopeptidomes

Proteomics, January 2018, Volume 18, 1700252, https:doi.org/10.1002/pmic.201700252

Bruno Alvarez[1,†], Carolina Barra[1,†], Morten Nielsen[1,2] and Massimo Andreatta[1,⋆]

[1]Instituto de Investigaciones Biotecnológicas, Universidad Nacional de San Martín, CP1650 San Martín, Argentina
[2]Department of Bio and Health Informatics, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark

⋆ Corresponding author (massimo.andreatta@gmail.com)
† These authors contributed equally to the paper as first authors

Abstract

Recent advances in proteomics and mass-spectrometry have widely expanded the detectable peptide repertoire presented by major histocompatibility complex (MHC) molecules on the cell surface, collectively known as the immunopeptidome. Finely characterizing the immunopeptidome brings about important basic insights into the mechanisms of antigen presentation, but can also reveal promising targets for vaccine development and cancer immunotherapy. This report describes a number of practical and efficient approaches to analyze immunopeptidomics data, discussing the identification of meaningful sequence motifs in various scenarios and considering current limitations. Guidelines are provided for the filtering of false hits and contaminants, and to address the problem of motif deconvolution in cell lines expressing multiple MHC alleles, both for the MHC class I and class II systems. Finally, it is demonstrated how machine learning can be readily employed by non-expert users to generate accurate prediction models directly from mass-spectrometry eluted ligand data sets.

Introduction

The comprehensive set of peptides presented on the cell surface by major histocompatibility complex (MHC) molecules, collectively referred to as the immunopeptidome, represents a unique fingerprint of the health of a cell. T lymphocytes routinely scan this pool of MHC-associated peptides, and can help eliminating infected or cancerous cells that present abnormal peptides on their surface. MHC class I molecules mainly bind peptides derived from intracellular pathogens (such as viruses and some bacteria) and present them to cytotoxic T lymphocytes; MHC class II epitopes are mainly derived from extracellular proteins and arepresented to T-helper lymphocytes.

Recent technological advances in the field of mass spectrometry (MS) have brought about a revolution in the study of immunopeptidomes (reviewed by Caron et al. [168]), with several thousands of peptides that can be detected in a single experiment. Large data sets of naturally presented peptides have been beneficial to define more accurately the rules of peptide-MHC binding [162, 169, 170] and have also a tremendous potential in defining pathogen-derived T-cell epitopes [171, 172] and neo-epitopes unique to cancerous cells [44, 173–175]. Part of the appeal of MS-based approaches is that they do not require prior knowledge of MHC motifs, and there is no human intervention in defining a library of candidate sequences to be tested. Therefore, MS provides a large but relatively unbiased sampling of the population of processed and presented peptides available for T-cell recognition. [170]

In most MS-based pipelines, spectra from eluted peptides are matched against a reference database of natural proteins using algorithms like MaxQuant [176] or PEAKS [48,177], and filtered against a decoy database to limit the false discovery rate (FDR). Strict FDR filters (typically in the order of 1%) should ensure that most spectra are correctly assigned to bona fide ligands, but often leads to discarding a large portion of the spectra. Several approaches have been proposed to increase the yield of spectral assignment. For example, Mascot Percolator performs machine learning on high-confidence matches to rescore database search results for lower-confidence peptides [178]. Instead of matching spectra to an entire protein database, SpectMHC constructs reduced, targeted databases of potential MHC ligands, effectively reducing the amount of spurious decoy hits. [179] Recent work has also suggested that a portion of the unassigned spectra may also be explained by proteasome-generated spliced peptides, which would require the inclusion of spliced variants in the target database [180, 181].

After spectral assignment to amino acid sequences, peptides must often be aligned and/or clustered to extract meaningful sequence motifs of antigen presentation. The analysis protocols here will generally differ depending on the type of receptor (MHC I vs MHC class II) and type of sample used (cellular vs soluble MHC molecules and mono- vs poly-allelic cell lines). On one hand, MHC I ligands have a limited range of lengths, typically 8–11 amino acids long, and are characterized by very conserved amino acid preferences at the positions interacting with the MHC binding groove (anchor positions). On the other hand, MHC II ligands are normally longer, with only a portion, the binding core, directly interacting with the MHC groove [182]; in this case a more sophisticated alignment process

is needed to extract conserved binding preferences. In transgenic cells expressing a single MHC molecule (mono-allelic), only one specificity is expected to be present in the data and motif identification is relatively straightforward. Conversely, unmodified cells will naturally present peptides bound to multiple MHC alleles (up to six for HLA class I), with generally different binding preferences; in this case, the multiple specificities contained in the data must be deconvoluted, either by assigning MHC restriction with predictive methods, or by unsupervised clustering.

A popular tool for the unsupervised identification of sequence motifs in immunopeptidomes is GibbsCluster [165, 166], a web-based and downloadable method that has been included into numerous pipelines for the deconvolution of ligand motifs in the MHC class I [40, 175, 183, 184] and MHC class II [185–187] systems. The GibbsCluster algorithm takes as input a list of peptide sequences (potentially of variable length), and uses a heuristic search to group them into information-rich groups. Besides the sequence motif defining each group, additional properties such as the ligand length distribution of each cluster can be analyzed. A similar method, MixMHCp [169, 188], has shown performance comparable to GibbsCluster, with the limitation that it can only handle peptides of uniform length. A useful feature of GibbsCluster is the "trash cluster," a check on internal motif consistency that can filter out outliers that cannot be assigned to any clusters. In the context of MS eluted ligand data, spurious data points can originate both from LC-MS/MS contaminants and from erroneous spectral matches. As a noise filter, GibbsCluster can be beneficial also for mono-allelic data sets where no motif deconvolution is required.

While sequence motifs are generated by GibbsCluster in an unsupervised manner, the method cannot directly assign the MHC restriction of each ligand; this must be done by comparing the unsupervised motifs with published binding motifs of the MHC molecules in the sample. [189] While this comparative approach is in most cases feasible for human MHC, whose most prevalent alleles have been well characterized and documented, it will fall short for samples containing uncharacterized specificities. Aiming to overcome this limitation, Bassani-Sternberg et al. [188] suggested a strategy for automatic, unbiased annotation of MHC restriction by comparing motifs detected in multiple data sets with known haplotypes. Exploiting the co-occurrence of MHC alleles across different data sets, they were able to assign motifs to individual alleles without relying on a priori assumptions on their binding specificity, also for alleles without previously documented ligands.

Over the past decades, many efforts have been dedicated to the development of computational methods for the prediction of peptide binding to MHC class I molecules. Most of these T-cell epitope prediction methods have been traditionally trained solely on in vitro data of peptide-MHC binding affinity. Although peptideMHC affinity is arguably the most selective step in antigen presentation, other factors influence the likelihood of a peptide being presented on the cell surface for T-cell recognition [190, 191]. In vitro binding affinity data does not address the fact that antigen presentation is a complex, integrative physiological process that combines antigen processing, transport and binding affinity/stability of the peptide-MHC complex. Finally, in vitro data fails to reflect any peptide length preference of different MHC-I alleles. Because naturally eluted ligands incorporate information about these additional properties of antigen presentation, large MS-derived sets of peptides can potentially enable the generation of more accurate prediction models. Recent studies have suggested that models trained on MHC class I ligand data outperform binding affinity-based predictors when it comes to identification of eluted ligands and T-cell epitopes, both in an allele-specific setting [169, 170] as well as with pan-allelic coverage [162]. Generic tools for machine learning from peptide sequences such as NNAlign [159, 192] can be applied to individual MS data sets to generate custom-made prediction models, which can in turn be employed for further downstream analyses of the immunopeptidome.

The rapidly expanding collection of naturally eluted ligands revealed by MS and the analysis toolkits developed in its wake hold great promise in understanding the structure of the immunopeptidome and the rules of antigen presentation. However, because of the complexities inherent to MS eluted ligand data, it is not a trivial task to analyze and interpret the information these data sets contain. In this report, we seek to address some common issues and describe strategies to analyze MS ligand data and derive sequence motifs in the various scenarios outlined above (MHCI vs MHCII; mono-allelic vs poly-allelic cell lines),

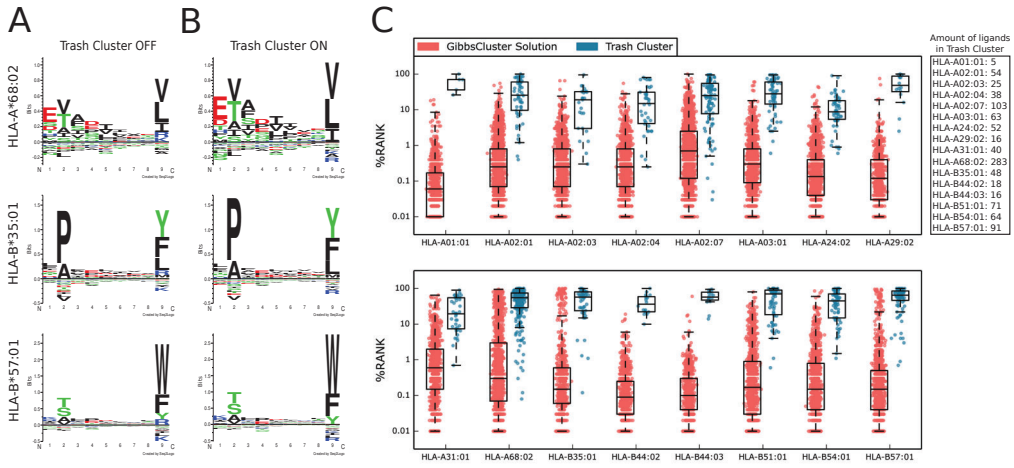

Figure 2.2. Visualizing motifs and removing contaminants with GibbsCluster. Sequence motifs of three representative alleles (A) before trash cluster filtering and (B) after filtering. The post-filtering motifs have higher information content and lack the putative K/R contamination at P9. C) Distribution of NetMHCpan-3.0 percentile rank scores for peptides in the main cluster (red) and in the trash cluster (blue).

with guidelines and examples on publicly available datasets.

MHC Class I, Mono-Allelic Cells

In a recent publication, Abelin et al. [170] described the development of transgenic cells that express a single human MHC class I allele (HLA), and used them to generate a large set of MHC ligands covering 16 HLA class I alleles. There are obvious advantages in using mono-allelic cells to characterize MHC ligands: firstly, no deconvolution/clustering is required to define motifs at the single-allele resolution; secondly, the assignment of individual peptides to their allele does not have to depend on binding predictions or prior knowledge of the motifs. Apart from technical difficulties in the cell generation, a possible drawback is that the relative level of expression of different MHC alleles in a given cell, and the amount of ligands they present, is lost in a monoallelic setting. The amount of ligands presented by different alleles may also depend on competition between MHC molecules, where the newly available digested peptides from an unfolding antigen fragment would presumably be captured by MHCs with the highest affinity [193].

Although most software for MS spectra mapping uses a strict false discovery rate (FDR) threshold, incorrect ligands may still be present among the matches that pass the FDR check. These may consist of common contaminants such as keratin or histone proteins, as well as residual peptides from previous runs of the LC-MS/MS instruments used for sample preparation [194,195]. GibbsCluster is a useful tool to detect and remove such contaminants and false hits. For each allele in the Abelin data set [170], we applied GibbsCluster-2.0 with default preset options for "MHC class I ligands of lengths 8–13," specifying a single cluster. Between 0.4 and 16% of the peptides (mean 4%) of lengths 8–13 were inconsistent with the motif identified by GibbsCluster-2.0 and were removed by the program as noise. While distinct motifs can be discerned before trash cluster filtering (see three representative alleles in Figure 2.2-A), the post-filtering motifs have higher information content and more well-defined anchor residues (Figure 2.2-B). Peptides in the "trash cluster" may sometimes hint at the origin of the contamination: for example, the observation of terminal arginine/lysine preferences at the C-terminus in several of the 16 alleles points towards tryptic peptides polluting the mixtures (Supplementary Figure 2.S8). The ligands in the Abelin data set have in general very good correspondence to known MHC binding preferences, with an average NetMHCpan-3.0 percentile rank [196] well below 1% for most alleles (Figure 2.2-C, red boxplots). In contrast, peptides in the trash cluster match very poorly the preferences of their MHC and are assigned high NetMHCpan rank scores (Figure 2.2-C, blue boxplots).

MHC Class I, Poly-Allelic Cells

Unmodified antigen-presenting cells will generally express up to six different MHC class I alleles (two each for HLA-A, HLA-B, and HLA-C). The immunopeptidome of these cells therefore consists of multiple specificities mixed together, where the global haplotype is known but the restriction of each individual ligand is unknown. For example, Bassani-Sternberg et al. [40] described the LC-MS/MS analysis of peptides eluted from seven different cancer cell lines and primary cells, which had been HLA-typed at high resolution, and demonstrated how the GibbsCluster approach could be used to deconvolute the individual peptide restrictions. Here we illustrate the application of GibbsCluster to one of the cell lines from the Bassani-Sternberg study, HCC1143, which expresses the five alleles HLA-A*31:01, HLAB*35:08, HLA-B*37:01, HLA-C*04:01, HLA-C*06:02.

GibbsCluster finds an optimal solution of four clusters, with a close correspondence to all but one of the HC1143 alleles (Figure 2.3), failing to separate HLA-C*04:01 ligands. HLA-C molecules have low expression levels and rather degenerate binding preferences, [188, 197] making the deconvolution of their motifs more challenging. The motifs determined by unsupervised clustering show a remarkable correspondence with the binding preferences predicted by NetMHCpan-3.0 [196]. There are, in some instances, subtle differences between the NetMHCpan and GibbsCluster motifs, as in the case of additional secondary anchors (e.g., a positively charged P5 for HLA-B*37:01). This suggests that motifs directly derived from eluted ligands may carry an additional level of information on peptide presentation (for instance, secondary anchors conferring improved peptide-MHC complex stability) compared to the NetMHCpan motifs, which were constructed from in vitro binding affinity data. The sizes of the clusters give an indication of the relative level of expression of the different alleles, with the largest group corresponding to the homozygous HLA-A*31:01 (1253 peptides), followed by the two HLA-B alleles (610 and 460 peptides, respectively) and by the lowly expressed HLA-C*06:02 (409 peptides). Finally, 45 peptides were collected by the trash cluster. Interestingly, for six out of seven cell lines in the Bassani-Sternberg data set, we noted a C-terminal enrichment for arginine/lysine in peptide discarded in the trash cluster (Supplementary Figure 2.S9). A similar observation was made for the Abelin data set discussed previously, and hints that residual peptides derived from trypsin digestion may often be present in the LC column.

As an alternative approach to unsupervised clustering, one can assign each peptide to a MHC allele using peptide-MHC binding prediction methods; then deriving sequence motifs from the resulting groups of peptides. We applied NetMHCpan [196] to the peptides eluted from the HCC1143 cell line, assigning peptides to the MHC molecule in the haplotype with the lowest predicted NetMHCpan percentile rank. If a peptide could not be assigned to any MHC molecule with rank 2%, then it was discarded in a trash cluster. While this setup mimics the GibbsCluster strategy, it has the very important difference that NetMHCpan utilizes known motif preferences of the MHC molecules to make the assignments, whereas GibbsCluster is unsupervised and requires no prior knowledge of the motifs. In the case of the HCC1143 cell line, the MHC molecules are all well characterized and the solutions found by the two approaches are remarkably similar (Supplementary Figure 2.S10). Assignment by NetMHCpan has the potential advantage that at least a fraction of peptides could be assigned to HLA-C*04:01, a specificity that was not detected by unsupervised clustering. However, in cases where the haplotype is not fully characterized, or when the known MHC alleles have poorly studied motifs, the assignment by NetMHCpan would fail. This is exemplified by a recent study of bovine MHC ligands [167], for which the motifs derived by GibbsCluster differed dramatically from the assignments made by NetMHCpan due to paucity of training data available to NetMHCpan for these alleles. Note also, that the number of ligands discarded to the trash cluster using this approach was more than ten times higher compared to those discarded by GibbsCluster (463 versus 45).

MHC Class II, Mono-Allelic Cells

Analyzing MHC class II binding data is for many reasons more complex compared to MHC class I. First and foremost, the HLA class II binding groove is open at both ends, accommodating peptides of a wide range of length by letting them protrude at either terminus of the nonamer binding core. Sophisticated alignment methods are therefore required to

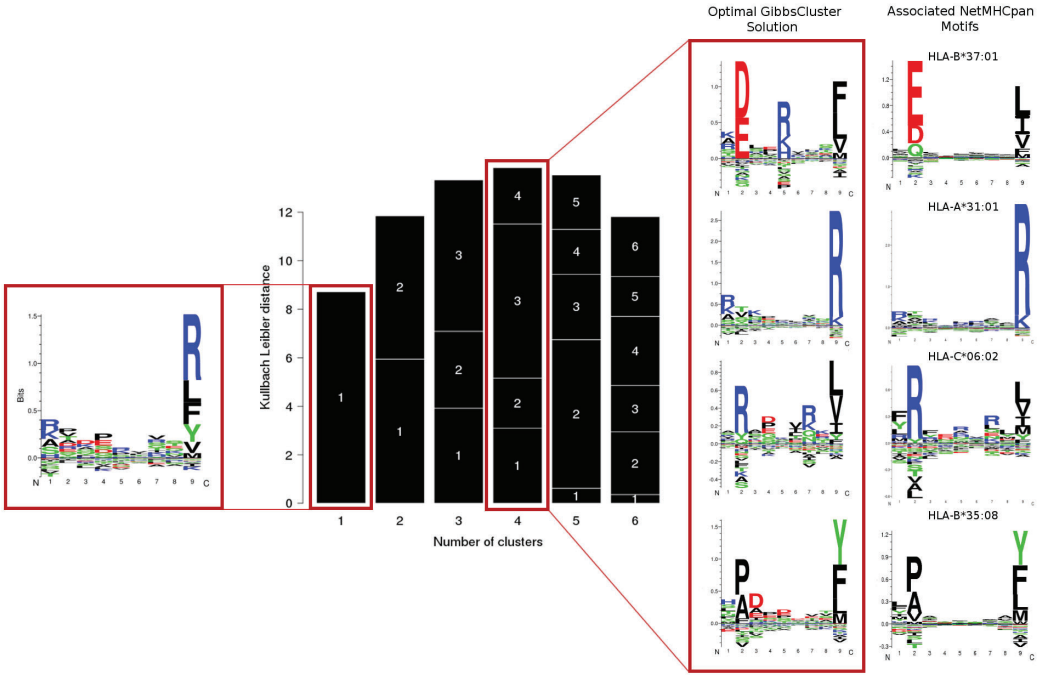

Figure 2.3. Clustering results for the HCC1143 cell line. The single cluster solution (left) is a mixture of multiple specificities, dominated by the most abundant alleles. The solution with highest information content corresponds to four clusters, with motifs highlighted in the red box (center). The motifs identified by unsupervised clustering show a remarkable correspondence with those predicted by NetMHCpan-3.0 (right). The GibbsCluster method was run using the default preset parameters for "MHC class I ligands of lengths 8–13," except for the number of iterations which was set to 100 (slower but more accurate), and number of groups, which was allowed to vary between 1 and 6. NetMHCpan logos were obtained from the NetMHCpan-3.0 website (`http:/www.cbs.dtu.dk/services/NetMHCpan-3.0/logos.php`) and were constructed from the top 1% scoring peptides from a large set of natural random peptides.

identify the conserved binding preferences of MHC class II molecules [161, 198, 199]. Secondly, the binding motifs for MHC class II are in general more degenerate compared to the highly conserved MHC class I motifs [200, 201]. These observations make the analysis and interpretation of MHC class II binding data, including MS ligands, highly challenging.

In a recent paper by Ooi et al. [202], MS eluted ligand data were used to investigate how patients expressing different HLA class II alleles have different susceptibility to autoimmune diseases. To characterize the specificity for each allele, they generated transgenic mice bearing the human HLA-DR1 MHC class II allele. On these data, we illustrate how the GibbsCluster method can be used to identify the binding motif of MHC class II molecules from mono-allelic MS ligand data and at the same time remove potential outliers. The 5740 non-redundant raw eluted peptide sequences were uploaded to the GibbsCluster web server, setting the recommended preset parameters for MHC class II peptides, except for the number of iterations per sequence per temperature step (set to 100) and the number of temperature steps (set to 50); these parameters entail a slower, but more accurate, motif search. The method recovered the binding motif for allele HLA-DRB1*01:01, with strong amino acid preferences at anchor residues at P1, P4, P6, and P9 (Figure 2.4-A). These preferences were observed both without (Figure 2.4-A, left panel) or with a trash cluster activated (Figure 2.4-A, right panel). By activating the trash cluster option with a threshold of 2, 179 peptides (3% of data) were removed, and the logo showed a 20% increase in information content (Figure 2.4-A, right panel).

MHC Class II, Poly-Allelic Cells

Another data set obtained from the Ooi et al. study [202] consists of peptides eluted from HW09013 cells that express the HLA-DR15/DR51 class II alleles. On this poly-allelic data set of MS eluted ligands, we set out to demonstrate how the GibbsCluster can be used to

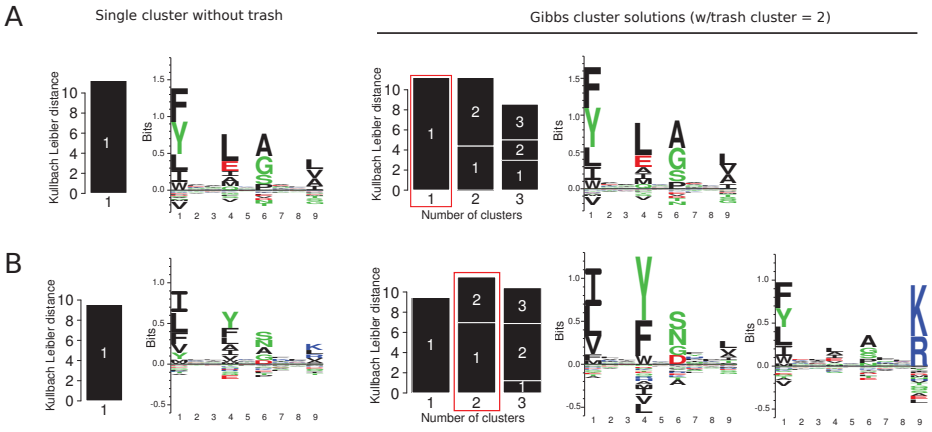

Figure 2.4. Sequence motifs identified by GibbsCluster-2.0 on MHC class II ligand data. The method identifies distinct amino acid preferences at the anchor positions P1, P4, P6, and P9 both without (left panels) and with (right panels) the trash cluster activated. A) Visualizing the motif and removing outliers from the mono-allelic human-DR1 mouse-transfected cell lines. B) Motif identification on mixed allelic data of DR15-DR51-EBV transformed cell lines.

separate multiple specificities in MHC class II ligand data. The set of 2782 unique eluted peptides was submitted to GibbsCluster, using the recommended preset parameters for MHC class II and allowing the program to search up to three clusters. The unfiltered, single-cluster solution shows a motif with the correct P1, P4, P6, and P9 anchor positions, but with low information content and preferences that are a mixture of the two alleles in the sample (Figure 2.4-B, left panel). Activating the trash cluster with a threshold of 2, the maximum information content is observed for the solution with two clusters (Figure 2.4-B, right panel). The amino acid preferences identified by GibbsCluster resemble previously published motifs derived from binding affinity data for HLA-DRB1*15:01 and HLA-DRB5*01:01 [192,203], and closely overlap with the global peptidome of DR15/51 characterized in a recent study [204]. Specifically, cluster 1 was composed of 1610 peptides (57.9%) and its motif resembles the HLA-DR15 binding preferences; cluster 2 comprised 1050 peptides (37.7%) and corresponds to the HLA-DR51 alleles; 122 peptides (4.4%) did not match to either group and were collected by the trash cluster.

In order to validate the solutions generated by GibbsCluster, we examined the composition of the clusters in terms of binding potential predicted by NetMHCIIpan-3.1 [205]. Both for the monoallelic DR1 and poly-allelic DR15/51 serotypes discussed above, we obtained predicted percentile rank scores for all peptides in the cluster solutions and in their relative trash cluster (Figure 2.5). The predicted median rank score for HLADRB1*01:01 in the DR1 cluster was 4% (first quartile (Q1) = 0.9, third quartile (Q3) = 12), whereas the trash cluster had a median rank score of 41% (Q1 = 20.5, Q3 = 75). In the poly-allelic data, cluster 1 was associated with HLA-DRB1*15:01, and showed a median rank score of 13% (Q1 = 5, Q3 = 30); cluster 2 was associated to HLA-DRB5*01:01 and obtained a median rank score of 4% (Q1 = 1.1, Q3 = 11); peptides in the trash cluster were evaluated against both alleles, assigning the best rank of the two, which resulted in an average rank score of 41% (Q1 = 23, Q3 = 75) (Figure 2.5). Overall, the NetMHCIIpan percentile score distributions suggest that the trash cluster could successfully collect peptides with very poor correspondence to the known preferences of the MHC class II molecules, and that probably derived either from incorrect spectral matches or from contaminants. The relatively high predicted rank values for the peptides mapped to the HLA-DRB1*15:01 cluster further suggest that the binding motif for this molecule predicted by NetMHCIIpan-3.1, which was trained on binding affinity data, shared a rather weak overlap with the binding motif contained within the MS ligand data. This observation underlines the high potential of MS ligand data to complement our knowledge on peptide characteristics required for MHC antigen presentation, as previously remarked for MHC class I [40,162,169,170,184].

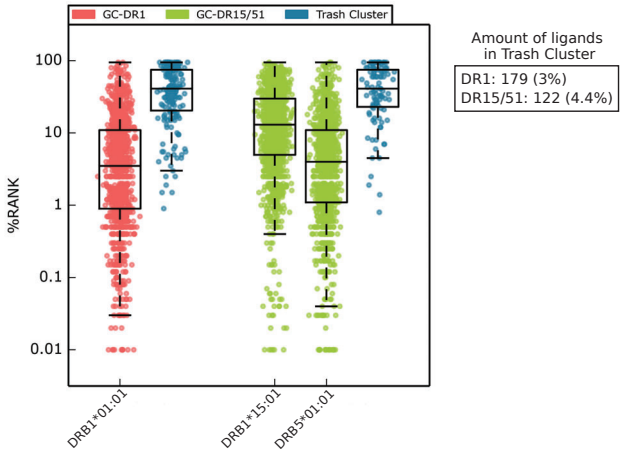

Figure 2.5. NetMHCIIpan percentile rank score for GibbsCluster solutions in the DR1 and DR15/51 data sets. Percentile rank scores were predicted by netMHCIIpan-3.1 for each GibbsCluster group with matching alleles present in MS data samples. In the case of the mixed allele dataset DR15/51, peptides in the trash cluster were scored by NetMHCIIpan to both DRB1*15:01 and DRB5*01:01, selecting the lowest rank score of the two.

Generating Prediction Models from MS Ligand Data

The approaches described so far in this report are mainly concerned with extracting and visualizing meaningful patterns within complex, often noisy, mixtures of peptides sequences. A further step is the generalization of the motifs identified in the data at hand, by constructing prediction models. Machine learning algorithms such as NNAlign [159], when provided with training examples suitably labeled (e.g., ligands vs non-ligands), can be instructed to automatically learn the features that distinguish positive from negative examples. Such models can then be applied on external data sets to discover more occurrences of the patterns learned on the training data. In the context of peptideMHC interactions, a good prediction model should have the ability to capture the binding preferences contained in the training data, both in terms of sequence motifs and peptide length distribution. In the next two sections, we illustrate some simple examples of prediction models directly constructed from MHC class I and class II eluted ligands.

MHC Class I Prediction Model

As an example application, we continue with the Abelin ligand elution dataset previously analyzed and filtered using GibbsCluster-2.0 (Figure 2.2). For each of the representative alleles HLA-A*68:02, HLA-B*35:01, and HLA-B*57:01, we prepared a training set consisting of post-filtering ligands (positive instances) and random natural peptides (negative instances). Positive instances were labeled with a target value of 1, negatives with a target value of 0. In line with earlier work [162], the amount of random negatives was imposed to be the same for each length 8–13, and corresponded for each length to five times the amount of positives for the most abundant peptide length. This uniform length distribution of the random negatives was adopted as a background against which machine learning can be employed to learn the amino acid and length preference of the natural binders.

On each of the three data sets, we trained a prediction model with the NNAlign-2.0 web server, using the recommended preset options for MHC class I ligands of variable length. In a crossvalidation experiment, the three models returned an area under the Receiver Operating Characteristic curve (AUC) of 0.961, 0.984, and 0.979, respectively. In order to derive the amino acid and peptide length preferences learned by the model, we used it to evaluate a large set of 900 000 random natural peptides with a flat length distribution, and extracted the top 0.1% scoring peptides. The composition of these high-scoring peptides should reflect the main preferences identified by the method to distinguish positive from negative instances. Indeed, the binding motif drawn from the top 0.1% peptides closely reflects the amino acid preferences of the training data (Figure 2.6-A,B). Moreover, all three methods could capture the preference for 9mer peptides over other peptide lengths; 10mers were moderately allowed, 8mers and 11mers were observed more infrequently (Figure 2.6-C).

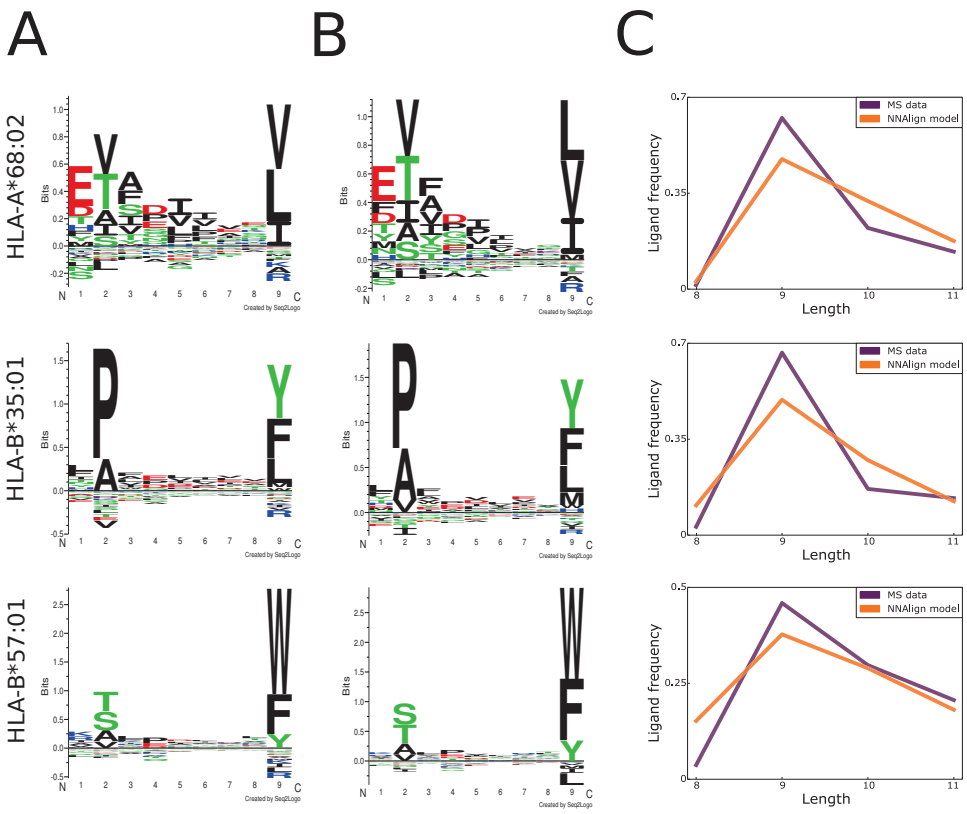

Figure 2.6. Generating prediction models from MS ligand data. A) Sequence motifs of the training data for three MHC class I alleles, aligned and filtered by GibbsCluster; B) sequence motifs captured by NNAlign-2.0; C) ligand length preferences in the training MS data compared to length preferences learned by the NNAlign model.

MHC Class II Prediction Model

To illustrate how the NNAlign framework can be used to construct MHC class II prediction models, we go back to the DR1 and DR15/51 data sets from Ooi et al. [202] previously filtered and clustered with GibbsCluster (Figure 2.4). To enrich the positive instances with artificial negative examples, a set of natural random negatives of lengths 11–19 amino acids was added to each eluted ligands data set. Positive instances were labeled with a target value of 1, negatives with a target value of 0. Similarly to the training set preparation described above for MHC class I, the amount of random negatives for each length corresponded to five times the amount of positives for the most abundant peptide length. For each of the three specificities deconvoluted by GibbsCluster in the DR1 and DR15/51 cells, we applied NNAlign2.0 to generate a prediction model, using the preset parameters for MHC class II recommended by the NNAlign server. For the mono-allelic DR1 serotype, all ligands except those removed by the trash cluster were used to train a model. For the DR15/51 cells, for which the clustering analysis revealed two separate specificities, we generated a separate model from the ligands contained in each of the two clusters.

The three models revealed high internal consistency, with cross-validated performance of AUC = 0.952, 0.974, and 0.952, respectively. NNAlign automatically generates a matrix (and logo) representation of the motif learned by the method, constructed from the top 1% scoring predictions from a large set of random natural peptides. We may compare the motifs learned by NNAlign to: i) the binding preferences in the MS training data, identified by GibbsCluster, ii) the GibbsCluster motifs identified in tetramer-validated epitopes extracted from the IEDB for the three DR molecules, iii) the binding preferences predicted by NetMHCIIpan-3.1 for these DR molecules. In general, the motifs learned by the NNAlign models share a remarkable overall correspondence to the preferences found by GibbsCluster for the MS ligand data, with similar amino acid enrichments at the anchor positions P1, P4, and P6, as well as the strong P9 for the DR51-associated ligands (Figure 2.7, first and

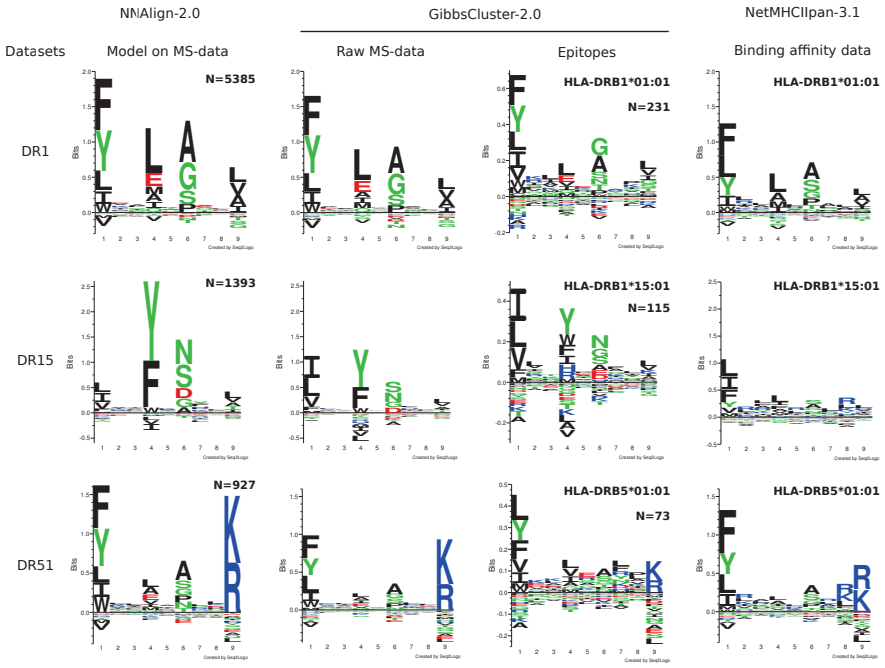

Figure 2.7. Comparison of motifs generated by different approaches for three HLA-DR alleles. NNAlign-2.0 motifs were obtained by training artificial neural networks on each MS data set, and evaluating 100 000 random peptides. The top scoring 1% peptides were used to build logos. Raw MS data were aligned, clustered, and filtered in an unsupervised manner using GibbsCluster, with a trash cluster threshold = 2. The same procedure was applied to tetramer-positive data downloaded from the IEDB. Note that due to small data set size, epitope logos are shown in a different y-axis scale. Binding motifs for NetMHCIIpan-3.1 were determined by evaluating 100 000 random peptides, and visualizing the core motif of the top 1% scoring sequences.

second columns). Likewise, the binding motifs constructed from the rather small amount of tetramer-validated epitopes obtained from the Immune Epitope Database (IEDB) [206] for the three DR molecules (231 for HLA-DRB1*01:01, 129 for HLA-DRB1*15:01, 73 for HLA-DRB5*01:01) correspond well with the motifs of the NNAlign models, and the MS ligand data (Figure 2.7, third column). In contrast, the logos derived from in vitro binding affinity data (NetMHCIIpan) in all cases show substantial differences to both the MS- and epitope-derived motifs (Figure 2.7, fourth column). These discrepancies are most pronounced for HLA-DRB1*15:01, where the NetMHCIIpan motif has weakly defined preferences at the anchor residues, and an enrichment of arginine (R) throughout the binding motif: a preference that is completely absent from the MS and epitope-derived motif. Another, more subtle difference is the enrichment of glutamic acid (E) at P4 in the MS and epitope motifs for HLADRB1*01:01; this preference is absent in the NetMHCIIpan motif. Finally, NetMHCIIpan displays a preference for R/K at position P8 for HLA-DRB5*01:01; this anchor is completely absent in the motif derived from MS and tetramer-validated epitope data. Taken together, these results show that ligand elution is a stronger correlate of epitope presentation than peptide-MHC binding affinity, suggesting that epitope prediction models may greatly benefit from incorporating MS eluted ligand data.

Discussion

The binding specificities of MHC molecules have been traditionally characterized using in vitro assays of binding affinity. The peptide-MHC binding data amassed through decades of painstakingly low-throughput experiments have had a tremendous contribution to the characterization of the binding preference for the most prevalent MHC molecules, and more generally to the understanding of the peptide repertoire available for T-cell recognition. However, because of the extreme polymorphism of the MHC-encoding genes, with up to several thousand allelic variants per locus, the full characterization of their specificities remains infeasible. Tandem mass spectrometry has emerged in the past decade as a powerful, high-throughput alternative for the identification of peptides eluted on the surface of

antigen-presenting cells.

The appeal of MS-based techniques does not only reside in the sheer amount of ligand data that can be detected in a single experiment. Because MS ligands are derived from a biological system that incorporates all properties of antigen presentation including binding affinity, binding stability, proper peptide processing and translocation, and impact of MHC binding chaperones, these techniques should capture additional signals besides the binding affinity measurable by in vitro assays. Accurate tools for the identification of sequence motifs in eluted ligand datasets are essential to interpret the patterns underlying the immunopeptidome and to benefit from this data deluge.

In this report, we described some straightforward, efficient approaches to extract motifs from immunopeptidomes in a number of scenarios commonly encountered in the field. We outlined analyses for MHC class I and class II, both in cell lines expressing a single MHC allele and in unmodified cells with multiple MHC allelic variants. GibbsCluster [166] is our tool of choice because it can effectively remove residual contaminants after FDR filtering, deconvolute multiple motifs in a mixture of peptides of variable length, and because it works both for MHC class I and class II ligands. In general, MHC class I molecules have strong, welldefined motifs, and even in samples containing several specificities it is often feasible to separate them into individual clusters. Unambiguously associating each cluster to individual MHC molecules remains an unresolved problem, especially for alleles with unknown binding motifs. So far only Bassani et al. [188] have attempted to tackle this question, exploiting the co-occurrence of MHC class I alleles across different data sets of known haplotype to assign motifs to individual alleles. More work along these lines is needed to automatically annotate the MHC restriction of peptides in poly-allelic datasets. The current implementation of GibbsCluster assumes that each peptide is restricted to one and only one MHC molecule. When cells express different alleles with similar binding motifs, or in the case of MHC class II ligands binding to multiple alleles in different alignment frames, it is likely that an individual peptide can act as ligand for multiple MHCs in a mixture. Future improvements to the algorithm should aim to address this limitation and account for potential multiple restrictions of individual ligands.

Ultimately, prediction methods can only be as good as the data used to train them. While MHC ligands sequences obtained by mass spectrometry show remarkable reproducibility and produce binding motifs consistent with those derived with more low-throughput assays, there remain several potential sources of error and bias in MS-based pipelines for ligand sequencing. For example, there is a documented underrepresentation of cysteine in MHC ligand data sets, as this amino acid interferes with MS precursor fragmentation [170, 188]. Different software tools for spectrum-peptide mapping use different functions to score candidate sequences, and they will generally identify nonidentical sets of ligands. Post-translational modifications (PTMs) have also been shown to have a role in shaping the MHC ligand repertoire [207]. However, accounting for such modified residues further complicates accurate spectrum-peptide matching and PTMs are often not comprehensively considered in MS pipelines. Finally, common contaminants such as keratin and histone proteins are often co-eluted with MHC ligands and add a level of noise to the sequenced immunopeptidome [194, 195]. Reducing biases and sources of error in the data-generation pipelines will also inevitably affect in a positive way the data interpretation and the prediction tools constructed on these data.

A number of recent reports have described the first prediction methods trained on MHC class I ligand elution data from mass spectrometry [162,170,189,208]. Their results indicate that methods trained on naturally presented peptides largely outperform prediction methods trained solely on in vitro binding affinity data when it comes to the identification of MHC ligands and epitopes. No reports have yet been published on models directly trained on MHC class II eluted ligands. Because the performance of MHC class II prediction methods still lags far behind their class I counterparts for epitope prediction, antigen processing factors are likely to play a major role in the generation of MHC class II ligands. Incorporating naturally processed ligand from MS experiments in the training pipelines of MHC class II prediction methods is an exciting and yet unexplored opportunity to close that gap. A simple but powerful approach to generate prediction models from ligand data is the NNAlign method [159]. We illustrated the construction of models from MS eluted ligands both for

MHC class I and MHC class II, and showed that they capture the preferences of the training data both in terms of binding motif and ligand length distribution. Taken together, these computational tools allow researchers to interpret motifs contained in immunopeptidomes and generate prediction models to scan protein databases for epitope candidates.

Supplementary Material

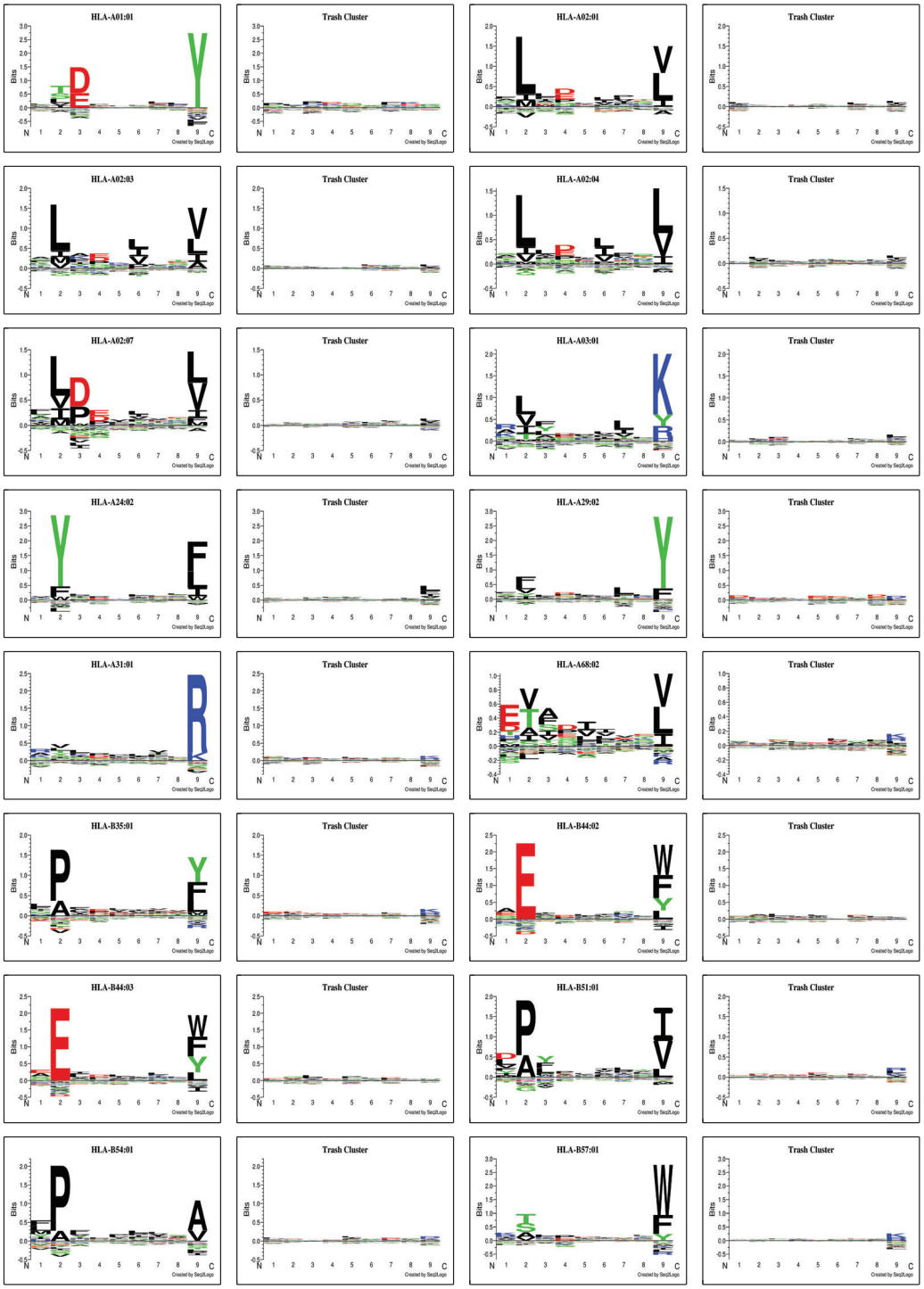

Figure 2.S8. Sequence motifs of peptides collected by the main cluster and by the trash cluster for the 16 alleles in the Abelin data set.

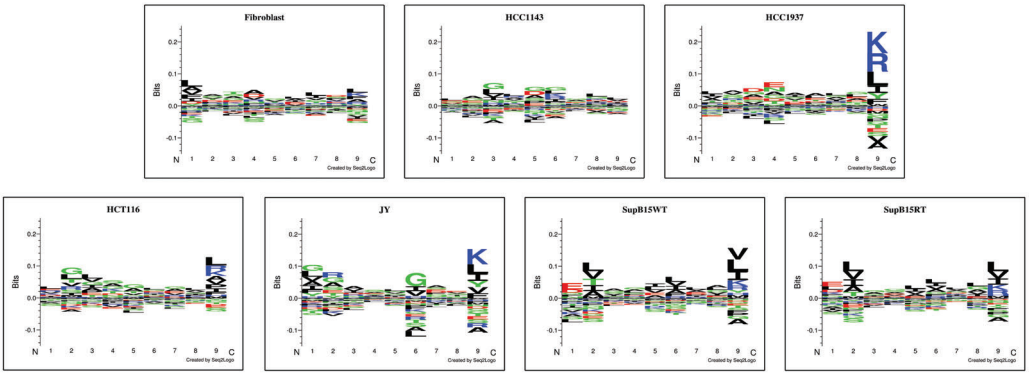

Figure 2.S9. Sequence motifs of peptides collected by the trash cluster on the 7 alleles in the Bassani-Sternberg data set.

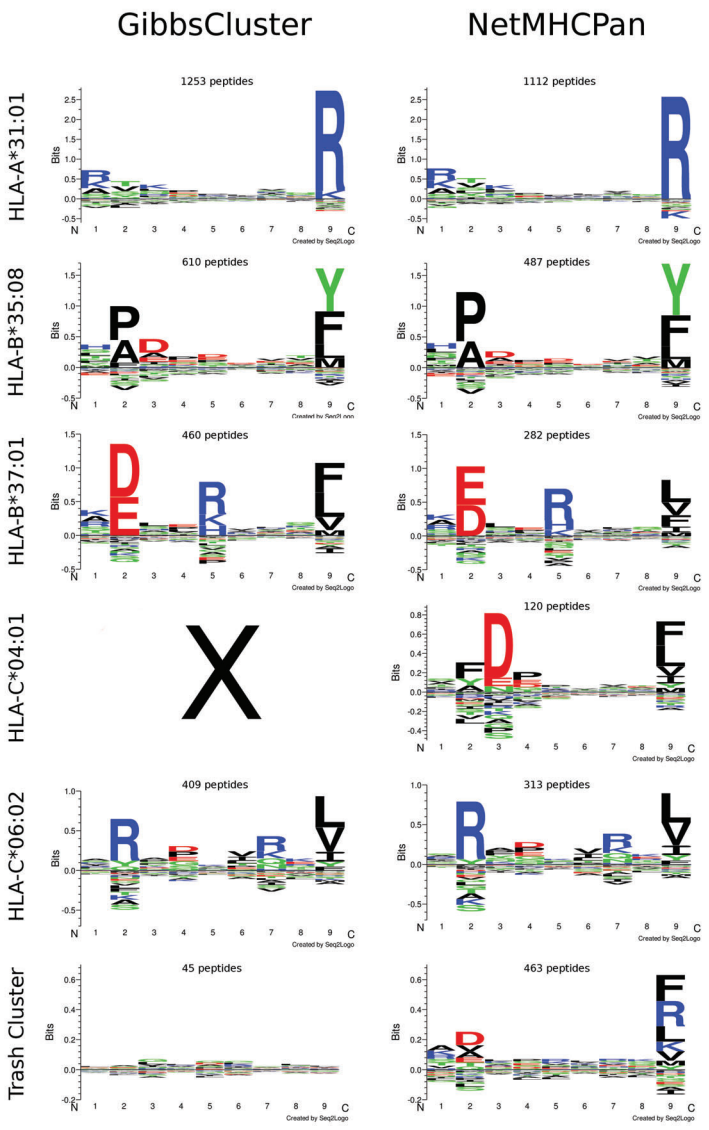

Figure 2.S10. Clustering of the HCC1143 cell line by GibbsCluster (left) and NetMHCpan (right). Sequences were assigned by NetMHCpan to the allele in the haplotype with the lowest predicted %Rank. If a peptide could be assigned to any MHC allele with %Rank $\leq 2\%$, then it was discarded to the trash cluster. Note that, in this case, GibbsCluster could not deconvolute HLA-C*04:01 peptides.

Chapter 3

NNAlign_MA: an improved motif discovery algorithm for immunopeptidomics data

3.1 Summary

This chapter introduces the article "NNAlign_MA; MHC Peptidome Deconvolution for Accurate MHC Binding Motif Characterization and Improved T-cell Epitope Predictions", in which a second generation approach is presented in order to improve characterization and exploitation of MHC binding motifs contained in immunopeptidomics data.

Such approach is an expansion of the NNAlign-2.0 software, termed NNAlign_MA, which introduces a custom training loop that enables shared usage of BA, EL SA and EL MA data. NNAlign_MA is capable of clustering peptides into individual MHC specificities and automatically annotate such clusters to an MHC molecule, while also training a pan-specific prediction model covering all MHCs present in the training set. This new NNAlign version represents a self-contained algorithm that overcomes the limitations of the GibbsCluster + NNAlign combined strategy presented in the previous chapter. Moreover, since it expands MHC allelic coverage of training data, identification of T-cell epitopes and natural ligands becomes boosted.

NNAlign_MA was extensively tested on data from three antigen presentation systems (HLA-I, HLA-II and Bovine BoLA-I), outperforming prior versions and other competitors. With this, we believe NNAlign_MA offers a state-of-the-art, standalone solution to analyze and exploit immunopeptidomes.

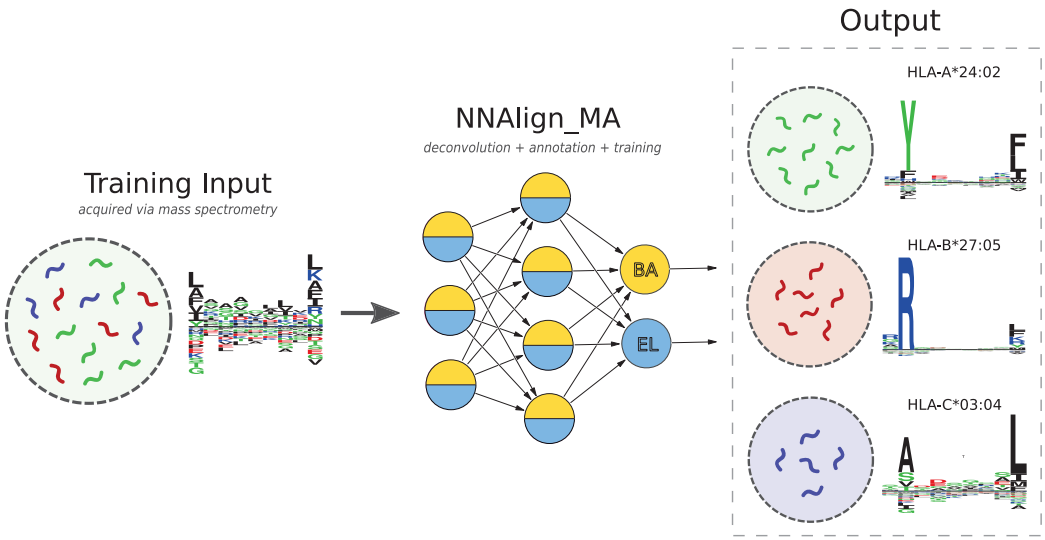

Figure 3.1. Graphical abstract of chapter three. Here, the same theoretical cell line present in the graphical abstract of chapter two (expressing green, red and purple MHCs) is being analyzed. After acquisition through Mass Spectrometry, the mixed EL MA peptide list is fed to the NNAlign_MA algorithm together with EL SA and BA datasets. During training, and together with sequence alignment, a full MHC deconvolution and annotation is performed upon the peptide input list. As a result, peptide-MHC associations are unambiguously assigned, enabling accurate predictions and reconstruction of the corresponding binding preference logos.

3.2 Paper II

NNAlign_MA; MHC Peptidome Deconvolution for Accurate MHC Binding Motif Characterization and Improved T-cell Epitope Predictions

Molecular & Cellular Proteomics, December 2019, Volume 18, 12,
https://doi.org/10.1074/mcp.TIR119.001658

Bruno Alvarez[1], Birkir Reynisson[2], Carolina Barra[1], Søren Buus[3], Nicola Ternette[4], Tim Connelley[5], Massimo Andreatta[1] and Morten Nielsen[1,2,⋆]

[1]Instituto de Investigaciones Biotecnológicas, Universidad Nacional de San Martín, CP1650 San Martín, Argentina
[2]Department of Health Technology, Technical University of Denmark, Lyngby, Denmark, DK-2800 Kgs. Lyngby, Denmark
[3]Department of Immunology and Microbiology, Faculty of Health Sciences, University of Copenhagen, Copenhagen, Denmark
[4]The Jenner Institute, Nuffield Department of Medicine, Oxford, United Kingdom;
[5]The Roslin Institute, Edinburgh, Midlothian, United Kingdom

⋆ Corresponding author (morni@dtu.dk)

Abstract

The set of peptides presented on a cell's surface by MHC molecules is known as the immunopeptidome. Current mass spectrometry technologies allow for identification of large peptidomes, and studies have proven these data to be a rich source of information for learning the rules of MHC-mediated antigen presentation. Immunopeptidomes are usually poly-specific, containing multiple sequence motifs matching the MHC molecules expressed in the system under investigation. Motif deconvolution -the process of associating each ligand to its presenting MHC molecule(s)- is therefore a critical and challenging step in the analysis of MS-eluted MHC ligand data. Here, we describe NNAlign_MA, a computational method designed to address this challenge and fully benefit from large, poly-specific data sets of MS-eluted ligands. NNAlign_MA simultaneously performs the tasks of (1) clustering peptides into individual specificities; (2) automatic annotation of each cluster to an MHC molecule; and (3) training of a prediction model covering all MHCs present in the training set. NNAlign_MA was benchmarked on large and diverse data sets, covering class I and class II data. In all cases, the method was demonstrated to outperform state-ofthe-art methods, effectively expanding the coverage of alleles for which accurate predictions can be made, resulting in improved identification of both eluted ligands and T-cell epitopes. Given its high flexibility and ease of use, we expect NNAlign_MA to serve as an effective tool to increase our understanding of the rules of MHC antigen presentation and guide the development of novel T-cell based therapeutics.

Introduction

Major Histocompatibility Complex (MHC) molecules play a central role in the cellular immune system as cell-surface presenters of antigenic peptides to T-cell receptors (TCR). Onpresentation, the peptide-MHC complex (pMHC) is scrutinized by T cells and an immune response can be initiated if interactions between the pMHC and TCR are established. The collection of peptides presented by MHC molecules is referred to as the immunopeptidome. Because of the extreme polymorphism of the MHC, immunopeptidomes can vary dramatically within a population, contributing to the personalized attributes of the vertebrate immune system.

Because of the essential role of the MHC in defining immune responses, large efforts have been dedicated to understanding the rules that shape the immunopeptidome, as well as its alterations in disease, either as a result of pathogen infection or cancerous mutation [209]. A crucial step toward defining the immunopeptidome of an individual is the characterization of the binding preferences of MHC molecules. The peptide-binding domain of MHC molecules consists of a groove, with specific amino acid preferences at different positions. MHC class I, by and large, loads peptides between eight and thirteen residues long [210, 211]. MHC class II molecules have an open binding groove at both ends and can bind much longer peptides, and even whole proteins [187, 212].

Peptide-MHC binding affinity (BA) assays represented the first attempts of studying binding preferences of different MHC molecules in vitro [34, 35]. However, BA characterization ignores many in vivo antigen processing and presentation features, such as protein internalization, protease digestion, peptide transport, peptide trimming, and the role of different chaperones involved in the folding of the pMHC complex [213]. Further, BA assays most often are conducted one peptide at a time, thus becoming costly, time-consuming, and lowthroughput. Recently, advances in liquid chromatography mass spectrometry (in short, LC-MS/MS) technologies have opened a new chapter in immunopeptidomics. Several thousands of MHCassociated eluted ligands (in short, EL) can with this technique be sequenced in a single experiment [168] and numerous assessments have proven MS EL data to be a rich source of information for both rational identification of T-cell epitopes [170, 214] and learning the rules of MHC antigen presentation [40, 215].

In this context, we have demonstrated how a modeling framework that integrates both BA and EL data achieves superior predictive performance for T-cell epitope discovery compared with models trained on either of the two data types alone [162, 215]. In these studies, the modeling framework was an improved version of the NNAlign method [159], which incorporated two output neurons to enable training and prediction on both BA and EL data types. In this setup, weight-sharing allows information to be transferred between the two

data types resulting in a boost in predictive power. For MHC class I, we have demonstrated how this framework can be extended to a pan-specific model, capturing the specific antigen presentation rules for any MHC molecule with known protein sequence, including molecules characterized by limited, or even no, binding data [162, 167, 216].

Except genetically engineered cells, all nucleated cells express multiple MHC-I alleles and all antigen presenting cells additionally express multiple MHC-II alleles on their surface. The antibodies used to purify peptide-MHC complexes in MS EL experiments are mostly pan- or locus-specific, and the data generated in an MS experiment are thus inherently polyspecific - i.e. they contain ligands matching multiple binding motifs. For instance, in the context of the human immune system, each cell can express up to six different MHC class I molecules, and the immunopeptidome obtained using MS techniques will thus be a mixture of all ligands presented by these MHCs [40]. The poly-specific nature of MS EL libraries constitutes a challenge in terms of data analysis and interpretation, where, to learn specific MHC rules for antigen presentation, one must first associate each ligand to its presenting MHC molecule(s) within the haplotype of the cell line.

Several approaches have been suggested to address this task, including experimental setups that employ cell lines expressing only one specific MHC molecule [170, 217–219], and approaches inferring MHC associations using prior knowledge of MHC specificities [179] or by means of unsupervised sequence clustering [169]. For instance, GibbsCluster [165, 166] has been successfully employed in multiple studies to extract binding motifs from EL data sets of several species, both for MHC class I and MHC class II [175, 184, 185, 187]. A similar tool, MixMHCp [169] has been applied to the deconvolution of MHC class I EL data with performance comparable to GibbsCluster. However, neither of these methods can fully deconvolute the complete number of MHC specificities present in each data set, especially for cell lines containing overlapping binding motifs and/or lowly expressed molecules (as in the case of HLA-C). Moreover, for both methods the association of each of the clustered solutions to a specific HLA molecule must be guided by prior knowledge of the MHC binding motifs, for instance by recurring to MHC-peptide binding predictions [167]. Therefore, both methods require some degree of manual intervention for deconvolution and allele annotation.

A recently published method was suggested to overcome this limitation. The computational framework by Bassani-Sternberg et al. [188] employs MixMHCp [169] to generate peptide clusters and binding motifs for a panel of poly-specificity MS data sets, and next links each cluster to an HLA molecule based on allele co-occurrence and exclusion principles. Although this approach constitutes a substantial step forward for aiding the interpretation of MS EL data, it has some substantial shortcomings. First and foremost, the success of the method is tied to the ability of MixMHCp to identify all the binding motifs in a given MS data set, an ability that is challenged in particular for cell lines containing MHC alleles with similar binding motifs, and for molecules characterized by low expression levels [169, 220]. Secondly, successful HLA labeling of the obtained clusters is limited by allele co-occurrences and exclusions across multiple cell line data sets. Although one may argue that this shortcoming is destined to wane as more immunopeptidomics data sets are accumulated in public databases, there currently remain multiple cases when co-occurrence and exclusion principles fail to completely deconvolute peptidome specificities [188].

Inspired by the framework outlined by Bassani-Sternberg et al. [188] and by the earlier success of the pan-specific NNAlign framework for modeling peptide-MHC binding [162], we here present a novel machine learning algorithm resolving these shortcomings, enabling a fully automated clustering and labeling of MS EL data. The algorithm is an extension of the NNAlign neural network framework [158, 159, 192], and is capable of taking a mixed training set composed of single-allele (SA) data (peptides assigned to single MHCs) and multi-allele (MA) data (peptides with multiple options for MHCs assignments) as input and deconvolute the individual MHC restriction of all MA peptides while learning the binding specificities of all the MHCs present in the training set. Compared with earlier approaches for peptidome deconvolution, annotation, and prediction model training (e.g. GibbsCluster NNAlign [220] and MixMHCp MixMHCpred [188]), NNAlign_MA performs these three tasks simultaneously, by iteratively updating the clustering, MHC annotation and peptide binding predictions in an integrated framework. NNAlign_MA does not require manual curation to assign the correct number of clusters, nor for the annotation of clusters to their

respective MHC molecule. NNAlign_MA is available at: `www.cbs.dtu.dk/suppl/immunology/NNAlign_MA/NNAlign_MA_testsuite.tar.gz`.

Materials and Methods

Peptide Data

Several types of MHC peptide data for human (HLA) and bovine (BoLA) class I, and HLA class II were gathered to train the predictive models presented in this work. Peptide data was classified as single allele data (SA, where each peptide is associated to a single MHC restriction) and multi allele data (MA, where each peptide has multiple options for MHC restriction). MA data are generated from MS MHC ligand elution assays where most often a pan-specific antibody is applied for class I and either a pan-specific class II or a pan-DR specific antibody is applied for class II in the immuno-precipitation step leading to data sets with poly-specificities matching the MHC molecules expressed in the cell line under study. SA data were obtained from binding affinity assays, or from mass spectrometry experiments performed using genetically engineered cell lines that artificially express one single allele.

HLA class I: SA data -both binding affinity (BA), and MS MHC eluted ligands (EL)- was extracted from Jurtz et al. [162]. The MA data was collected from eight different sources [40, 175, 184, 188, 221–224]. Both data sets were filtered to include only peptides of length 8-14 amino acids. Additional information concerning the HLA class I MA data can be found in Supplementary Table 3.S3 and information concerning the SA BA and EL data sets in Supplementary Table 3.S4.

HLA class II: BA data was extracted from the NetMHCIIpan-3.2 publication [163]. As for EL data, the Immune Epitope Database [225] (IEDB) was queried to identify publications with a large number of allele annotated EL data, both SA and MA [185, 202, 226–233]. Ligands were extracted from these publications, excluding any ligands with post translational modifications. Both BA and EL data was length filtered to include only peptides of length 13-21. Details on the composition of the HLA class II MA data are shown in Supplementary Table 3.S5.

BoLA: SA data was extracted from Nielsen et al. [167] and the MA data was collected either from the same publication (data for the MHC homozygous cell lines expressing the haplotypes A10, A14, A18) or were generated for this study (data for the cell lines expressing the haplotypes A11/A11, A19/A19, A20/A20, A15/A15, and A12/ A15). All data sets were filtered to include only peptides of length 8-14. A summary of the BoLA MA data is given in Supplementary Table 3.S6.

BoLA EL Data Generated for This Study

BoLA Cell Lines and BoLA-I-Associated Peptide Purification

Associated Peptide Purification were performed according to the procedures described in [167].

LC-MS2 Analysis

Samples were suspended in 20 μl of loading buffer (1% acetonitrile, 0.1% TFA in water) and analyzed on an Ultimate 3000 nano UPLC system online coupled to a Fusion Lumos mass spectrometer (Thermo Scientific). Peptides were separated on a 75 μm $\times$ 50 cm PepMap C18 column using a 1h linear gradient from 5 to 25% buffer B in buffer A at a flow rate of 250 nL/min (600 bar). Peptides were introduced into the mass spectrometer using a nano Easy Spray source (Thermo Scientific) at 2000V and ion transfer tube temperature of 305 °C. Subsequent isolation and higher energy C-trap dissociation (HCD) was induced on the most abundant ions per full MS scan at 2 s cycle time. Ions with a charge of 2-4 were measured at an accumulation time of 120 ms, AGC target of 200,000, quadrupole isolation width of 1.2 Da, and energy level 28. All fragmented precursor ions were actively excluded from repeated selection for 60 s.

MS Data Analysis

MS data was searched against a database comprising the 23/12/2017 download Uniprot entries for organism bos bovis (32,207 entries) concatenated with 4,084 Theileria muguga protein sequences annotated from RNAseq data of the schizont stage of T. parva (GenBank, BioSample accession SAMN03981746) plus a single entry for beta-galactosidase of E. coli. 36,692 entries were searched simultaneously in Peaks v8.5. No specificity was set for enzymatic digestion and no modifications

	SA (BA)			SA (EL)			MA (EL)		
	Pos	Neg	# MHCs	Pos	Negs	# MHCs	Pos	Neg	# MHCs
HLA-I	50,344	127,169	104	46,183	740,939	51	225,751	5,399,788	67
BoLA-I	50,361	150,833	7	84,717	1,644,976	-	92,339	1,788,293	16
HLA-II	55,178	76,185	59	32,51	337,72	8	15,494	152,445	16

Table 3.1. Training data overview. For each MHC system (first column), the number of positive and negative instances is shown for each type of training data. SA: Single Allele; MA: Multi Allele; BA: Binding Affinity; EL: Eluted Ligands.

of amino acids allowed. Mass tolerance for precursor ions was 5 ppm, whereas fragment mass tolerance was set to 0.03 Da. Score threshold was set corresponding to a false discovery rate of 1.0% as determined by simultaneous decoy database searches integrated in the Peaks 8.5 software. The full list of MS identified peptides generated for this work can be found in Supplementary Table 3.S10.

In Vitro Binding Data

Recombinnt BoLA-1*00901 and human beta-2 microglobulins (β2m) were produced as previously described [234]. In brief, biotinylated BoLA-1*00901 was generated in Escherichia coli, harvested as inclusion bodies, extracted into Tris-buffered 8 M urea and purified using ion exchange, hydrophobic, and gel filtration chromatographies. MHC-I heavy chain proteins were never exposed to reducing conditions, which allows for purification of highly active pre-oxidized BoLA molecules, which folds efficiently when diluted into an appropriate reaction buffer. The pre-oxidized, denatured proteins were stored at -20 °C in Tris-buffered 8 M urea. Human β2m was expressed and purified as previously described [235].

Nonameric peptide binding motifs were determined for BoLA1*00901, using PSCPL as previously described [197, 234, 236]. Recombinant, biotinylated BoLA heavy chain molecules in 8 M urea were diluted at least 100-fold into PBS buffer containing 125I-labeled human β2m and peptide to initiate pMHC-I complex formation. The final concentration of BoLA was between 10 and 100 nM, depending on the specific activity of the heavy chain. The reactions were carried out in the wells of streptavidin-coated scintillation 384-well FlashPlate® PLUS microplates (Perkin Elmer, Waltham, MA). Recombinant radiolabeled human β2m and saturating concentrations (10 μM) of peptide were allowed to reach steady state by overnight incubation at 18 °C. After overnight incubation, excess unlabeled bovine β2m was added to a final concentration of 1 μM and the temperature was raised to 37 °C to initiate dissociation. pMHC-I dissociation was monitored for 24 h by consecutive measurement of the scintillation microplate on a scintillation TopCount NXT multiplate counter (Perkin Elmer, Waltham, MA). PSCPL dissociation data were analyzed as described [222]. Briefly, following background correction, the area under the dissociation curve (AUC) was calculated for each sublibrary by summing the counts from 0 to 24 h. The relative contribution of each residue in each position (i.e. the relative binding, RB) was calculated as RB = (AUC_sublibrary/AUC_X9). The RB values were next normalized to sum to one for each peptide position and used as input to Seq2Logo to generate the in vitro BoLA-1*00901 binding-motif.

Training Data

Three training sets were constructed, one for each of the systems under study (Table 3.1). To ensure an unbiased performance evaluation on the MA data, duplicated entries between the SA EL and MA data were first removed from the SA EL data set for each training set. Next, random peptides were extracted from the UniProt database and used as negative instances for the EL data in each case. Here, an equal amount of random negatives was used for each length, consisting of five times the amount of peptides for the most abundant length in the given positive EL data set as described earlier [167, 215]. This enrichment with random natural negative peptides was done for each individual SA and MA EL data set. The amount of positive and negative peptides in each training set is shown in Table 3.1.

Evaluation Data

For HLA class I, an independent evaluation data set of HLA restricted CD8+ epitopes was obtained from Jurtz et al. [162]. After removal of epitopes overlapping with the HLA-I training data, the final evaluation data consisted of 558 HLA-epitope entries. For the evaluation with MixMHCpred, MHCFlurry, and MHCFlurry_EL (the version of MHCFLurry trained including EL data), the epitope data set was further filtered to only include epitopes restricted to HLA molecules covered by all method. This resulted in a data set of 541 epitopes. Because MixMHCpred cannot make predictions for peptides containing X, all such peptides were removed from the benchmark before

evaluation.

For BoLA-I, a set of BoLA restricted epitopes were obtained from Nielsen et al. [167]. For HLA-I and BoLA-I evaluation, the source protein sequence of each epitope was in-silico digested into overlapping 8-14mers, and the performance reported as the Frank score, i.e. proportion of peptides with a prediction score higher than that of the epitope [162]. Using this measure, a value of 0 corresponds to a perfect prediction (the known epitope is identified with the highest predicted binding value among all peptides found within the source protein) and a value of 0.5 to random prediction

For HLA class II all CD4+ epitopes measured by Intracellular Cytokine Staining(ICS) assay were downloaded from the IEDB. The set was filtered to include only positive epitopes with four letter resolution HLA typing. Further, epitopes overlapping with the HLA-II training data (100% identity) were removed. As for HLA-I, the Frank score was used to validate the model performance, here in-silico digesting the source protein into overlapping of a length equal to that of the epitope. Finally, to exclude potential noise, epitopes were discarded if none of the prediction methods included in the benchmark could identify the epitope with a Frank value of 0.2 or less. This resulted in a set of 221 HLA-II epitopes for evaluation.

NNAlign_MA Modeling and Training Hyperparameters

Models for peptide-MHC binding prediction were trained with hyperparameters and model architectures similar to those described earlier [159, 215, 220] for prediction of peptide-MHC binding based on the data sets described in Table 3.1. Positive instances in EL data sets (for both SA and MA) were labeled with a target value of 1, and negatives with a target value of 0. To avoid performance overestimation and model overfitting, training sets were split into 5 partitions for cross-validation purposes using the common motif algorithm [156] with a motif length of 8 amino acids for class I (corresponding to the shortest binding mode for class I peptides) and 9 amino acids for class II (corresponding to the size of the class II binding core) as described earlier [162, 215].

A single and simple yet highly critical step sets the updated NNAlign_MA method proposed here aside from its ancestors. To be able to accurately handle and annotate MA data, NNAlign_MA imposes a burn-in period where the method is trained only on SA data. After the burn-in period, each data point in the MA data set is annotated by predicting binding to all possible MHC molecules defined in the MA data set and assigning the restriction from the highest prediction value (see the Prediction score rescaling section for variations on this). After this annotation step, the SA and MA data are merged respecting the data partitioning to further train the algorithm. This MA annotation step is repeated in each training cycle.

In the case of HLA-I and BoLA-I, models were trained on the full set of SA and MA data as an ensemble of 50 individual networks, generated from 5 different seeds; 56 and 66 hidden neurons; and 5 partitions for cross-validation. Models were trained for 200 iterations (using early stopping), with a burn-in period of 20 iterations. For performance comparison, a SA-only model was trained for HLA-I using the same architecture and hyper-parameters by excluding all MA data from the cross-validation partitions, thus including only data for SA while respecting the training data structure.

Regarding HLA-II, default settings for MHC-II prediction as previously described [215,220] were used. Models were trained and evaluated on 5-fold cross validation partitions defined by common motif clustering with a motif of length 9. The final ensemble of models consists of 250 networks (2, 10, 20, 40 and 60 hidden neurons and 10 random weight initiation seeds for each CV fold). Networks were trained for 400 iterations, without early stopping and using a burn-in period of 20.

All networks have an input layer, a single hidden layer and an output layer with two output values (one for binding affinity and one for eluted ligand likelihood). Networks were trained using back-propagation with stochastic gradient descent and a fixed learning rate of 0.05. When making predictions using the network ensembles, the average over the individual network predictions was used.

Prediction Score Rescaling

To level out differences in the prediction scores between MHC alleles imposed by the differences in number of positive training examples and distance to the training data included in the SA data set, a rescaling of the raw prediction values was implemented and applied in the MA data annotation. The rescaling was implemented as a z-score transformation of the raw prediction values using the relation $z = (p - \bar{p})/\sigma$, where p is the raw prediction value of the peptide to a given MHC molecules, and $\bar{p}$ and σ are the mean and standard deviation of the distribution of prediction values for random

natural peptides for the MHC molecule. Here, the score distribution was estimated by predicting binding of 10,000 random natural 9mer peptides to MHC molecule in question. Next, the mean and standard deviation were estimated from a positive normal distribution, iteratively excluding outliers (z-score < -3 or z-score > 3). For an example on how z-score is applied to transform prediction score distributions, see Supplementary Figure 3.S14. This estimation of $\bar{p}$ and σ was repeated in each iteration round before annotating the MA data. As the rescaling is imposed to level out score differences between MHC molecules characterized in the SA training binding data and molecules from the MA data distant to the training data, the need for rescaling should be leveled out as the MA data are included in the training and the NNAlign_MA training progresses. To achieve this, the values of $\bar{p}$ and σ were modified to converge toward uniform values $\bar{p}_u$ and σ_u defined as the average of $\bar{p}$ and σ over all molecules in the MA data set. This convergence was defined as $p' = w \cdot \bar{p} + (1-w) \cdot \bar{p}_u$ and $\sigma' = w \cdot \sigma + (1-w) \cdot \sigma_u$, where $w = 1/(1+e^{(x-75)/10})$ and x is the number of training iterations. With this relation, when w is close to 1 after pre-training ($x = 20$), the terms $\bar{p}_u$ and σ_u vanish; on the other hand, as x passes 100 iterations, w converges to 0 and the terms $\bar{p}$ and σ will vanish. With this, one can modulate the rescaling of the data as a function of the iterations and the type of data being used for training (SA or MA). The shift value of the exponential present in w (75) is a tunable parameter that defines this adjustment schedule. Similar results as the ones shown in this work were obtained varying this value in the range 50 –100 (data not shown).

Distance Between Pairs of MHC Molecules
The distance between MHC molecules was calculated as described earlier [164] from the sequence similarity between the pseudo sequences of the two molecules. Likewise, was the distance of an MHC molecule to the data used to train a given prediction model, defined as the closest distance to any MHC molecule included in the training data.

Pruning the HLA Supertype Tree - HLA Models with Removed Specificities
To quantify how MA data can boost the performance of a peptide-MHC predictor, we constructed additional models, where SA data associated with HLA molecules from the A2 and A3 supertypes where exclude from the training data. In short, this was achieved by first identifying the alleles in the MA data for the two supertypes [237], resulting in the following allele list: HLA-A*02:01, HLA-A*02:05, HLA-A*02:06, HLA-A*02:20, HLA-A*68:02, HLA-A*03:01, HLA-A*11:01, HLA-A*30:01, HLA-A*31:01, and HLA-A*68:01. Next, all data for alleles with a distance (see above) of less than 0.2 to any of the alleles in this list were removed from the SA data. Finally, a SA model was trained as described above on the remaining SA data, and MA model on the remaining SA data combined with the complete MA data, respecting the original data partitioning.

Sequence Motif Construction
Sequence binding motif were visualized as Kullback-Leibler logo plots using Seg2Logo [29]. Amino acids are grouped by negatively charged (red), positively charged (blue), polar (green) or hydrophobic (black). If not otherwise specified, binding motifs were generated from the top 0.1% of 200,000 random natural peptides (9mers for class I and 15mers for class II) as described earlier [162].

Binding Motifs Similarity Comparison
The similarity between two HLA binding motifs was estimated in terms of the Pearson's correlation coefficient (PCC) between the two vectors of 9*20 elements (9 positions and 20 amino acid propensity scores at each position).

Model Performance Evaluation
For model comparison, the AUC (Area Under the ROC Curve) and AUC 0.1 (Area Under the ROC Curve integrated up to a False Positive Rate of 10%) performance measures were used. For a given model, each test set was predicted using the model trained during cross-validation. Next, all test sets were concatenated, and an AUC/AUC0.1 value was calculated for each MHC molecule/cell line identifier. In case of multi-allele data, the prediction score to each peptide was assigned as the maximal prediction value over the set of possible MHC molecules.

To evaluate the "cleanness" of a given cluster/motif identified by NNAlign_MA, positive-predictive values (PPV) were calculated. For each cell line, we calculated the number of ligands N predicted to be bound to each allele from the concatenated test set predictions. Next, the PPV for each motif was calculated as the fraction of peptides in the top N*0.95 predictions that were actual ligands. The values of 95% was selected to tolerate a certain proportion of noise in the EL data [220].

Results

A key issue associated with the interpretation and analysis of LC-MS MHC eluted ligand data sets (EL data) stems from the challenge of deconvoluting and linking each ligand back to the presenting

MHC molecule(s) of the investigated cell lines. In the following, we describe the NNAlign_MA framework resolving this challenge, and showcase how the framework can be applied to effectively integrate MA EL data in a semi-supervised manner into machine-learning models for improved prediction of MHC antigen presentation and T-cell epitopes on the three large data sets of human (class I and class II) and cattle MHC class I ligand and T-cell epitope data.

The NNAlign_MA Algorithm

The NNAlign_MA algorithm is an extension of the NNAlign neural network framework, and is capable of taking a mixed training set composed of singleallele data (SA, peptides assigned to single MHCs) and multiallele data (MA, peptides that are assigned to multiple MHCs), and fully deconvolute the individual MHC restriction of all MA peptides, learning the binding specificities of the MHCs present in the training set. In short, the NNAlign framework underlying NetMHCpan-4.0 method, is an artificial neural network method integrating SA binding affinity and EL data with sequence information of the MHC molecules, allowing information to the leveraged both between data types and MHC data sets, resulting in pan-specific predictive power [162].

The MA extension of NNAlign consists of various critical steps (see Figure 3.2 for a schematic overview). First, a neural network is pre-trained on SA data only during a burn-in period, using the NNAlign framework. This results in a panspecific model with potential to infer binding specificities also for MHC molecules not included in the SA data set [162, 164]. After this initial training period (from now on referred to as "pre-training"), the data in the MA data sets are annotated. That is, binding for each positive peptide in the MA data set is predicted (using the ligand likelihood prediction value from the pre-trained model) to all the possible MHC molecules of the given cell line and the restriction is inferred from the highest prediction value (for details see Materials and Methods). For negative MA data, a random MHC molecule from the given cell line is tagged. Next, the SA and now single-MHC annotated MA data are merged, and the model is retrained on the combined data. Note, that the MHC allele annotation is updated at each iteration and will in general change as the training progresses. Implicitly, the algorithm exploits the principles of co-occurrence and exclusions outlined by BassaniSternberg et al. [188] : i.e. sequence motifs that consistently occur across multiple cell lines sharing only specific MHC alleles are assigned to the shared MHC(s) by the iterative annotation step. For an illustration of this effect refer for instance to HLA-B*13:02, the only allele in common between the two cell lines CM467 and pat-NS2. The binding motif for this molecule (and the other HLA molecules in each cell line) as obtained by NNAlign_MA are shown in Supplementary Figure 3.S9. Here, it is apparent that only one motif is shared by these two cell lines, and the co-occurrence principle allows NNAlign_MA to assign this motif to HLA-B*13:02. In ambiguous cases where co-occurrence and exclusion principles are insufficient, the pan-specific nature of the method will help tilt the annotation toward the correct MHC. An example of this is showcased by the motif of BoLA-1:00901 in Figure 3.2. BoLA-1:00901 is only present in the MA data, and hence are data for this molecule only presented to the model after the pre-training. Because no MHC molecule in the SA data share a strong preference for H at P9, H at P9 is absent in the predicted motif for BoLA-1:00901 after pre-training. After pre-training, the data from the two cell lines (A12/A15 and A15/A15) expressing BoLA-1:00901 are presented to the model, hence for the first-time showing peptides with a H at P9. NNAlign_MA is now faced with the challenge of assigning these peptides to one or more of the alleles expressed in the two cell lines. The method does this in two ways, by within each cell line predicting for each peptide binding to all expressed alleles and selecting the most favorable as the potential restriction element, and by transferring knowledge of binding preferences for the different alleles imposed by ligands in other cell line data sets. Analyzing the binding of peptides using the pre-training model, BoLA-1:00901 (in contrast to BoLA-4:02401 and BoLA-2:02501) is found to tolerate H at P9 (the H is not present in the motif because this motif only displays amino acid enrichments). Further, both BoLA-4:02401 and BoLA-2:02501 are expressed in a third cell line (A14), but none of the motifs identified here share a preference for H at P9 (see Supplementary Figure 3.S12). Taken together, these properties enable the model to assign peptides with H at P9 to BoLA-1:00901. For details on model hyper-parameters and model training setup, see Materials and Methods.

HLA-I Benchmark

To benchmark NNAlign_MA, we trained a model on the complete set of SA data described in the NetMHCpan-4.0 paper, combined with an extensive set of MA data covering 50 different cell lines with typed HLA allotypes described in Bassani-Sternberg et al. [188]. Note that, before training, we removed all overlapping peptides between the SA and MA data from the SA data set. This was done to fully demonstrate the power of the NNAlign_MA to annotate MA data also in situations where the information cannot simply be transferred from the SA data. After training, each MA data point ends up being annotated to a single MHC molecule, and from this annotation the distribution

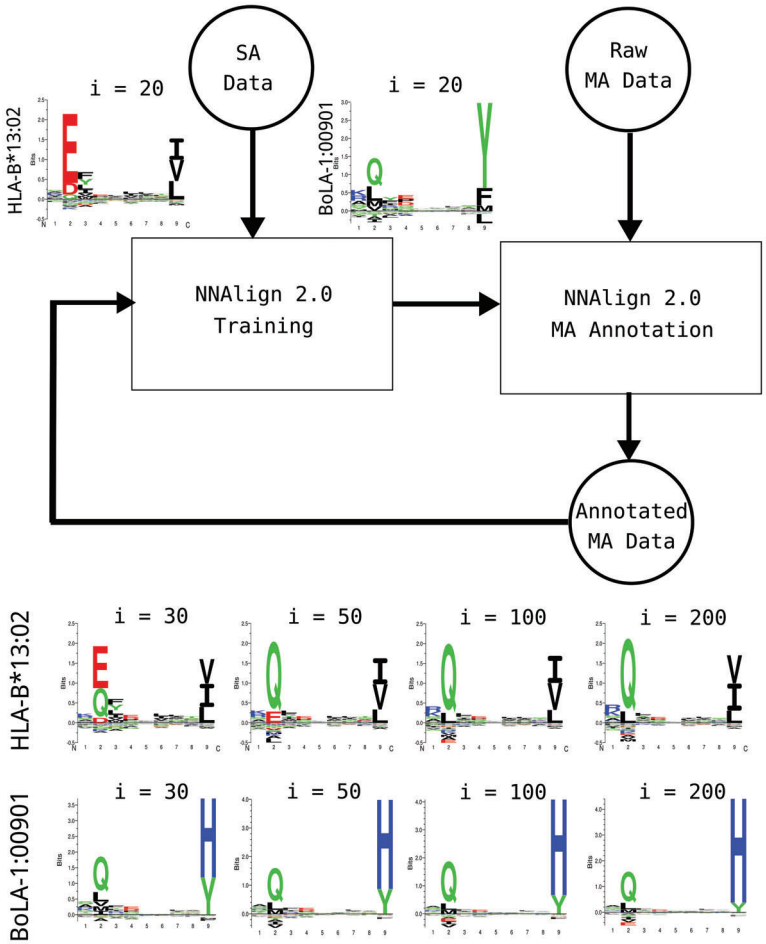

Figure 3.2. Full NNAlign_MA framework. Initially, a model is pre-trained using SA data only ("NNAlign 2.0 Training" box); next, MA data are annotated and merged with the SA data ("NNAlign 2.0 MA Annotation" box), generating newly annotated MA data; then, the training is repeated iteratively using such new data. The NNAlign_MA algorithm encompasses all the steps indicated in the flow chart. It is important to notice that, for alleles that are part of one or more MA data sets, a prediction score rescaling is applied in every epoch (iteration) after the pre-training, as a part of the MA annotation step. In the upper left part are displayed examples of binding motifs of two MHC molecules just after pre-training (i = 20). In the lower part of the figure are shown the changes to predicted binding motifs of the same two MHC molecules as NNAlign_MA iterates over the data. Here "i" refers to the number of iterations.

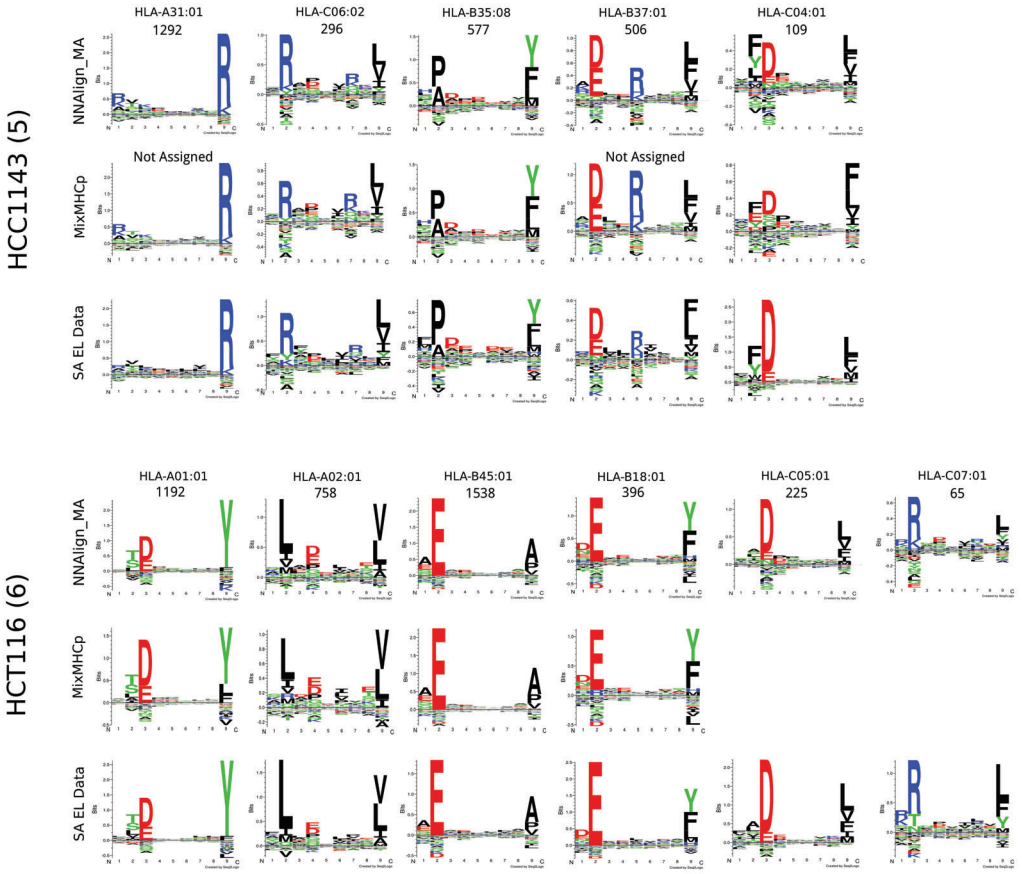

Figure 3.3. Sequence clustering and labeling comparison between NNAlign_MA, MixMHCp and NetMHCpan-4.0 for the HCC1143 and HCT116 cell lines. Motifs corresponding to NNAlign_MA were constructed based on ligands from the given MA data set (cell line) predicted using cross-validation to be restricted by the given HLA molecule; the quantity of peptides associated to each HLA molecule is given on top of the corresponding logos; allele annotation was performed automatically by NNAlign_MA. MixMHCp motifs were constructed by running the algorithm on the ligands associated to each cell line; allele annotation was obtained from [188]. SA EL data motifs were derived from single-allele (SA) data available from the IEDB [225]

of ligands associated with each HLA molecules was recovered. Based on the predicted 9mer binding cores of the ligands, logos for all the HLA alleles expressed by each of the cell lines under study was constructed (Supplementary Figure 3.S9). In this benchmark, NNAlign_MA was capable of not only clustering the EL data into a set of groups matching the number of expressed HLA alleles in each cell line (this is guaranteed by the construction of the method), but also to assign each group to a single corresponding HLA allele. As a point of comparison, on the same benchmark data, MixMHCp was only capable of achieving a complete deconvolution of all HLA specificities in 26% of the 50 cell line data sets (failing to identify motif corresponding to HLA-C alleles in 61% of the cases), and could not annotate at least one cluster in 16% of the samples. Two examples of this are given in Figure 3.3, showing the NNAlign_MA deconvolution of the two cell lines HCC1143 and HCT116. In the first case, MixMHCp correctly identified 5 motifs, but could not assign two of the five to their corresponding allele (respectively HLA-A*31:01 and HLA-B*37:01). For HCT116, MixMHCp was able to deconvolute and assign only four of the six expressed alleles, missing the deconvolution of the motifs for two HLA-C alleles, HLA-C*05:01 and HLA-C*07:01. The accuracy of the 4 motifs additionally identified by NNAlign_MA was confirmed by reference to SA data available from IEDB [225] (see Figure 3.3).

As stated above, the NNAlign_MA method by construction is guaranteed to cluster the MA data into several groups matching the number of HLA alleles expressed in each cell line. The association of each cluster to the correct HLA molecule, and the accuracy of each cluster are, however, not guaranteed. By investigating the deconvolution solutions for the different cell lines (Supplementary Figure 3.S9), it is apparent that the accuracy of the motifs identified by NNAlign_MA (as expected) depends on the number of ligands assigned to a given HLA, and that complete charac-

terization of the HLA's in a given cell line, for a few cases, is impeded by this fact (a few examples include the motif for HLA-C*07:04 from Fibroblast, HLA-C*08:01 from Mel-624, and HLA-C*02:10 from RPMI8226). These are further examples of alleles only present in single MA data sets, limiting the ability of NNAlign_MA to transfer information of the binding motifs from other data sets.

To further quantify the accuracy of the cluster-HLA association, we compared the motifs obtained by NNAlign_MA to the motifs obtained from SA data in situations where such data were available from the IEDB (Supplementary Figure 3.S10). Here, we in the vast majority of cases observed an excellent agreement, with an average correlation between the two motifs of 0.883 over the 46 alleles included (p value for the correlation being random was in each case $p < 0.001$, exact permutation test, for details on how the correlation was calculated refer to Materials and Methods. Note, that this correlation was equally high for alleles characterized by SA EL training data and alleles not characterized by SA EL data (average PCCs of 0.883 and 0.876, respectively). As expected, the agreement between the MA and SA motifs also here was highest for the cases where both motifs were characterized by large data sets.

Next, we compared the motifs of individual HLA alleles obtained across different cell lines, for example the HLAC*03:03 allele, shared between 5 cell lines. Using again a simple correlation analysis, we quantified the similarity of these different motifs, and could in all cases confirm a high consistency, with an overall averaged correlation of 0.901 over the 17 alleles shared by 5 of more cell lines (Supplementary Figure 3.S11). These correlation values were all significantly different from random ($p < 0.0001$, exact permutation test), and significantly higher than the correlations obtained by comparing motifs assigned to different HLA molecules ($p < 10^{-5}$), t test). Also, the correlations were lowest for the comparison between motifs characterized by small data sets (as exemplified by the motifs for HLA-C*07:02 from the HCC1937 and Mel-8 cell lines, each characterized by 48 and 31 ligand data points respectively (see Supplementary Figure 3.S9), resulting in a correlation between the two motifs of 0.68). Finally, we evaluated the "cleanness" of each cluster/motif by calculating predicted positive (PPV) values. Here, all clustering solutions were found to have very high accuracy, with an average PPV value of 75% (for details on the calculation of the PPV refer to Materials and Methods, and for the complete list of PPV values refer to Supplementary Table 3.S7.

Taken together, these results demonstrate the high performance of NNAlign_MA, achieving in most cases an accurate, consistent and complete (including for HLA-C) deconvolution of MA EL data sets.

Given these encouraging results, we next conducted a fullscale performance evaluation for prediction of eluted HLA ligands. To this end, we first compared the performance of NNAlign_MA trained on the complete HLA-I data set (referred to as the MA model) to the performance when trained only on the subset of SA data (referred to as the SA model). Note, that this SA model is trained identically to NetMHCpan-4.0, with the only exception of the removal of overlapping peptides between the SA and MA described above. This benchmark (Figure 3.4-A) demonstrated that the MA model exhibited a consistently (and statistically significant, $p < 0.0001$, paired t test) higher performance when evaluated on the MA data (median AUC of 0.9769), compared with the SA model (median AUC of 0.9712). On the other hand, as expected, the MA and SA models showed an overall comparable predictive performance when evaluated on the SA data (median AUC of 0.9842 versus 0.9839). We hypothesized that NNAlign_MA would demonstrate a performance gain over NNAlign_SA for alleles where the SA data are either limited or absent. The results displayed in Figure 3.4 confirmed this. Here, the median number of positives for SA data sets where NNAlign_MA outperforms NNAlign_SA was 57 whereas the number for the SA data sets where NNAlign_SA won was 435. Further, was the performance gain of NNAlign_MA on the MA data found to be largest for data sets characterized by alleles absent from the SA data. Last, we investigated if the MA model also demonstrated improved performance compared with the SA model for binding affinity (BA) predictions. Here, we evaluated the performance of the two models in terms of the allele specific PCC on the BA data using cross validation. Also, here did the MA significantly outperform the SA model with median PCC values of 0.766 and 0.759 ($p < 0.005$, binominal test).

Next, we investigated how the peptidome for the MA EL data in each data set was distributed among the alleles of the three loci HLA-A, HLA-B and HLA-C. To do this, we extracted the number of ligands predicted by NNAlign_MA to be restricted by HLA-A, HLA-B and HLA-C for each cell line present in the MA EL data set and then calculated the proportion of ligands associated to a given loci relative to the total amount of peptides in the cell line. The result of this analysis is shown in Figure 3.4-B and confirms the general notion that HLA-A and HLA-B have comparable peptidome repertoire size, whereas the peptidome size of HLA-C, in comparison, is substantially reduced [197]. Although this is not a novel observation, to the best of our knowledge this is the

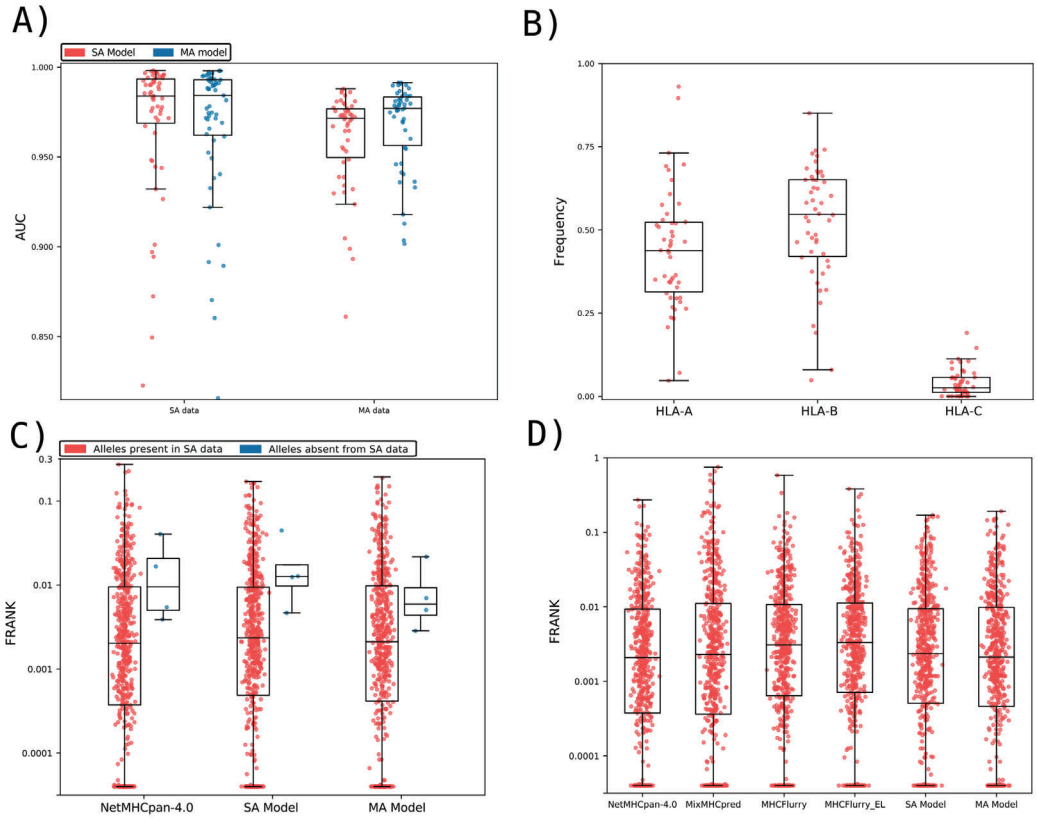

Figure 3.4. Benchmark of the prediction method on the HLA-I data. A, Performance of the SA and MA trained models on the SA EL and MA EL data sets, expressed in terms of AUC. Each point corresponds to one SA or MA data set. SA data corresponds to 55 single-allele EL data set. MA data consist of EL data from 50 different cell lines, each expressing more than one MHC molecule. To evaluate the MA data, each data point was assigned the highest prediction value across all possible MHC restrictions in the given cell line data set (for further details, see Materials and Methods). Performance values for the SA model on the SA data sets, and for the MA model on the SA and MA data, were extracted from the cross-validated predictive performance. B, The relative peptidome size for the three loci (HLA-A, HLA-B and HLA-C) as predicted by NNAlign_MA for the different MA data sets. Each data point gives the relative proportion of ligands in each MA data set predicted to be restricted by a HLA from the given locus. The HLA restriction for each ligand data was estimated from the evaluation performance of the cross-validation as described in Material and Methods. Only data sets where HLA expression is annotated for all three loci were included. C, Frank values for the epitope evaluation data set for NetMHCpan-4.0 and models trained with only SA data (SA Model) and with SA and MA data (MA Model). Red dots correspond to epitopes restricted to HLAs that were part of the SA training data, whereas blue points refer to Frank values for epitopes with HLA restrictions absent from the SA training set. For visualization, Frank values of 0 are displayed with a value of 0.00004. HLAs in the category "Alleles absent from SA data" are HLA-B*13:02, HLA-B*55:01 and HLA-C*01:02. D) Frank values for the epitope evaluation on NetMHCpan-4.0, MixMHCpred, MHCFlurry (trained on BA data), MHCFlurry_EL (trained on BA and EL data) and the models trained with only SA data (SA Model) and MA data (MA Model). For visualization, Frank values of 0 are displayed with a value of 0.00004.

first fully automated analysis of EL data demonstrating this. Some clear outliers are present in the figure where either the HLA-A or HLA-B peptidome repertoires are highly reduced compared with the median values. A few such examples include the HL-60, CA46, Mel-624 and HEK293 cell lines where either the entire HLA-A or HLA-B locus appears to have been deleted or made non-functional. These observations agree with results from earlier studies for these cell lines (for details refer to Supplementary Table 3.S8), suggesting the power of NNAlign_MA also for identification of loss or down-regulation of HLA expression directly from EL data sets.

Evaluation on HLA-I Epitopes from IEDB
To further investigate the predictive power of NNAlign_MA, we employed an evaluation set of epitopes of length 8 –14 extracted from IEDB (see Materials and Methods for details). Here, we divided the data set into two subsets; one containing the epitopes restricted to HLAs that were part of the SA training data set, and one where the HLAs were not present in the SA training data. The results of the evaluation on these two data sets for the SA and MA models, and the state-of-the-art method NetMHCpan-4.0 are shown in Figure 3.4-C in terms of Frank values. This measure reflects the false-positive rate, and a value of 0 corresponds to the perfect prediction (for details see Materials and Methods). For epitopes restricted to HLA molecules that are part of the SA data set (red points), the figure displays a comparable performance of the MA and NetMHCpan-4.0 methods (median Frank values 0.0021 and 0.0020, $p < 0.3$, paired t test), and a significantly worse performance of the SA method (median Frank 0.0022, $p < 0.0025$ paired t test)). Further, when evaluated on the small set of epitopes whose HLA restrictions are only present in the MA data (blue points), the performance of the MA model is substantially increased compared with both the SA and NetMHCpan-4.0 models (average Frank of 0.0091 compared with 0.0186 and 0.0166 for the SA and NetMHCpan models). These results demonstrate how the NNAlign_MA model also when it comes to prediction of T-cell epitopes achieved state-of-the-art performance, and further is capable of benefiting from MA data to expand the allelic coverage outside SA data set to improve the allelic coverage and predictive power.

Finally, in Figure 3.4-D, the evaluation was expanded to include the MHCFlurry (trained without and with EL data) and MixMHCpred methods limiting the benchmark to include only HLA molecules covered by all methods, thus including only HLA alleles with previously well-characterized binding motifs (for details on the benchmark refer to Materials and Methods). The results of this benchmark confirmed a comparable performance of NNAlign_MA (median Frank 0.0021) to NetMHCpan-4.0 (median Frank 0.0021), and a small (but statistically significant, $p < 0.05$ paired t test, in all cases except for MHCFlurry) drop in performance of MHCFlurry (median Frank 0.0031), MHCFlurry_EL (median Frank 0.0033), MixMHCpred (median Frank 0.0023) and NNAlign_SA (median Frank 0.0024). This result confirms the state-of-the-art performance of NNAlign_MA.

A Specificity Leave-out Benchmark

All the benchmarks performed hereto were conducted in situations where the MA data shared high HLA overlap with the SA data. By way of example, over 75% (51 of 67) of the alleles in the MA data set were part of the SA data, and 94% (63 of 67) share a distance of less than 0.1 to an allele in the SA data as measured from the similarity between pseudo sequences (for details on this similarity measure see Materials and Methods), a distance threshold earlier demonstrated to be associated with high predictive accuracy of the pan-specific prediction model [238].

As stated above and confirmed by the results in Figure 3.4-A and 3.4-C, the main power of NNAlign_MA is to effectively extend the allele-space covered by HLA annotated EL data leading to an improved predictive power outside the space covered by SA data. Given the high allelic overlap between the MA and SA data set, it is hence not surprising that the impact of including MA data in these benchmarks was limited. Therefore, to further test the power of the NNAlign_MA method in a more extreme setting, we conducted an experiment where parts of the SA data were left out from the training data leaving part of the HLA space covered only by MA data. In short, we removed all SA data for HLA molecules belonging to (or similar to alleles in) the HLA-A2 and HLA-A3 supertypes [237], effectively pruning off whole branches from the tree of HLA specificities (Figure 3.5 left panel) (for details on this pruning refer to Materials and Methods). This scenario thus simulates a situation where the MA data describes binding specificities that a novel compared with any specificity contained in SA training data. Given this, the binding motifs in the MA data cannot simply be inferred from a close neighbor in the SA data, making the challenge of HLA deconvolution non-trivial. This experiment therefore allows us to investigate how the NNAlign_MA method can benefit from MA data to accurately characterize the binding specificity of HLA molecule not characterized by SA data, and from such MA data expand the HLA coverage of the trained prediction model.

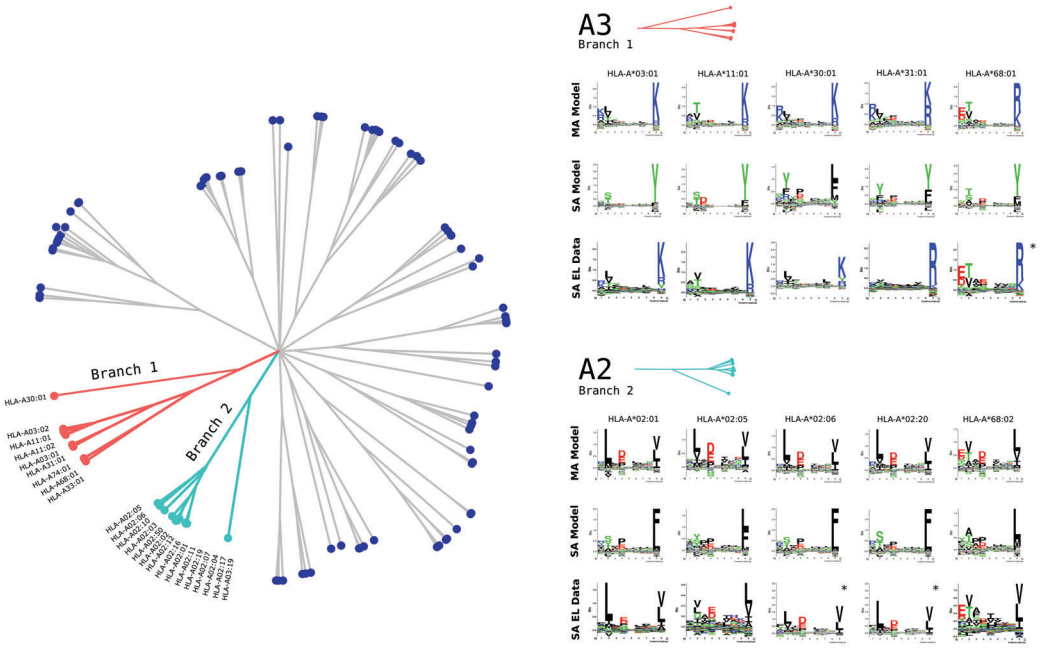

Figure 3.5. HLA supertype pruning experiment. The left panel shows a functional tree of HLA specificities estimated using MHCcluster [239]. Branches in light blue and red correspond to the A2 and A3 HLA supertypes. In the HLA supertype tree pruning experiment, SA data for HLA molecules belonging to both these branches were removed. Right panel shows binding motifs for A2 and A3 supertype alleles from the MA EL data set predicted by the MA and SA models. Motifs were constructed from the top 1% of 1,000,000 random natural 9mer peptides predicted by each model. SA EL data show motifs derived from SA EL data (not included in the training). For alleles marked with * no SA EL data was available and motifs were obtained from NetMHCpan from `http://www.cbs.dtu.dk/services/NetMHCpan/logos_ps.php`

In the benchmark, SA and MA models were trained as described above on the pruned SA and complete MA data, and the predictive performance was evaluated on SA EL data for the alleles on the pruned branch. Note, that this evaluation was done respecting the data partitioning of the cross validation to avoid introducing a bias in favor of the MA model. The performance was estimated in terms of AUC0.1 resulting in average values of 0.599 versus 0.852 for the SA and MA models respectively (for details on the performance values see Supplementary Table 3.S9).

Next, binding motifs for the alleles in the MA data from the A2 and A3 supertypes were estimated for the MA and SA models and compared with motifs derived from SA EL data if available (see Figure 3.5 right panel). Here, we observed a high overlap between the motifs of the MA model motifs obtained from SA EL data, and a likewise low overlap of the motifs obtained by the SA model. Note, also here that the agreement between the MA model and SA EL data motifs was dependent on the number of ligands assigned to the given allele from the MA data, i.e. HLA-A*02:01 was assigned 26,038 and HLAA*02:05 only 917 ligands from the MA data resulting in a somewhat lower agreement between the two motifs for HLAA*02:05 compared for HLA-A*02:01. Finally, we reinvestigated the performance of the SA and MA models trained on the pruned SA data set, on the subset of 358 epitopes restricted by the 11 alleles covered by the two A2 and A3 supertypes. This benchmark confirmed the superior performance of the MA model over the SA model with median Frank values of 0.0044 compared with 0.0393 ($p < 10^{-}15$, paired t test). Note, the performance of the full MA model on this epitope data set was 0.0032. Taken together these results demonstrate the power of the NNAlign_MA to accurately characterize individual binding motifs of molecules from MA data only.

BoLA-I Benchmark

Having demonstrated how NNAlign_MA was capable of benefitting from MA EL data to boost predictive power and expand the allelic coverage also in a setting where the MA data shared limited allelic overlap to the SA data, we next turned to the BoLA (Bovine Leukocyte Antigen) system. Because binding data (both BA and EL) is more scarce for BoLA compared with HLA, and because the relative expression of MHC molecules within a given cell line varies in a more dramatic

manner for the bovine system compared with humans, analyzing and deconvoluting BoLA MA EL data is more challenging compared with HLA, and working within this system allowed us to better appreciate and assess the strength and potential limitations of the NNAlign_MA framework.

In a previous work, a prediction model for BoLA peptide interactions, NetBoLApan, was trained using NNAlign on SA BA (including binding affinity data for 7 BoLA molecules) and EL MA data from 3 BoLA-I homozygous cell lines, describing the BoLA haplotypes A10, A14, and A18 [167]. Because of the prior limitation of the NNAlign framework only admitting SA data for training, in NetBoLApan the EL MA data had to be first deconvoluted using GibbsCluster and then manually annotated to the individual BoLA molecules of each cell line by visual inspection. Since this earlier publication, we have generated MA EL data for additional 5 cell lines. Using these data, we trained and evaluated the NNAlign_MA framework on the SA data described above combined with EL MA data for a total of 8 BoLA cell lines (for details on these data sets, refer to Materials and Methods and Supplementary Table 3.S6).

After training, we proceeded to investigate the different binding motifs captured by the model. We were interested in the motifs of the BoLA molecules shared between multiple haplotypes. One such example is BoLA-2*02501, present in the A14, and A15 haplotypes. Although the motif for this molecule in our earlier work showed a strong preference for G/Q and L at P2 and P9 respectively (Figure 3.6-B), the new NNAlign framework captured a completely different signal, with a consistent proline (P) signal on most positions (see Fig. 3.6-A). Also, the number of ligands assigned to BoLA-2*02501 was extremely low for the three cell lines expressing this molecule (less than 1.5% in all three cases). A prime source of this result is the very large distance (0.426) of BoLA-2*02501 to the SA training data (for details on this distance measure refer to section Distance Between Pairs of MHC Molecules above). In comparison, the maximum distance between any molecule in the MA data to the SA training data for the HLA system is less than 0.13. These large pairwise distances in the BoLA system have two strong impacts on the predictive behavior of NNAlign_MA. First and foremost, the model pretrained on the SA is expected to have limited power to predict the binding motif of the BoLA-2*02501 molecule (in the pretrained model, BoLA-2*02501 prefers P at P2). Secondly, the prediction values of the pre-trained model will be lower for this molecule compared with molecules that share higher similarity to the SA data used for pre-training. To deal with the latter of these two issues, we devised a rescaling scheme for the prediction score in the MA annotation step of the NNAlign_MA framework, and rescaled the raw prediction by comparing it to a score distribution obtained from a large set of random natural peptides (for details refer to the Prediction Score Rescaling section in materials and methods). This score distribution was recalculated in each training iteration before the MA annotation. Including this rescaling step, the number of ligands estimated by cross-validation to be assigned to BoLA-2*02501 from the three cell lines increased to 13% on average.

However, investigating the motifs from the ligands predicted to be associated with BoLA-2*02501 from the A14 MA data to that from A12/A15 and A15 MA data, an inconsistency became apparent (see Figure 3.6-C and 3.6-D). Here, the motif obtained from the A14 MA data showed an additional preference for G at P2 that was completely absent for the motif obtained from the A15 and A12/A15 MA data. Re-examining the original publication that described the BoLA allele expression profile in the A14 haplotype [240], suggested an explanation to the apparent inconsistencies in the predicted BoLA-2*02501 binding motifs. In that paper, A14 was found to express 4 and not 3 BoLA alleles, as was assumed in our earlier publication and used in the first NNAlign_MA analysis. The extra allele expressed is BoLA-6*04001. After including this extra allele in the A14 haplotype and retraining the model, we obtained the binding motifs displayed in Figure 3.6-E, showing a motif for BoLA2*02501 consistent with the motif identified in the A12/A15 and A15 MA data (Figure 3.6-D), and a likewise well-defined motif for BoLA-6*04001 (Figure 3.6-F). These results clearly suggest that the motif earlier reported for BoLA-2*02501 (Figure 3.6-B) was a mixture of the motifs of BoLA-2*02501 and BoLA-6*04001.

The benchmark on the BoLA EL data confirmed the power of the NNAlign_MA method to achieve complete, consistent and well-defined deconvolution and motif identification of MHC alleles in the MA EL data sets, also in this challenging case with limited overlap between the SA and MA data (for details on the deconvolution refer to Supplementary Figure 3.S12). One notable example demonstrating this is BoLA-1:00901, a molecule contained within the A15 haplotype. BoLA-1:00901 shares limited overlap with the SA training data (distance to the SA data D = 0.137), and the motif predicted by the NNAlign_MA method after the pre-training on the SA data share, as expected, high similarity to the motif predicted by NetBoLApan (Figure 3.7). This pre-trained motif is, however, altered substantially after the training of the model on the EL MA BoLA data, resulting in a strong preference for Histidine (H) at PΩ (Post-training motif in Figure 3.7). To validate the accuracy of the motif predicted for BoLA-1:00901, we performed in vitro binding assays of a combinatorial peptide library to the BoLA1:00901 molecule (for details see Materials and Meth-

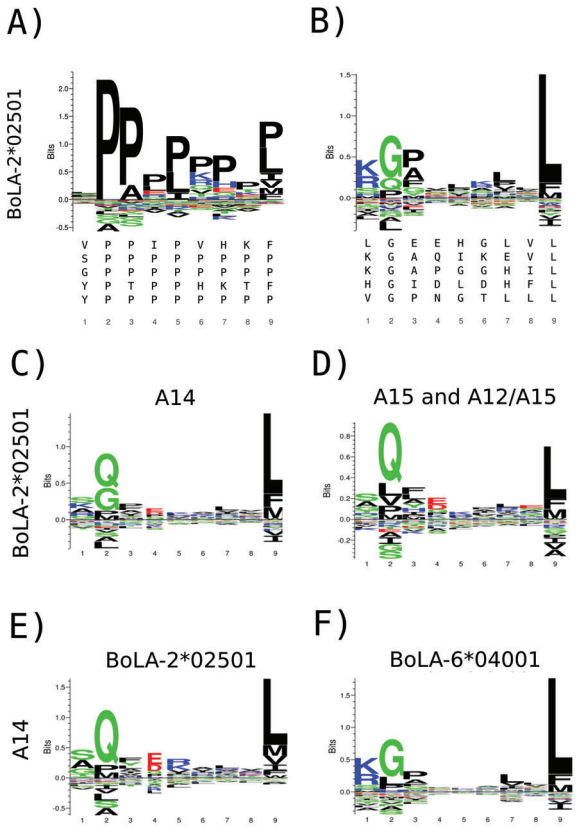

Figure 3.6. Identification of binding motifs for the BoLA-2*02501 molecule. (A) Binding preference for this molecule found by the NNAlign_MA method without score rescaling; (B) motif logo found in our previous work [167]. The top five repeating binding cores present in each motif alignment are shown below each logo. Binding motifs for BoLA-2*02501 obtained by NNAlign_MA trained including three BoLA alleles (BoLA-1*02301, BoLA-4*02401, and BoLA-2*02501) for the A14 MA data from ligands predicted using cross-validation to be restricted by BoLA-2*02501 from the A14 MA data (C), and from the A15 and A12/15 MA data sets (D). Binding motifs for the BoLA-2*02501 (E) and BoLA-6*04001 (F) molecules as estimated from ligands in the A14 MA data as predicted by NNAlign_MA using cross-validation when expanding the list of alleles in A14 to include BoLA-6*04001.

ods). The in vitro binding motif showed very high similarity to the Posttraining motif predicted by NNAlign_MA (Figure 3.7).

Evaluation on BoLA-I Epitopes

Having demonstrated the power of the proposed model also for the challenging BoLA system, we next evaluated its predictive power on a set of experimentally validated BoLA restricted CD8 epitopes. The result of this evaluation for NNAlign_MA and NetBoLApan confirmed the high performance of the proposed model (Table 3.2). Overall, the performance of the NNAlign_MA model is comparable to that of NetBoLApan. However, one notable example where the two models showed very different performance is the FVEGEAASH epitope, restricted by BoLA-1: 00901. For this epitope, the Frank performance value of NetBoLApan was 0.121-in other words, the true positive is found 12.1% down the list of candidate peptides predicted by NetBoLApan.Including the novel BoLA EL data and training the model using the NNAlign_MA framework, the Frank value for this epitope improved to 0.000—the epitope is the single top candidate predicted by NNAlign_MA. This result aligns with the experimental binding motif analysis of the BoLA-1:00901 molecule, exhibiting a strong preference for H at the C-terminal (Figure 3.7). In summary, the results displayed in Table 3.2 demonstrate the high predictive power of the model trained including the BoLA EL data also for prediction of CD8 epitopes. The average Frank value of NNAlign_MA over the 16 epitopes is 0.0033, meaning that on average 99.67% of the irrelevant peptide space can be excluded by the prediction model while still identifying 100% of the epitopes.

Performance values for NetBoLApan and NNAlign_MA are reported as Frank. In short, we predicted binding for all overlapping 8 –11mer peptides from the source protein of the epitopes to

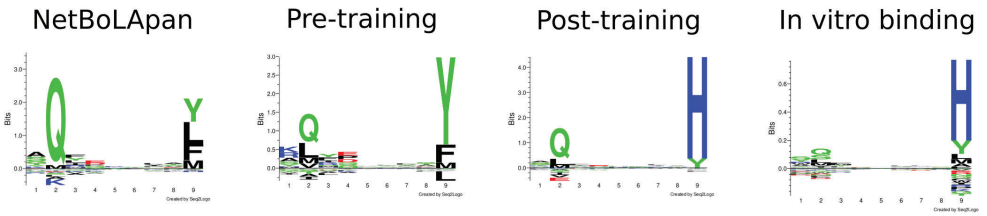

Figure 3.7. Binding motifs for the BoLA-1*00901 molecule estimated by different in silico and in vitro binding methods. Binding motifs for the three in silico methods were estimated from the top 0.1% of 1,000,000 random natural 9mer peptides with predicted binding by the given method, for BoLA-1:00901. The in vitro binding motifs were estimated using a position scanning combinatorial peptide library, as described in Materials and Methods. The three in silico methods are: NetBoLApan [167], trained including EL data for the cell lines A10, A14 and A18; Pre-training, the NNAlign_MA method pre-trained on SA data; Post-training, the NNAlign_MA method after completing the training including MA data.

	Allele	Epitope	Antigen	Npep	NetBoLApan	NNAlign_MA
T. parva	BoLA-6*01301	VGYPKVKEEML	Tp1	2138	0.0098	**0.0070**
	BoLA-6*04101	EELKKLGML	Tp2	662	**0.0000**	**0.0000**
	BoLA-2*01201	SSHGMGKVGK	Tp2	662	0.0060	**0.0045**
	BoLA-T2c	FAQSLVCVL	Tp2	662	0.0136	0.0151
	BoLA-2*01201	QSLVCVLMK	Tp2	662	0.0015	0.0015
	BoLA-AW10	TGASIQTTL	Tp4	2282	**0.0000**	**0.0000**
	BoLA-1*00902	SKADVIAKY	Tp5	586	**0.0000**	**0.0000**
	BoLA-T7	FISFPISL	Tp7	2850	0.0172	**0.0112**
	BoLA-3*00101	CGAELNHFL	Tp8	1726	0.0029	0.0035
	BoLA-1*02301	AKFPGMKKS	Tp9	1302	0.0054	**0.0023**
BHV	BoLA-1:00901	FVEGEAASH	ICP4	5350	0.1215	**0.0000**
	BoLA-3:00201	AGPDLQLARL	ICP4	5350	**0.0000**	**0.0000**
	BoLA-3:00201	TTPEILIEL	Circ	1006	**0.0000**	**0.0000**
	BoLA-3:01701	TGARAGYAA	ICP4	5350	0.0350	**0.0013**
	BoLA-4:02401	GAFCPEDW	ICP22	1214	0.0066	0.0058
	BoLA-2:01801	APAPSPGAL	Circ	978	0.0020	**0.0000**
	Average				0.0138	0.0033

Table 3.2. Predictive performance of NNAlign_MA and NetBoLApan on the set of known CD8 epitopes.

the known BoLA-I restriction molecule. Then, the performance for each epitope was reported as the Frank score. Epitope data for Theilera parva (T. parva) and Bovine Herpes Virus (BHV) were obtained from three sources [167, 241, 242]. The lowest Frank value for each epitope is highlighted in bold.

HLA-II Benchmark

To prove that the ability of NNAlign_MA to deal with MA data also extended to MHC II, a separate study was conducted on a set of MHC II BA and EL data. Here, we compared the cross-validated performance of NNAlign_MA trained on SA data alone (SA model) versus NNAlign_MA trained on the full data set including MA data (MA model). Both models were evaluated individually on the SA and MA data sets as described earlier for the HLA-I benchmark. The conclusions from this evaluation (Figure 3.8-A) were similar to those obtained for the HLA-I data: when evaluated on SA data, the SA model demonstrated a modest (and statistically insignificant, $p = 0.125$, binomial test) performance gain compared with the MA model. However, when it comes to the MA data, the MA model significantly outperformed the SA model ($p = 7.6 * 10^{-6}$, binomial test excluding ties). In Supplementary Figure 3.S13, we further show the binding motifs for MA data included in this study demonstrating that also for class II is the NNAlign_MA framework in most cases capable of achieving clear and consistent MHC motif deconvolution. However, as for the class I data does the accuracy of the motifs identified by NNAlign_MA also here depend on the number of ligands assigned to a given HLA. For example, is the motif for HLA-DRB1*13:01 most often derived from a very small number of ligands resulting in a limited similarity between the motifs obtained from the different MA data sets. This observation underlines the critical dependence of NNAlign_MA on the quality of the input data.

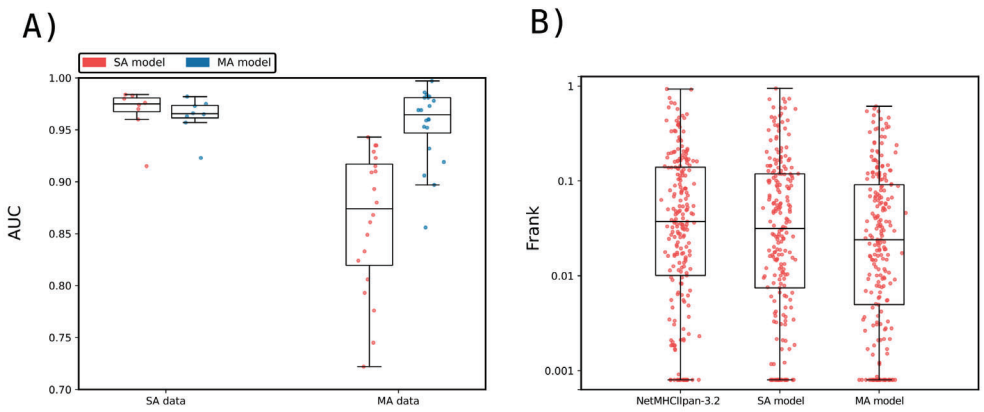

Figure 3.8. Method benchmarking on HLA class II data. A, Performance of NNAlign_MA trained on SA and MA data evaluated in crossvalidation. Model performance is given in terms of AUC and calculated as described in Figure 3.4. Each point refers to an individual SA or MA data set. B, Epitope Frank scores of NetMHCIIpan-3.2 and NNAlign_MA trained on SA and MA data evaluated on CD4+ epitopes from the IEDB [225]. The plot compares the Frank distribution of the MA and SA models and each point represents the Frank of an epitope. For visualization, Frank values of 0 are displayed with a value of 0.0008.

Next, we evaluated the performance of the SA and MA models together with NetMHCIIpan-3.2 on an independent data set of CD4+ epitopes (for details on this data set refer to Materials and Methods. The results of this benchmark are depicted in Figure 3.8-B in terms of Frank values. Here, the median Frank values were respectively 0.02403, 0.03155, and 0.03734 for the three methods MA, SA and NetMHCIIpan-3.2. The difference in Frank between the MA and the two other methods was in both cases found to be statistically significant ($p < 0.01$, binomial test excluding ties). These results strongly suggest that the NNAlign_MA framework extends its predictive power also into MHC class II.

Discussion

Advances in Mass Spectrometry have dramatically increased the throughput of immunopeptidomics experiments, with several thousands of peptides directly eluted from their cognate MHC molecule in a single experiment. This type of data has greatly changed our knowledge base for characterizing MHC antigen processing and presentation. In general, MS eluted ligands originate from multiple MHC molecules, and MS data sets therefore consist of a mixture of motifs, each corresponding to the binding specificity of one of the MHC molecules expressed by the cell line. Although several tools for the deconvolution of multiple motifs have been proposed, they all tend to underestimate the number of specificities in a sample, especially for haplotypes with overlapping MHC binding motifs and for alleles with low protein expression. Even for peptidomes that can be confidently deconvoluted, the pairing between motifs and the expressed MHC alleles is often not trivial, and in many cases must be done manually by visual inspection - with the potential sources of error this process entails.

Here, we have described a fully automated approach, NNAlign_MA, aiming to resolve these challenges. The approach taken in NNAlign_MA is very simple. The method applies a pre-training period where only single allele data (peptide data characterized by having a single MHC association) are included. After this pre-training, the multi-allele data (peptide data characterized by having two or more MHC associations) are annotated using the current prediction model to predict binding to all MHC molecules possible for the peptide, and next defining a single MHC association from the highest prediction value. In this annotation step, multiallele data are thus casted into a single-allele format, becoming manageable by the NNAlign method and therefore enabled for training. This multi allele annotation step is iteratively performed in each training cycle.

We have applied the NNAlign_MA method to analyze and interpret three large-scale multi-allele MHC eluted ligand data sets, and demonstrated its unprecedented performance compared with state-of-the-art methods. First, we applied the method to analyze multi-allele HLA MS eluted ligand data from 50 cell lines. Using this data, we demonstrated how the method in most cases was capable of correctly identifying distinct binding motifs for each of the HLA molecules expressed in a given cell line. This result contrasts with findings using earlier methods such as GibbsCluster and MixMHCp that in most cases fail to identify one or more motifs. Also, NNAlign_MA was in close to all cases capable of accurately associating each identified motif with a specific HLA

molecule. These results highlighted the high performance of NNAlign_MA compared with current state-of-the-art methods such as MixMHCp/MixMHCpred, where the association of binding motifs to individual HLA molecules is achieved by exclusion principles identifying binding motifs shared uniquely between different cell line data sets.

In terms of predictive performance, the models trained using the NNAlign_MA method were found to outperform conventional methods (trained on single-allele data), both for prediction of HLA eluted ligand data and CD8 epitopes. As expected, this performance gain was most pronounced for ligands/epitopes restricted by HLA molecules characterized by limited or no single-allele data. This observation underlines the single most important power of NNAlign_MA, namely to effectively expand the part of the HLA space covered by accurate predictions. By way of example, the SA EL data included in this study covers 51 HLA molecules. Earlier studies have demonstrated that pan-specific prediction methods allow to accurately predict the binding specificity also for HLA molecules not characterized by binding data, if their distance to a molecule characterized by binding data is 0.1 or less (for a definition of this distance refer to Materials and Methods) [238]. Applying this rule to the set of 10,558 functional HLA class I A, B and C alleles contained within IPD-IMGT/HLA release 3.35 [243] results in a coverage of 76% (8,051 out of 10,558 molecules). By integrating the MA data, the number of alleles covered by EL data is expanded to 85, and number of HLA molecules covered by accurate predictions to 94% (9,949 of 10,558 molecule).

This power of NNAlign_MA to expand the allelic coverage was further demonstrated in a specificity leave-out experiment. Here, entire HLA specificity groups were removed from the single-allele data set, and the NNAlign_MA framework applied to analyze and characterize multi-allele data including HLA molecules from these removed specificity groups. The result of this experiment confirmed the power of NNAlign_MA to expand the allelic coverage and accurately identifying binding motifs for individual HLA molecules in multi-allele data, also in situations where no explicit information about the binding preferences of the investigated molecules was part of the single-allele training data.

The HLA system has been studied in great detail over the past decades, and peptide-MHC binding data are available for hundreds of alleles. To further explore the predictive power of NNAlign_MA for MHC systems characterized by limited data, we turned to the Bovine Leukocyte Antigen (BoLA) system, and applied NNAlign_MA to analyze MS MHC eluted ligands data sets from 8 cell lines expressing a total of 8 haplotypes covering 16 distinct BoLA molecules. These BoLA molecules shared, for most parts, very low similarity to the molecules included in the single-allele data. Also in this setting, NNAlign_MA was demonstrated to accurately identify binding motifs in all BoLA data sets, and the model trained on the BoLA MA data demonstrated a high predictive power for identification of known BoLA restricted CD8 epitopes, identifying the epitopes within the top 0.3% of the peptides within the epitope source protein sequence. These results thus further demonstrated how NNAlign_MA was capable of correctly deconvoluting binding motifs present in multi-allele data in situations with limited shared similarity to the single-allele data.

As a final validation, the NNAlign_MA framework was applied to MS EL data from MHC II. Also here, the models were evaluated in cross-validation and on an independent set of CD4+ epitopes and the results were in agreement with the results obtained for MHC I. That is, the model trained including MA data showed significantly improved performance compared with models trained on SA data only, when evaluated on both MA EL data and CD4 epitopes.

In a recent work, Bulik-Sullivan et al. [244] have suggested an alternative approach to deconvolute and train MHC antigen presentation prediction models using an allele-specific architecture, thus limiting the predictive coverage of the model to MHC alleles present in the training data. This contrasts with the architecture of NNAlign_MA, which enables pan-specific predictions covering alleles outside the training data (as described above). Also, the allele-specific nature of the method proposed by Bulik-Sullivan et al. limits the power of the tool to identify motifs and construct prediction models for the alleles included in the training data. By way of example in the data presented by Bulik-Sullivan et al., less than 65% of the alleles in their training set ended up covered by a prediction model. Future work and independent benchmarking will allow us to evaluate which of the two approaches is optimal for a given epitope discovery setting.

We have demonstrated how NNAlign_MA achieves binding motif deconvolution driven by similarity to MHC molecules characterized by single-specificity data, and by principles of co-occurrence and exclusion of MHC molecules between different poly-specificity MS eluted ligand data set. The NNAlign_MA failed to construct accurate binding motifs for a few limited HLA molecules. These cases were all characterize by very few ligand data, and by alleles only present in single MA data sets. This observation, combined with the power of NNAlign_MA to expand the allelic coverage of the resulting prediction model, points to a direct application to effectively achieve broad and

high accuracy allelic coverage for regions of the MHC repertoire with yet uncharacterized binding specificities. Guided by NNAlign_MA, sets of cell lines with characterized HLA expression should be selected for LC-MS/MS to maximize allele co-occurrence, allele exclusion and allele similarities with the comprehensive set of available EL data so that the NNAlign_MA motif deconvolution for the uncharacterized binding specificities can be achieved in an optimal manner. We believe this approach for generating additional MA data to be a highly effective manner to further improve prediction of MHC antigen presentation, moving beyond the limitations associated with fulfilling this task using artificial single allele MS setups.

Although peptide-MHC binding is arguably the most selective step in the MHC antigen presentation pathway, other properties contribute to determining immunogenicity of T-cell epitopes. The above-mentioned work by Bulik-Sullivan et al. [244], attempted to incorporate gene expression levels and proteasome cleavage preferences in a machine-learning model, showing promising improvements for the prediction of cancer neo-epitopes. For the MHC class II system, consistent signatures of peptide trimming and processing have been detected, with pioneering attempts to incorporate them in T-cell epitope prediction models [215, 245]. In future developments of the NNAlign_MA framework, the effect on the predictive power of incorporating such additional potential correlates of immunogenicity will be investigated.

Overall, we have evaluated the proposed NNAlign_MA framework on a large and diverse set of data, and demonstrated how the method in all cases was capable of achieving a complete deconvolution of binding motifs contained within poly-specific MS eluted ligand data, and how the complete deconvolution enabled training prediction models with expanded HLA allelic coverage for accurate identification of both eluted ligands and T-cell epitopes. In conclusion, we believe NNAlign_MA offers a universal solution to the challenge of analyzing large-scale MHC peptidomics data sets and consequently affords an optimal way of exploiting the information contained in such data for improving prediction of MHC binding and antigen presentation. The modeling framework is readily extendable to include peptides with posttranslational modifications [159, 246], and signals from antigen processing located outside the sequence of the ligands [215]. Given its very high flexibility, we expect NNAlign_MA to serve as an effective tool to further our understanding of the rules for MHC antigen presentation, as a guide for improved T-cell epitope discovery and as an aid for effective development of T-cell therapeutics.

Supplementary Material

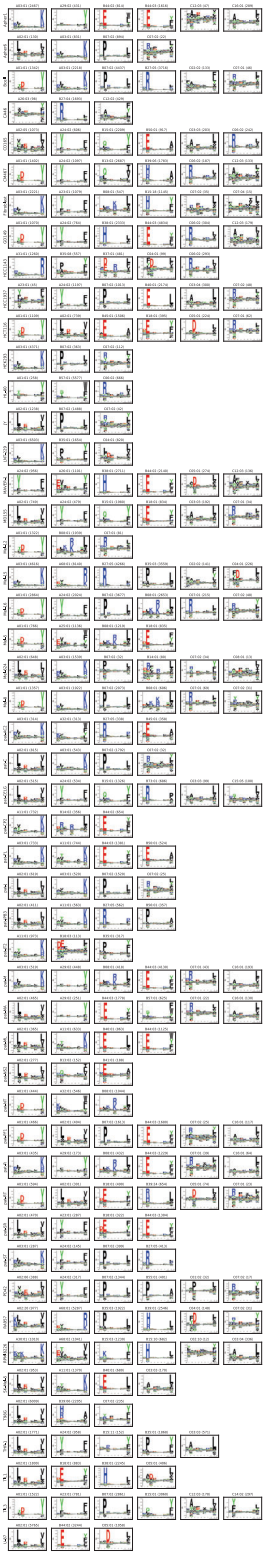

Figure 3.S9. Full NNAlign_MA motif deconvolution for the Multi Allele (MA) HLA-I data analyzed in this work. Each row corresponds to a cell line present in the training data (50 in total; for more details, refer to Supplementary Table 3.S6). Using cross validation, each ligand is assigned to one of the HLA alleles expressed in the given cell line. Using this assignment, binding motifs were generated for each allele in each cell line using Seq2Logo. To remove potential MS contaminants, only ligands with a prediction score greater than 0.01 were included. Above each logo is given the number of sequences associated to the corresponding HLA allele. For details on the accuracy of this clustering, refer to Supplementary Figures 3.S10 and 3.S11

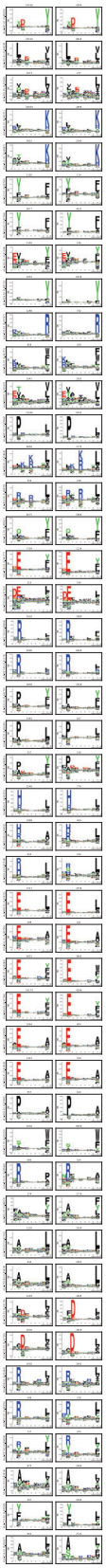

Figure 3.S10. Comparison between NNAlign_MA deconvoluted motifs and motifs derived from single-allele (SA) data from the IEDB [225] for alleles characterised by at least 100 ligands for both SA data and MA deconvolution data. For each HLA allele, the NNAlign_MA motif is displayed in the first column; the motif derived from available SA data in the second column. The Pearson correlation coefficient between two motifs is displayed next to the corresponding HLA name (for details on how this is calculated refer to materials and methods). Alleles whose logo was generated from data contained only in the MA training set (this is, no SA data was present in the training phase) are tagged with an asterisk. To remove potential MS contaminants, only ligands with a prediction score greater than 0.01 were included. The amount of sequences employed to construct a given logo is displayed on top of each logo.

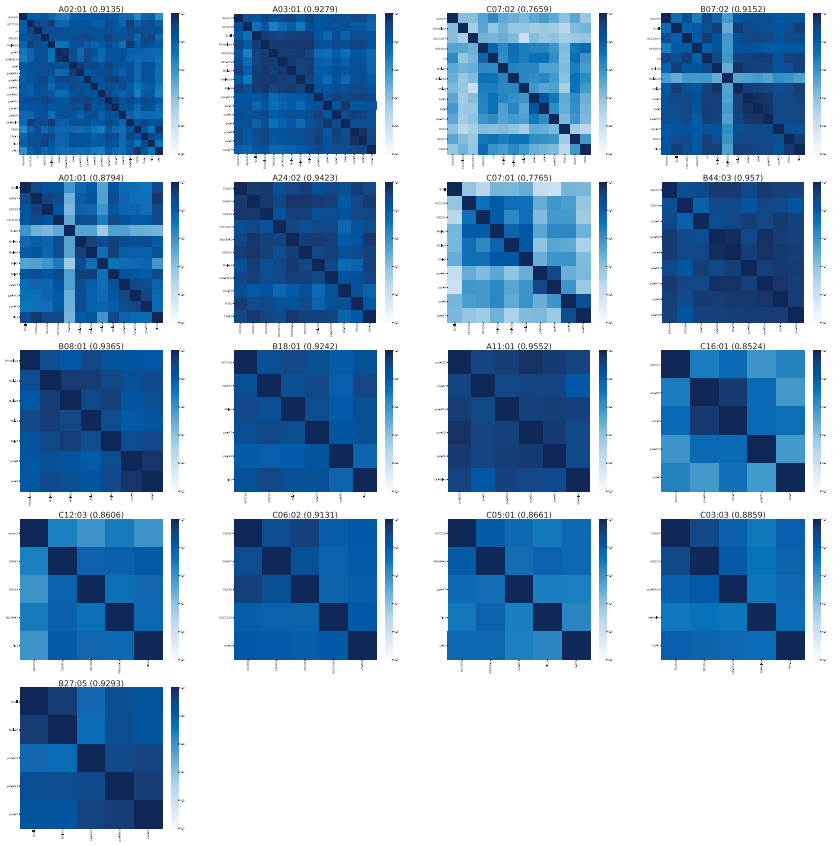

Figure 3.S11. Correlation matrices between NNAlign_MA motifs for HLA alleles that are shared between five or more cell lines. Each matrix displays the Pearson correlation coefficients between all motifs found by NNAlign_MA for a given HLA allele, across all the cell lines sharing the allele (for details on how the correlation is calculated refer to materials and methods). The average Pearson correlation coefficient for each matrix is given next to the corresponding allele name.

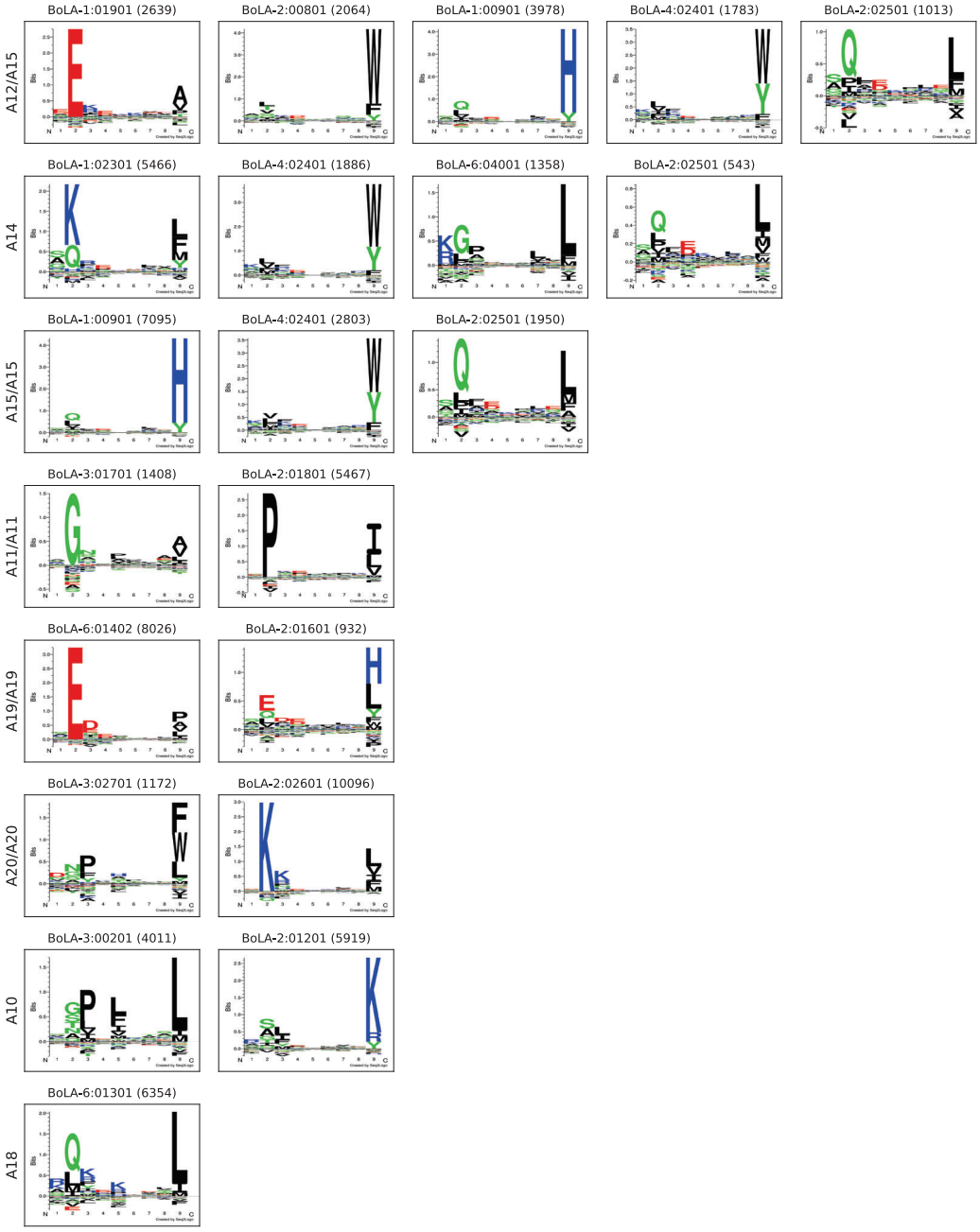

Figure 3.S12. Full NNAlign_MA motif deconvolution for the Multi Allele (MA) BoLA-I data analyzed in this work. Each row corresponds to a MA data set present in the training data. Using cross validation, each ligand is assigned to one of the BoLA alleles expressed in the given data set. Using this assignment, binding motifs were generated for each allele in each cell line using Seq2Logo. To remove potential MS contaminants, only ligands with a prediction score greater than 0.01 were included. Above each logo is given the number of sequences associated to the corresponding BoLA allele.

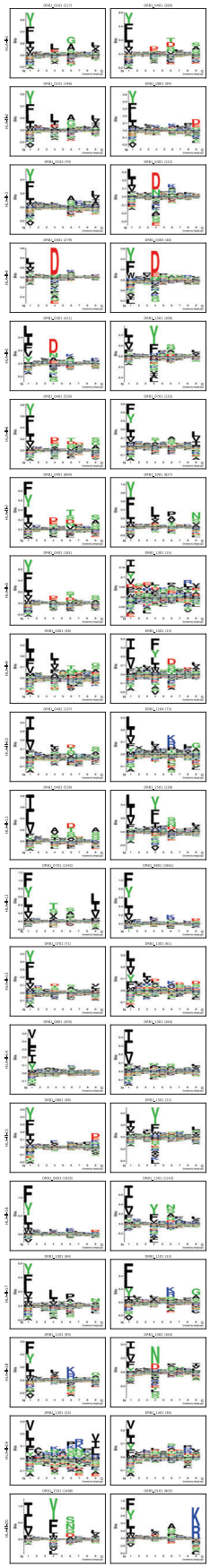

Figure 3.S13. Full NNAlign_MA motif deconvolution for the Multi Allele (MA) HLA-II data analyzed in this work. Each row corresponds to a MA data set present in the training data. Using cross validation, each ligand is assigned to one of the HLA alleles expressed in the given data set. Using this assignment, binding motifs were generated for each allele in each cell line using Seq2Logo. To remove potential MS contaminants, only ligands with a prediction score greater than 0.01 were included. Above each logo is given the number of sequences associated to the corresponding HLA allele.

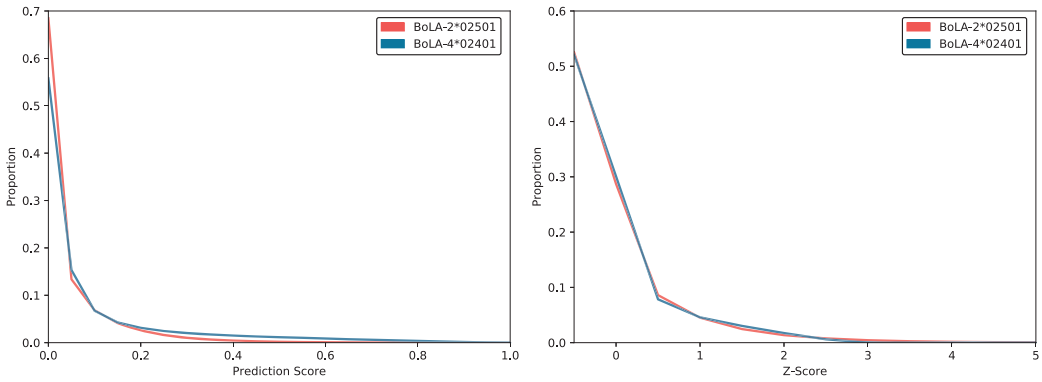

Figure 3.S14. Prediction score rescaling for the BoLA-2*02501 and BoLA-4*02401 molecules. Left panel: proportion of ligands as a function of the prediction scores for both BoLA molecules. In this case, distributions exhibit different characteristics (i.e. more than 13% of random natural peptides have a prediction score greater than 0.3 for BoLA-4*02401, while for BoLA-2*0251 this number is less than 3%). Right panel: proportion of ligands as a function of the Z-Score transformation of the prediction scores. This type of rescaling allows reshaping both distributions so they display similar silhouettes

Cell line ID	Positives	Negatives	HLA-A		HLA-B		HLA-C		Source PMID
CD165	5364	132869	HLA-A*02:05	HLA-A*24:02	HLA-B*15:01	HLA-B*50:01	HLA-C*03:03	HLA-C*06:02	28832583
CM467	7401	184646	HLA-A*01:01	HLA-A*24:02	HLA-B*13:02	HLA-B*39:06	HLA-C*06:02	HLA-C*12:03	
GD149	9756	208444	HLA-A*01:01	HLA-A*24:02	HLA-B*38:01	HLA-B*44:03	HLA-C*06:02	HLA-C*12:03	
MD155	4374	108036	HLA-A*02:01	HLA-A*24:02	HLA-B*15:01	HLA-B*18:01	HLA-C*03:03	HLA-C*07:01	
PD42	2577	48693	HLA-A*02:06	HLA-A*24:02	HLA-B*07:02	HLA-B*55:01	HLA-C*01:02	HLA-C*07:02	
RA957	11037	232658	HLA-A*02:20	HLA-A*68:01	HLA-B*35:03	HLA-B*39:01	HLA-C*04:01	HLA-C*07:02	
TIL1	5445	140312	HLA-A*02:01	HLA-A*02:01	HLA-B*18:01	HLA-B*38:01	HLA-C*05:01	-	
TIL3	8799	206212	HLA-A*01:01	HLA-A*23:01	HLA-B*07:02	HLA-B*15:01	HLA-C*12:03	HLA-C*14:02	
Apher1	6145	123349	HLA-A*03:01	HLA-A*29:02	HLA-B*44:02	HLA-B*44:03	HLA-C*12:03	HLA-C*16:01	
Apher6	1962	39798	HLA-A*02:01	HLA-A*03:01	HLA-B*07:02	-	HLA-C*07:02	-	
Mel-15	21813	395324	HLA-A*03:01	HLA-A*68:01	HLA-B*27:05	HLA-B*35:03	HLA-C*02:02	HLA-C*04:01	27869121
Mel-16	11980	264233	HLA-A*01:01	HLA-A*24:02	HLA-B*07:02	HLA-B*08:01	HLA-C*07:01	HLA-C*07:02	
Mel-12	3758	88425	HLA-A*01:01	HLA-A*01:01	HLA-B*08:01	-	HLA-C*07:01	-	
Mel-8	6251	119000	HLA-A*01:01	HLA-A*03:01	HLA-B*07:02	HLA-B*08:01	HLA-C*07:01	HLA-C*07:02	
Mel-5	4749	106896	HLA-A*01:01	HLA-A*25:01	HLA-B*08:01	HLA-B*18:01	-	-	
Fibroblast	5289	122127	HLA-A*03:01	HLA-A*23:01	HLA-B*08:01	HLA-B*15:18	HLA-C*07:02	HLA-C*07:04	25576301
HCC1143	2780	69565	HLA-A*31:01	-	HLA-B*35:08	HLA-B*37:01	HLA-C*04:01	HLA-C*06:02	
HCC1937	4976	102331	HLA-A*23:01	HLA-A*24:02	HLA-B*07:02	HLA-B*40:01	HLA-C*03:04	HLA-C*07:02	
HCT116	4174	93208	HLA-A*01:01	HLA-A*02:01	HLA-B*45:01	HLA-B*18:01	HLA-C*05:01	HLA-C*07:01	
JY	2868	60863	HLA-A*02:01	-	HLA-B*07:02	-	HLA-C*07:02	-	
Bcell	12199	220971	HLA-A*01:01	HLA-A*03:01	HLA-B*07:02	HLA-B*27:05	HLA-C*02:02	HLA-C*07:01	24616531
Mel-624	2375	49050	HLA-A*02:01	HLA-A*03:01	HLA-B*07:02	HLA-B*14:01	HLA-C*07:02	HLA-C*08:01	27600516
SK-Mel-5	3293	64537	HLA-A*02:01	HLA-A*11:01	HLA-B*40:01	-	HLA-C*03:03	-	
HEK293	4972	86634	HLA-A*03:01	-	HLA-B*07:02	-	HLA-C*07:02	-	26992070
MAVER-1	7403	171783	HLA-A*24:02	HLA-A*26:01	HLA-B*38:01	HLA-B*44:02	HLA-C*05:01	HLA-C*12:03	
HL-60	6607	115694	HLA-A*01:01	-	HLA-B*57:01	-	HLA-C*06:02	-	
RPMI8226	4524	113201	HLA-A*30:01	HLA-A*68:02	HLA-B*15:03	HLA-B*15:10	HLA-C*02:10	HLA-C*03:04	
THP-1	5542	142866	HLA-A*02:01	HLA-A*24:02	HLA-B*15:11	HLA-B*35:01	HLA-C*03:03	-	
CA46	2324	62647	HLA-A*26:03	-	HLA-B*27:04	-	HLA-C*12:02	-	
LNT-229	10311	177908	HLA-A*03:01	-	HLA-B*35:01	-	HLA-C*04:01	-	27412690
T98G	10011	216072	HLA-A*02:01	-	HLA-B*39:06	-	HLA-C*07:02	-	
U-87	11585	241396	HLA-A*02:01	-	HLA-B*44:02	-	HLA-C*05:01	-	
pat-AC2	1369	32168	HLA-A*03:01	HLA-A*32:01	HLA-B*27:05	HLA-B*45:01	-	-	27841757
pat-C	2983	49759	HLA-A*02:01	HLA-A*03:01	HLA-B*07:02	-	HLA-C*07:02	-	
pat-CELG	3814	72328	HLA-A*02:01	HLA-A*24:02	HLA-B*15:01	HLA-B*73:01	HLA-C*03:03	HLA-C*15:05	
pat-CP2	1790	36895	HLA-A*11:01	-	HLA-B*14:02	HLA-B*44:02	-	-	
pat-FL	3629	74392	HLA-A*03:01	HLA-A*11:01	HLA-B*44:03	HLA-B*50:01	-	-	
pat-J	2552	42497	HLA-A*02:01	HLA-A*03:01	HLA-B*07:02	-	HLA-C*07:02	-	
pat-JPB3	1937	35295	HLA-A*02:01	HLA-A*11:01	HLA-B*27:05	HLA-B*56:01	-	-	
pat-JT2	1467	29587	HLA-A*11:01	-	HLA-B*18:03	HLA-B*35:01	-	-	
pat-M	2476	53262	HLA-A*03:01	HLA-A*29:02	HLA-B*08:01	HLA-B*44:03	HLA-C*07:01	HLA-C*16:01	
pat-MA	3682	69891	HLA-A*02:01	HLA-A*29:02	HLA-B*44:03	HLA-B*57:01	HLA-C*07:01	HLA-C*16:01	
pat-ML	3139	55262	HLA-A*02:01	HLA-A*11:01	HLA-B*40:01	HLA-B*44:03	-	-	
pat-NS2	636	15212	HLA-A*02:01	-	HLA-B*13:02	HLA-B*41:01	-	-	
pat-NT	2190	53238	HLA-A*01:01	HLA-A*32:01	HLA-B*08:01	-	-	-	
pat-PF1	4646	86859	HLA-A*01:01	HLA-A*02:01	HLA-B*07:02	HLA-B*44:03	HLA-C*07:02	HLA-C*16:01	
pat-R	2372	49169	HLA-A*03:01	HLA-A*29:02	HLA-B*08:01	HLA-B*44:03	HLA-C*07:01	HLA-C*16:01	
pat-RT	2537	49846	HLA-A*01:01	HLA-A*02:01	HLA-B*18:01	HLA-B*39:24	HLA-C*05:01	HLA-C*07:01	
pat-SR	2632	57417	HLA-A*02:01	HLA-A*23:01	HLA-B*18:01	HLA-B*44:03	-	-	
pat-ST	1256	26963	HLA-A*03:01	HLA-A*24:02	HLA-B*07:02	HLA-B*27:05	-	-	

Table 3.S3. Summary of the multi-allele (MA) data included in the HLA-I benchmark. "Positives" and "Negatives" refer to the number of positive and negative instances contained in each cell line data. Further rows show the HLA-A, HLA-B and HLA-C expressed by a given cell line (two per locus), together with the Source ID for its corresponding dataset.

BA data	
Alleles	**#Peptides**
HLA-A01:01	3985
HLA-A02:01	11097
HLA-A02:02	3613
HLA-A02:03	5728
HLA-A02:04	3
HLA-A02:05	66
HLA-A02:06	4802
HLA-A02:07	66
HLA-A02:10	18
HLA-A02:11	1081
HLA-A02:12	1181
HLA-A02:16	919
HLA-A02:17	341
HLA-A02:19	1243
HLA-A02:50	134
HLA-A03:01	6513
HLA-A03:02	26
HLA-A03:19	30
HLA-A11:01	5614
HLA-A11:02	14
HLA-A23:01	1871
HLA-A24:02	2315
HLA-A24:03	1374
HLA-A25:01	959
HLA-A26:01	3729
HLA-A26:02	641
HLA-A26:03	535
HLA-A29:02	2286
HLA-A30:01	2711
HLA-A30:02	1564
HLA-A31:01	5152
HLA-A32:01	908
HLA-A32:07	88
HLA-A32:15	74
HLA-A33:01	2508
HLA-A66:01	207
HLA-A68:01	3190
HLA-A68:02	4542
HLA-A68:23	82
HLA-A69:01	2559
HLA-A74:01	15
HLA-A80:01	1167
HLA-B07:02	4100
HLA-B08:01	3171
HLA-B08:02	1018
HLA-B08:03	469
HLA-B14:01	42
HLA-B14:02	283
HLA-B15:01	4213
HLA-B15:02	164
HLA-B15:03	604
HLA-B15:09	830
HLA-B15:17	1444
HLA-B15:42	364
HLA-B18:01	2370
HLA-B27:01	4
HLA-B27:02	8
HLA-B27:03	874
HLA-B27:04	4
HLA-B27:05	3372
HLA-B27:06	7
HLA-B27:10	2
HLA-B27:20	92
HLA-B35:01	2724
HLA-B35:03	93
HLA-B35:08	1
HLA-B37:01	50
HLA-B38:01	500
HLA-B39:01	1785
HLA-B40:01	2984
HLA-B40:02	712
HLA-B40:13	59
HLA-B42:01	160
HLA-B42:02	18
HLA-B44:02	1954
HLA-B44:03	1006
HLA-B45:01	627
HLA-B45:06	362
HLA-B46:01	1798
HLA-B48:01	881
HLA-B51:01	2383
HLA-B52:01	12
HLA-B53:01	1341
HLA-B54:01	731
HLA-B57:01	2640
HLA-B57:02	18
HLA-B57:03	34
HLA-B58:01	3116
HLA-B58:02	56
HLA-B73:01	122
HLA-B81:01	26
HLA-B83:01	339
HLA-C03:03	153
HLA-C04:01	552
HLA-C05:01	172
HLA-C06:02	309
HLA-C07:01	241
HLA-C07:02	142
HLA-C08:02	87
HLA-C12:03	172
HLA-C14:02	259
HLA-C15:02	252
HLA-E01:01	96
HLA-E01:03	55
Mamu-A01	2466
Mamu-A02	1188
Mamu-A07	535
Mamu-A11	1144
Mamu-A20102	132
Mamu-A2201	582
Mamu-A2601	142
Mamu-A70103	95
Mamu-B01	444
Mamu-B03	973
Mamu-B04	2
Mamu-B08	952
Mamu-B1001	140
Mamu-B17	1384
Mamu-B3901	439
Mamu-B52	846
Mamu-B6601	101
Mamu-B8301	368
Mamu-B8701	144
Patr-A0101	337
Patr-A0301	262
Patr-A0401	233
Patr-A0602	1
Patr-A0701	495
Patr-A0901	621
Patr-B0101	636
Patr-B0901	1
Patr-B1301	196
Patr-B1701	8
Patr-B2401	293
SLA-10401	185
SLA-10701	23
SLA-20401	105
SLA-30401	76
BoLA-AW10	166
BoLA-D18.4	238
BoLA-HD6	268
BoLA-JSP.1	158
BoLA-T2a	167
BoLA-T2b	157
BoLA-T2C	90
Gogo-B0101	14
H-2-Db	2580
H-2-Dd	276
H-2-Kb	3694
H-2-Kd	811
H-2-Kk	364
H-2-Ld	260
H-2-Lq	2

EL data	
Alleles	**#Peptides**
HLA-A01:01	3405
HLA-A02:01	5349
HLA-A02:05	98
HLA-A02:07	30
HLA-A03:01	427
HLA-A11:01	313
HLA-A24:02	2699
HLA-A24:06	169
HLA-A24:13	49
HLA-A26:01	97
HLA-A29:02	4377
HLA-A31:01	31
HLA-A32:01	29
HLA-B07:02	3324
HLA-B08:01	458
HLA-B13:01	57
HLA-B15:01	455
HLA-B15:02	52
HLA-B18:01	46
HLA-B27:02	2334
HLA-B27:03	278
HLA-B27:04	569
HLA-B27:05	2567
HLA-B27:06	646
HLA-B27:07	1253
HLA-B27:08	1306
HLA-B27:09	1363
HLA-B35:01	680
HLA-B35:03	23
HLA-B35:08	93
HLA-B37:01	39
HLA-B39:06	495
HLA-B40:01	1286
HLA-B40:02	1548
HLA-B41:01	19
HLA-B41:03	55
HLA-B41:04	37
HLA-B44:02	1662
HLA-B44:03	303
HLA-B44:27	24
HLA-B44:28	18
HLA-B45:01	150
HLA-B49:01	119
HLA-B50:01	114
HLA-B51:01	2424
HLA-B57:01	370
HLA-C03:04	29
HLA-C04:01	366
HLA-C05:01	435
HLA-C07:02	19
HLA-C16:01	222
H2-Db	808
H2-Kb	1906
H2-Kd	663
Mamu-B008:01	495

Table 3.S4. Single Allele (SA) Binding Affinity (BA) and Eluted Ligands (EL) training data summary for the HLA-I system. For the BA training set, the total amount of sequences per MHC molecule (discarding artificial negatives) is shown; in the case of EL data, the total amount of positives is displayed.

ID	Positives	Negatives	HLA-DRB		Source PMID
HLA-II-1	509	5355	DRB1*01:01	DRB1*04:01	27726376
HLA-II-2	240	2655	DRB1*01:01	DRB1*08:01	27726376
HLA-II-3	200	1890	DRB1*01:03	DRB1*03:01	27726376
HLA-II-4	327	3285	DRB1*03:01	DRB1*03:05	27726376
HLA-II-5	595	6930	DRB1*03:01	DRB1*15:01	27452731, 27726376
HLA-II-6	670	7740	DRB1*04:01	DRB1*07:01	27452731
HLA-II-7	1772	17460	DRB1*04:01	DRB1*10:01	27726376
HLA-II-8	213	2520	DRB1*04:01	DRB1*13:01	27452731
HLA-II-9	51	480	DRB1*04:01	DRB1*15:01	27726376
HLA-II-10	210	2475	DRB1*04:02	DRB1*11:04	27726376
HLA-II-11	682	7335	DRB1*04:03	DRB1*15:01	27726376
HLA-II-12	3216	29565	DRB1*07:01	DRB1*08:01	29632711
HLA-II-13	145	1710	DRB1*07:01	DRB1*13:01	27452731
HLA-II-14	496	4860	DRB1*08:01	DRB1*13:01	27452731, 29632711
HLA-II-15	121	1440	DRB1*08:01	DRB1*15:01	27726376
HLA-II-16	3080	29745	DRB1*08:03	DRB1*13:01	29632711
HLA-II-17	118	1215	DRB1*10:01	DRB1*11:01	27726376
HLA-II-18	257	2835	DRB1*11:01	DRB1*13:02	27726376
HLA-II-19	65	585	DRB1*13:01	DRB1*14:01	27452731
HLA-II-20	2426	22365	DRB1*15:01	DRB5*01:01	28467828

Table 3.S5. Multi Allele (MA) data summary for the HLA-II benchmark. "Positives" and "Negatives" refer to the number of positive and negative instances contained in each MA EL data set. Further rows show the HLADRB alleles expressed by a given cell line, together with the Source PMID(s) for its corresponding dataset.

Cell line ID	Haplotype	Positives	Negatives	BoLA-1	BoLA-2	BoLA-3	BoLA-4	BoLA-6	Source PMID
2123	A12/A15	11872	271523	BoLA-1*01901	BoLA-2*00801	-	BoLA-4*02401	-	-
				BoLA-1*00901	BoLA-2*02501				
5072	A11	8542	155590	-	BoLA-2*01801	BoLA-3*01701	-	-	
2824	A19	9582	153620	-	BoLA-2*01601	-	-	BoLA-6*01402	
5350	A20	11726	240196	-	BoLA-2*02601	BoLA-3*02701	-	-	
2408	A15	24305	552309	BoLA-1*00901	BoLA-2*02501		BoLA-4*02401	-	
1011/500004	A10	10188	148801	-	BoLA-2*01201	BoLA-3*00201	-	-	29115832
641	A18	6615	80170	-	-	-	-	BoLA-6*01301	
2229/104003	A14	9509	186084	BoLA-1*02301	BoLA-2*02501	-	BoLA-4*02401	BoLA-6*04001	

Table 3.S6. Multi Allele (MA) data summary for the BoLA benchmark. "Positives" and "Negatives" refer to the number of positive and negative instances contained in each cell line data. Further rows show the BoLA-1, BoLA-2, BoLA-3, BoLA-4, and BoLA-6 alleles expressed by a given cell line, together with the Source PMID for its corresponding dataset. Allele annotation was obtained from Vasoya, D. et al. [240]

Cell Line ID	HLA Allele	PPV
Aphar1	HLA-A03:01	0.80
	HLA-A24:02	0.72
	HLA-B44:02	0.81
	HLA-B44:03	0.91
	HLA-C12:03	0.71
	HLA-C16:01	0.89
Aphar6	HLA-A02:01	0.72
	HLA-A03:01	0.82
	HLA-B07:02	0.91
	HLA-C07:02	0.96
Ibis6	HLA-A01:01	0.79
	HLA-A03:01	0.75
	HLA-B07:02	0.86
	HLA-B27:05	0.93
	HLA-C02:02	0.93
	HLA-C07:01	0.94
CA46	HLA-A26:03	0.35
	HLA-B27:06	0.9
	HLA-C12:02	0.77
CD165	HLA-A02:05	0.84
	HLA-A24:02	0.81
	HLA-B15:16	0.87
	HLA-B50:01	0.92
	HLA-C03:03	0.75
	HLA-C06:02	0.83
CH 467	HLA-A01:01	0.9
	HLA-A24:02	0.89
	HLA-B13:02	0.8
	HLA-B39:06	0.75
	HLA-C06:02	0.83
	HLA-C12:03	0.89
Fibroblast	HLA-A03:01	0.86
	HLA-A23:01	0.88
	HLA-B08:01	0.97
	HLA-B15:16	0.88
	HLA-C07:02	0.91
	HLA-C07:04	0.5
GD149	HLA-A01:01	0.79
	HLA-A24:02	0.76
	HLA-B08:01	0.9
	HLA-B44:03	0.92
	HLA-C06:02	0.92
	HLA-C12:03	0.95
HCC1143	HLA-A31:01	0.89
	HLA-B35:08	0.92
	HLA-B37:01	0.77
	HLA-C04:01	0.95
	HLA-C06:02	0.8
HCC1937	HLA-A23:01	0.53
	HLA-A24:02	0.97
	HLA-B07:02	0.93
	HLA-B40:01	0.9
	HLA-C02:02	0.74
	HLA-C07:02	0.35
HCT116	HLA-A01:01	0.86
	HLA-A02:01	0.74
	HLA-B08:01	0.86
	HLA-B45:01	0.89
	HLA-C05:01	0.71
	HLA-C07:01	0.59
HSK293	HLA-A03:01	0.9
	HLA-B07:02	0.71
	HLA-C07:02	0.94
HL-60	HLA-A31:01	0.36
	HLA-B57:01	0.92
	HLA-C06:02	0.81
J7	HLA-A02:01	0.85
	HLA-B07:02	0.91
	HLA-C07:02	0.48
LNT-229	HLA-A03:01	0.92
	HLA-B35:01	0.89
	HLA-C04:01	0.47
MAVER-1	HLA-A24:02	0.85
	HLA-A26:01	0.81
	HLA-B08:01	0.92
	HLA-B44:02	0.9
	HLA-C05:01	0.94
MD255	HLA-C12:03	0.82
	HLA-A02:01	0.79
	HLA-A24:02	0.88
	HLA-B35:01	0.89
	HLA-B08:01	0.87
	HLA-C03:03	0.72
	HLA-C07:01	0.85
Mel-12	HLA-A31:01	0.77
	HLA-B08:01	0.91
	HLA-C07:01	0.83
Mel-15	HLA-A03:01	0.76
	HLA-A68:01	0.81
	HLA-B27:05	0.76
	HLA-B35:03	0.82
	HLA-C02:02	0.88
	HLA-C04:01	0.36
Mel-16	HLA-A03:01	0.77
	HLA-A24:02	0.85
	HLA-B07:02	0.89
	HLA-B08:01	0.87
	HLA-C07:01	0.8
	HLA-C07:02	0.52
Mel-5	HLA-A31:01	0.72
	HLA-A25:01	0.82
	HLA-B08:01	0.83
	HLA-B18:01	0.87
Mel-624	HLA-A02:01	0.82
	HLA-A03:01	0.86
	HLA-B07:02	0.91
	HLA-B14:01	0.71
	HLA-C07:02	0.82
	HLA-C08:01	0.25
Mel-8	HLA-A31:01	0.89
	HLA-A33:01	0.79
	HLA-B07:02	0.9
	HLA-B08:01	0.89
	HLA-C07:01	0.75
	HLA-C07:02	0.54
PD42	HLA-A02:06	0.56
	HLA-A24:02	0.88
	HLA-B07:02	0.91
	HLA-B51:01	0.82
	HLA-C01:02	0.53
	HLA-C07:02	0.48
RA957	HLA-A02:20	0.64
	HLA-A68:01	0.83
	HLA-B35:03	0.87
	HLA-B39:01	0.89
	HLA-C04:01	0.72
	HLA-C07:02	0.12
RPMI8226	HLA-A30:01	0.66
	HLA-A68:02	0.76
	HLA-B15:03	0.77
	HLA-B51:32	0.91
	HLA-C02:10	0.6
	HLA-C03:04	0.72
SK-MEL-5	HLA-A02:01	0.81
	HLA-A11:01	0.85
	HLA-B40:01	0.86
	HLA-C03:03	0.57
T98G	HLA-A02:01	0.74
	HLA-B39:06	0.65
	HLA-C07:02	0.43
THP-1	HLA-A02:01	0.81
	HLA-A24:02	0.83
	HLA-B15:11	0.63
	HLA-B35:01	0.86
	HLA-C03:03	0.76
TIL1	HLA-A02:01	0.78
	HLA-B08:01	0.81
	HLA-B08:01	0.91
	HLA-C05:01	0.53
TIL3	HLA-A01:01	0.89
	HLA-A23:01	0.8
	HLA-B07:02	0.89
	HLA-B15:01	0.85
	HLA-C12:03	0.81
	HLA-C14:02	0.88
U-87	HLA-A02:01	0.76
	HLA-B44:02	0.82
	HLA-C05:01	0.85
pat-AC2	HLA-A02:01	0.79
	HLA-A24:02	0.79
	HLA-B27:05	0.78
	HLA-B45:01	0.89
pat-C	HLA-A02:01	0.51
	HLA-A03:01	0.58
	HLA-B07:02	0.78
	HLA-C07:02	0.65
pat-CELG	HLA-A02:01	0.6
	HLA-A24:02	0.82
	HLA-B15:01	0.8
	HLA-B70:01	0.7
	HLA-C03:02	0.61
	HLA-C15:05	0.61
pat-CP2	HLA-A11:01	0.87
	HLA-B14:02	0.9
	HLA-B44:02	0.92
pat-FL	HLA-A03:01	0.77
	HLA-A11:01	0.83
	HLA-B44:02	0.88
	HLA-B50:01	0.78
pat-J	HLA-A02:01	0.84
	HLA-A03:01	0.7
	HLA-B07:02	0.79
	HLA-C07:02	0.51
pat-JP83	HLA-A02:01	0.85
	HLA-A11:01	0.82
	HLA-B27:05	0.83
	HLA-B56:01	0.72
pat-JT2	HLA-A11:01	0.89
	HLA-B58:01	0.76
	HLA-B35:01	0.92
pat-M	HLA-A03:01	0.63
	HLA-A24:02	0.81
	HLA-B08:01	0.88
	HLA-B44:03	0.86
	HLA-C07:01	0.84
	HLA-C16:01	0.58
pat-MA	HLA-A02:01	0.72
	HLA-A29:02	0.84
	HLA-B44:03	0.92
	HLA-B57:01	0.89
	HLA-C07:01	0.53
	HLA-C16:01	0.82
pat-ML	HLA-A02:01	0.7
	HLA-A11:01	0.77
	HLA-B40:01	0.86
	HLA-B44:03	0.93
pat-MS2	HLA-A02:01	0.87
	HLA-B13:02	0.76
	HLA-B41:01	0.71
pat-NT	HLA-A01:01	0.76
	HLA-A32:01	0.75
pat-PF1	HLA-A01:01	0.71
	HLA-A02:01	0.72
	HLA-B07:02	0.89
	HLA-B44:03	0.92
	HLA-C07:02	0.53
	HLA-C16:01	0.85
pat-R	HLA-A03:01	0.77
	HLA-A29:02	0.81
	HLA-B08:01	0.85
	HLA-B44:03	0.86
	HLA-C07:01	0.83
	HLA-C16:01	0.48
pat-RT	HLA-A01:01	0.89
	HLA-A02:01	0.71
	HLA-B08:01	0.85
	HLA-B39:24	0.79
	HLA-C05:01	0.53
	HLA-C07:01	0.65
pat-SK	HLA-A02:01	0.72
	HLA-A23:01	0.8
	HLA-B08:01	0.85
	HLA-B44:03	0.91
pat-ST	HLA-A03:01	0.82
	HLA-A24:02	0.86
	HLA-B07:02	0.89
	HLA-B27:05	0.84

Table 3.S7. Predicted Positive Predictive Values (PPV) for all clusters associated with each allele in each cell line the multi allele (MA) dataset after NNAlign_MA deconvolution. For details on the calculation of PPV, refer to materials and methods.

Cell line ID	HLA-A	HLA-B	HLA-C	Source PMID(s)
HL-60	47	851	102	26992070 17083564
CA46	71	739	191	26992070
Mel-624	931	48	21	7541714
HEK293	896	80	24	26258424

Table 3.S8. Cell lines with atypical HLA-A or HLA-B peptidome repertoire profiles. For each cell line, the relative peptidome size for the HLA-A, HLA-B and HLA-C loci is given. Peptidome sizes were calculated as described in Figure 3.4-B. The last column shows the references to earlier publications describing the loss of the locus for a given cell line.

	SA MODEL	MA MODEL
HLA-A*02:01	376	778
HLA-A*02:05	533	771
HLA-A*02:07	477	871
HLA-A*03:01	359	896
HLA-A*11:01	361	862
HLA-A*26:01	838	962
HLA-A*29:01	762	873
HLA-A*31:01	409	749
HLA-A*32:01	808	905

Table 3.S9. AUC0.1 performance values for the SA molecules left out from the training of the A2 and A3 specificity reduced SA and MA models. Note that not all the molecules included in the evaluation are part of the A2 and A3 supertypes; these molecules are included because they have a distance to the A2 and A3 molecules in the MA dataset less than 0.1.

Table 3.S10. Due to its dimensions, this table is not embedded in this manuscript. Please refer to the online supplementary material to access it.

Chapter 4

Upgrading the NetMHCpan suite with NNAlign_MA

4.1 Summary

This chapter presents the paper "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", in which the NNAlign_MA algorithm is deployed as the new engine of the (up to August 2021) newest version of the NetMHCpan suite.

The aforementioned suite consists of the NetMHCpan and NetMHCIIpan softwares, each one in charge of predicting binding to any MHC-I and MHC-II of known sequence, respectively. Both methods are trained using a joint, extensive dataset of BA, EL SA and EL MA sequences under the semi-supervised guidance of NNAlign_MA. This results in an overall state-of-the-art performance, but also the capacity of outperforming their competitors in the task of predicting eluted ligands and epitopes. Thanks to this, the newest NetMHCpan suite represents a highly valuable asset for rational epitope discovery.

After training and independent validation, NetMHCpan and NetMHCIIpan were uploaded to the internet as free web-servers for the scientific community. The work of this chapter also describes the available features these web interfaces have to facilitate user operation.

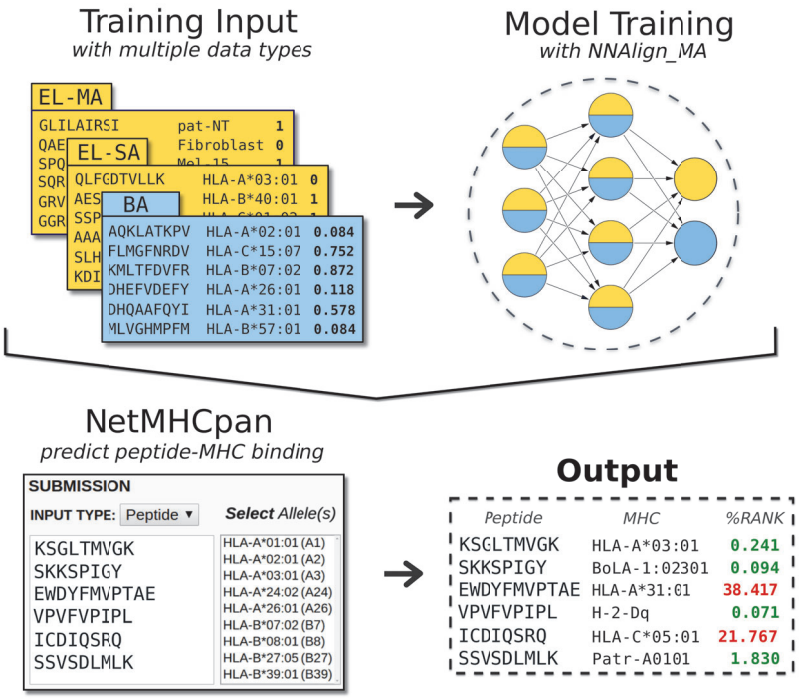

Figure 4.1. Graphical abstract of chapter four. In the upper left panel, three EL MA, EL SA and BA training datasets (consisting of peptides, MHC or cell line restrictions, and target values) are shown; such datasets are then jointly fed to the NNAlign_MA neural network framework in order to train it. The lower left panel shows a job submission box for the NetMHCpan web-servers, consisting of a list of query peptides and a selection of MHC restrictions to run the predictions against; after completing the job, the servers report the corresponding output (bottom right panel), where peptide-MHC pairs can be observed together with their normalized binding scores.

4.2 Paper III

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, July 2020, Volume 48, Issue W1,
https://doi.org/10.1186/s13073-018-0594-6

Birkir Reynisson[1,†], Bruno Alvarez[2,†], Sinu Paul[3], Bjoern Peters[3,4] and Morten Nielsen[1,2,⋆]

[1]Department of Bio and Health Informatics, Technical University of Denmark, Kgs. Lyngby, DK 28002, Denmark
[2]Instituto de Investigaciones Biotecnológicas, Universidad Nacional de San Martín, CP1650 San Martín, Argentina
[3]Division of Vaccine Discovery, La Jolla Institute for Allergy and Immunology, 9420 Athena Circle, La Jolla, CA 92037, USA
[4]Department of Medicine, University of California, San Diego, CA 92093, USA

⋆ Corresponding author (morni@dtu.dk)
† These authors contributed equally to the paper as first authors

Abstract

Major histocompatibility complex (MHC) molecules are expressed on the cell surface, where they present peptides to T cells, which gives them a key role in the development of T-cell immune responses. MHC molecules come in two main variants: MHC Class I (MHC-I) and MHC Class II (MHC-II). MHC-I predominantly present peptides derived from intracellular proteins, whereas MHC-II predominantly presents peptides from extracellular proteins. In both cases, the binding between MHC and antigenic peptides is the most selective step in the antigen presentation pathway. Therefore, the prediction of peptide binding to MHC is a powerful utility to predict the possible specificity of a T-cell immune response. Commonly MHC binding prediction tools are trained on binding affinity or mass spectrometryeluted ligands. Recent studies have however demonstrated how the integration of both data types can boost predictive performances. Inspired by this, we here present NetMHCpan-4.1 and NetMHCIIpan-4.0, two web servers created to predict binding between peptides and MHC-I and MHC-II, respectively. Both methods exploit tailored machine learning strategies to integrate different training data types, resulting in state-of-the-art performance and outperforming their competitors. The servers are available at `http://www.cbs.dtu.dk/services/NetMHCpan-4.1/` and `http://www.cbs.dtu.dk/services/NetMHCIIpan-4.0/`.

Introduction

The Major histocompatibility complex (MHC) is a fundamental cell surface protein of the cellular immune system of vertebrates. The primary function of MHC is to bind to peptides (small protein fragments) derived from the digestion of intracellular or extracellular proteins and display them to the intercellular space. If T cells recognize and bind to a peptide–MHC complex, an immune response can be triggered and the compromised cell will undergo lysis. Given this, the binding of antigenic peptides to MHC molecules represents a necessary step for cellular immunity, and understanding the rules of this event has large and valuable potential in human health applications.

MHC comes in two main variants: MHC Class I (MHCI) and MHC Class II (MHC-II). MHC-I binds peptides from intracellular proteins after these undergo proteasomal degradation, and serves as a control mechanism for antigenic variations in the self-peptidome repertoire. On the other hand, the MHC-II binds peptides generated by protease-digestion of extracellular proteins; with this, both MHC systems can exert control over foreign organisms via the presentation of non-self proteins to T cells [1]. In view of this fact, important efforts have been committed to developing computational methods capable of accurately predicting peptide binding to both MHC-I and MHC-II (reviewed in [247]).

Different types of experimental data have been used to train these methods. According to the nature of such training data, we can classify peptide-MHC binding predictors in three main categories. The first category corresponds to predictors trained on binding affinity (BA) data [196, 248–250]. This type of data imposes a substantial limitation on prediction performances, since it only models the single event of peptide-MHC binding, and neglects any other biological feature involved in the process. The second category covers methods that are either trained with data retrieved from mass spectrometry (MS) experiments, known as eluted ligands (EL) [53, 169, 170, 244, 245], or trained integrating both BA and EL data [162, 215, 220, 249, 251]. This latter data type incorporates information not only related to the peptide-MHC binding event, but also information about prior steps in the biological antigen presentation pathway processes. However, except for genetically engineered cells, cellular MHC expression profile is very diverse due to the multiple MHC allelic variants. Also, antibodies employed to purify peptide–MHC complexes in MS EL pipelines are mostly pan- or locus-specific, leading to inherently poly-specific (or Multi Allelic, MA) data (i.e., the data contains peptides matching multiple cognate MHC binding motifs). Thus, a prior, user biased peptide-MHC annotation criteria are, in general, needed in order to interpret such EL MA data, transform them to Single Allelic (EL SA, or single peptide-MHC annotations) and employ them for the training of MHC-specific binding predictors [167].

The third and last category of algorithms seeks to resolve this limitation of the second type of models, and incorporates, together with the training of a prediction algorithm, the capability of annotating EL MA sequences to single MHC restrictions [188, 252]. One such method is termed NNAlign_MA [252], which during the training process can cluster EL sequences with ambiguous cognate MHCs into single MHC specificities, using a strategy called pseudolabeling. This enables not only the possibility of novel motif discovery, but also a considerable expansion of the training set size, and therefore an overall improvement of the method's predictive power.

In this work, we deploy NNAlign_MA to update NetMHCpan and NetMHCIIpan, augmenting their training capabilities and also increasing their predictive performance. We do this by incorporating NNAlign_MA to the core of the new models, allowing us to expand their training sets greatly. Moving further, we perform a full independent epitope evaluation on both models and show how the updated methods outperform other current state-of-the-art algorithms.

The NNAlign_MA machine learning framework

The updated versions of NetMHCpan and NetMHCIIpan differ from their predecessors in two critical aspects: the training data and the machine-learning modeling framework. The training data have been vastly extended by accumulating MHC BA and EL data from the public domain. In particular, EL data were extended to include MA data. The combined dataset used for training of NetMHCpan4.1 consists of 13 245 212 data points covering 250 distinct MHC class I molecules, and the combined dataset used for training of NetMHCIIpan-4.0 consists of 4 086 230 data points covering a total of 116 distinct MHC class II molecules. For specific details on the training sets and data partitioning refer to Supplementary Material. The machine learning framework was updated from NNAlign to NNAlign_MA to allow for effective handling of these MA data. In short, the NNAlign framework is a singleallele framework permitting the integration of mixed data types (BA and EL) in the model training, which allows information to be leveraged across the different data types, resulting in a boosted predictive power [162, 215]. NNAlign_MA extends this training framework to allow for the incorporation of EL MA data. This is achieved by iteratively annotating the best single-allele to the MA data during the model training, effectively deconvoluting the MA binding motifs [252]. For specific details on the model hyper-parameters and cross-validation training performance, please refer to Supplementary Material.

Web Interface

Submission Page

Input Data

Both servers accept two different types of input; FASTA and PEPTIDE. The input data can be directly pasted into a submission box or uploaded from the user's local disk. For FASTA input, the user can specify the peptide length(s) to be included in the predictions (for class I, the length range goes from 8 to 14 amino acids, default is 8–11; for class II only one length is admitted with 15 being the default value). Also, for Class II, one can specify if CONTEXT encoding [215] is to be used. This context consists of amino acids spanning the source protein N and C terminal parts of the ligand. The submission page includes examples of input data for all accepted formats and provides buttons to upload sample data automatically.

MHC selection

Next, the servers provide a drop-down menu in order to select which MHC family and molecule(s) to be used. NetMHCpan-4.1 covers more than 11 000 MHC molecules, spanning human (HLA-A, HLA-B, HLA-C, HLA-E, HLA-G), mouse (H-2), cattle (BoLA), primates (Patr, Mamu, Gogo), swine (SLA), equine (EQCA) and dog (DLA), and NetMHCIIpan-4.0 covers a total of close to 1000 human (HLA-DR, HLA-DQ, HLA-DP) and mouse (H-2) MHC alleles. For DQ and DP, the user can make combinations of the covered alpha and beta protein chains. Furthermore, given the pan-specific nature of both methods, predictions can be run for any MHC molecule of known sequence by uploading a full-length MHC protein sequence in FASTA format.

Additional configuration

Both NetMHCpan methods inform if a sequence is a strong MHC binder (SB) or a weak MHC binder (WB) based on a %Rank score. Briefly, %Rank is a transformation that normalizes prediction scores across different MHC molecules and enables interspecific MHC binding prediction comparisons. %Rank of a query sequence is computed by comparing its prediction score to a distribution of prediction scores for the MHC in question, estimated from a set of random natural peptides. Given this, a %Rank value of 1% means that a queried sequence obtained a prediction score that corresponds to the top 1% scores obtained from random natural peptides. The %Rank values for detecting SBs and WBs can be modified by specifying the corresponding thresholds (by default, %Rank $< 0.5\%$ and %Rank $< 2\%$ thresholds are considered for detecting SBs and WBs for class I and %Rank $< 2\%$ and %Rank $< 10\%$, for SBs and WBs for class II). In addition, an option is available to only report sequences with a lower than a defined %Rank threshold, and for class II to print only the strongest binding peptide overlapping a given binding core if FASTA was selected as the input format.

Additionally, the user may opt to get the BA prediction scores of input sequences together with the EL likelihood, and to sort the output according to the corresponding EL predicted values (from

A)

Pos	MHC	Peptide	Core	Of	Gp	Gl	Ip	Il	Icore	Identity	Score_EL	%Rank_EL	Score_BA	%Rank_BA	Aff(nM)	BindLevel
1	HLA-A*30:01	ASQKRPSQR	ASQKRPSQR	0	0	0	0	0	ASQKRPSQR	seq1	0.3038680	0.569	0.316257	5.143	1632.63	<= WB
2	HLA-A*30:01	SQKRPSQRH	SQKRPSQRH	0	0	0	0	0	SQKRPSQRH	seq1	0.1472270	1.533	0.203325	13.611	5540.54	<= WB
3	HLA-A*30:01	QKRPSQRHG	QKRPSQRHG	0	0	0	0	0	QKRPSQRHG	seq1	0.0063890	15.486	0.116401	32.313	14190.74	
4	HLA-A*30:01	KRPSQRHGS	KRPSQRHGS	0	0	0	0	0	KRPSQRHGS	seq1	0.0050730	17.438	0.108557	35.232	15447.71	
5	HLA-A*30:01	RPSQRHGSK	RPSQRHGSK	0	0	0	0	0	RPSQRHGSK	seq1	0.0560270	3.810	0.280215	6.920	2411.28	
6	HLA-A*30:01	PSQRHGSKY	PSQRHGSKY	0	0	0	0	0	PSQRHGSKY	seq1	0.0028600	22.985	0.085228	45.997	19883.19	
7	HLA-A*30:01	SQRHGSKYL	SQRHGSKYL	0	0	0	0	0	SQRHGSKYL	seq1	0.3023670	0.573	0.513405	0.975	193.42	<= WB
8	HLA-A*30:01	QRHGSKYLA	QRHGSKYLA	0	0	0	0	0	QRHGSKYLA	seq1	0.0188000	8.324	0.166771	19.205	8228.45	
9	HLA-A*30:01	RHGSKYLAT	RHGSKYLAT	0	0	0	0	0	RHGSKYLAT	seq1	0.0038720	19.911	0.121768	30.487	13390.16	
10	HLA-A*30:01	HGSKYLATA	HGSKYLATA	0	0	0	0	0	HGSKYLATA	seq1	0.0284610	6.304	0.325222	4.800	1481.70	

B)

Pos	MHC	Peptide	Of	Core	Core_Rel	Identity	Score_EL	%Rank_EL	Exp_Bind	Score_BA	Affinity(nM)	%Rank_BA	BindLevel
8	DRB1_0434	QRHGSKYLATASTMD	6	YLATASTMD	0.860	seq1	0.109816	21.94	NA	0.540059	144.96	9.47	
9	DRB1_0434	RHGSKYLATASTMDH	5	YLATASTMD	0.953	seq1	0.397085	4.68	NA	0.603262	73.16	3.77	<=WB
10	DRB1_0434	HGSKYLATASTMDHA	4	YLATASTMD	0.953	seq1	0.542934	2.17	NA	0.639784	49.28	1.89	<=WB
11	DRB1_0434	GSKYLATASTMDHAR	3	YLATASTMD	0.947	seq1	0.661655	1.02	NA	0.666855	36.77	1.06	<=SB
12	DRB1_0434	SKYLATASTMDHARH	2	YLATASTMD	0.807	seq1	0.464566	3.32	NA	0.663527	38.11	1.14	<=WB
13	DRB1_0434	KYLATASTMDHARHG	1	YLATASTMD	0.620	seq1	0.156700	16.28	NA	0.625281	57.65	2.51	
14	DRB1_0434	YLATASTMDHARHGF	5	STMDHARHG	0.447	seq1	0.021961	51.57	NA	0.498187	228.04	15.59	
15	DRB1_0434	LATASTMDHARHGFL	4	STMDHARHG	0.827	seq1	0.016294	57.25	NA	0.397562	677.39	38.51	
16	DRB1_0434	ATASTMDHARHGFLP	3	STMDHARHG	0.820	seq1	0.025460	48.63	NA	0.364505	968.66	47.62	
17	DRB1_0434	TASTMDHARHGFLPR	2	STMDHARHG	0.680	seq1	0.010663	64.90	NA	0.363900	975.02	47.78	
18	DRB1_0434	ASTMDHARHGFLPRH	3	MDHARHGFL	0.640	seq1	0.007536	71.02	NA	0.354401	1080.56	50.46	

Figure 4.2. Example outputs for the NetMHCpan-4.1 and NetMHCIIpan-4.0 tools. (A) Example output for NetMHCpan-4.1, using as input the web server'sFASTA sample data and the HLA-A*30:01 allele, with a peptide length of nine and other options set to default. (B) Example output for NetMHCIIpan4.0, using as input the web server's FASTA sample data and the DRB1*04:34 allele, with all other options set to default. By default, prediction scores are for both methods displayed in terms of a Score EL (the likelihood of a peptide being an MHC ligand) column and a '%Rank EL' column (the EL percentile Rank score); if the user selects to include BA predictions, such values are reported as well. The 'BindLevel' column displays the presence of Strong Binders (SB) or Weak Binders (WB) amongst the queried peptides. 'Peptide' informs the list of peptides that have been interrogated against the selected MHC molecule(s) (exhibited in the 'MHC' column). The 'Pos' entry refers to the queried peptide's position in the selected FASTA input, and 'Core' refers to such peptide's identified binding core. 'Identity' is an automatically generated ID that is assigned to the input. Other columns refer to specific properties that depend on the MHC class being employed. For additional details on the interpretation of the different columns of the output, refer to the 'output format' page on both web servers homepages.

high to low). In addition, and for user convenience, the possibility to save the output as a *.XLS file (readable to most spreadsheet software) is also provided.

Output Page

The output from both servers details the binding prediction values of the provided input sequence(s) for the selected MHC molecule(s), together with additional information to guide the interpretation of results. As seen in Figure 4.2, NetMHCpan and NetMHCIIpan output consist of several plain text columns, which exhibit different pieces of information regarding the prediction outcome.

Evaluation and Examples

As independent validations, the models were benchmarked on sets of T-cell epitope data and for class I also EL SA data. For MHC class I the epitope dataset was taken from Jurtz et. al [162] combined with a comprehensive set of MHC multimer validated epitopes obtained from the IEDB and for MHC class II from Reynisson et al. [253]. The EL SA data were obtained from [254]. In all cases, the data were filtered to ensure no overlap with the training data (for further details on the data sets refer to Supplementary Material). For the epitope data, the predictive performance was estimated in terms of FRANK [162]. That is, for each epitope-HLA pair, binding to the HLA was predicted for all overlapping peptides of the source protein using the eluted ligand likelihood prediction score and the FRANK value was reported as the proportion of peptides with a prediction score higher than that of the epitope. Using this measure, a value of 0 corresponds to a perfect prediction (the known epitope is identified with the highest predicted binding value among all peptides found within the source protein), while a value of 0.5 corresponds to a random prediction. Further, was the corresponding AUC for each epitope reported, again assigning all overlapping peptides in the source protein except the epitope as negatives. For further details on the CD8 epitope benchmark, refer to Supplementary Table 4.S7. For the EL SA dataset, negative decoy peptides were added as described in the "Training and Test data" section of the Supplementary Material in 'Materials and Methods' and the performance evaluated in terms of AUC, AUC0.1 and PPV. Here, PPV was estimated from the fraction of positive peptides within the top N predictions, where N is equal to

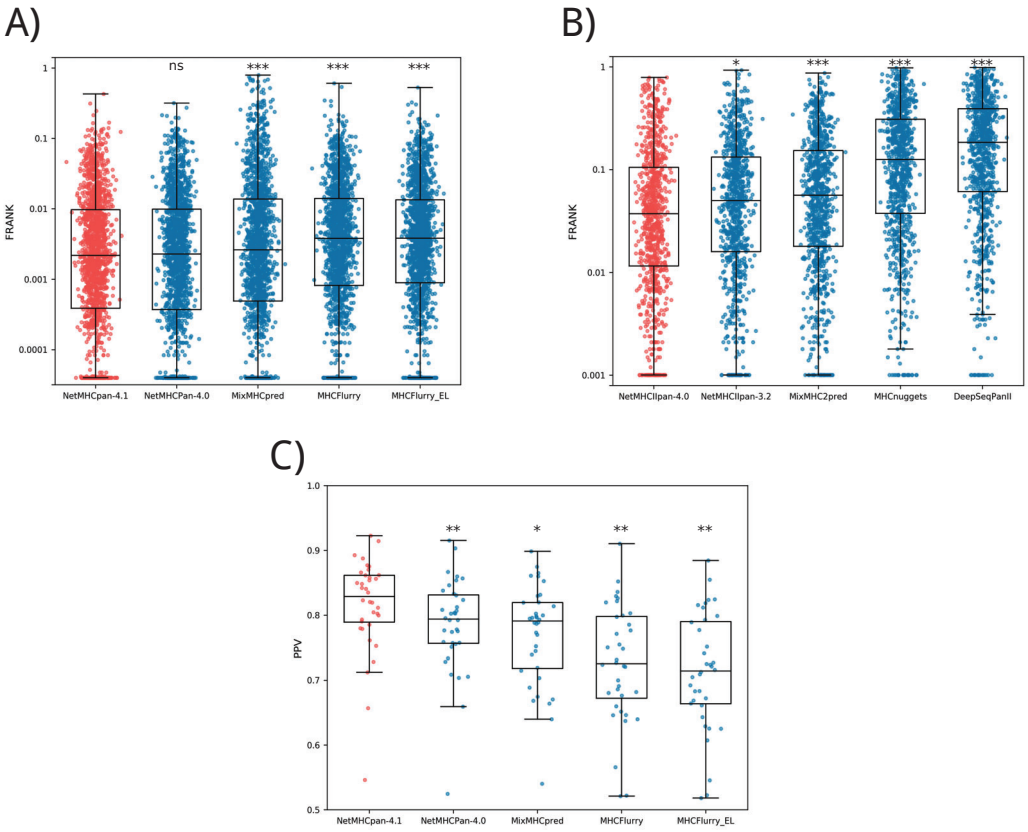

Figure 4.3. Epitope benchmark results for the NetMHCpan-4.1 and NetMHCIIpan-4.0 web servers. (A) Performance results for the CD8+ epitope benchmark. Median FRANK values for the different methods are: NetMHCpan-4.1, 0.00220; NetMHCpan-4.0, 0.00230; MixMHCpred, 0.00264; MHCFlurry, 0.00383; and MHCFlurry EL, 0.00386. (B) FRANK performance results for the CD4+ epitope benchmark. The median FRANK for the different methods are: NetMHCIIpan-4.0, 0.0351; NetMHCIIpan-3.2, 0.04825; MixMH2Cpred, 0.0513; MHCnuggets, 0.1219; and DeepSeqPanII, 0.1767. (C) PPV performance results for the MS MHC class I eluted ligand benchmark. Median PPV values for the different methods are: NetMHCpan-4.1, 0.8291; NetMHCpan-4.0, 0.7940; MixMHCpred, 0.7911; MHCFlurry, 0.7256; and MHCFlurry EL, 0.7144. P-values are shown as * P $<$ 0.05, ** P $<$ 10−6 and *** P $<$ 10−9. All p-values were calculatated using a two-tailed binomial test. The plotted boxes extend from the lower to upper quartile values of the data (25th to 75th percentile), with a line at the median; whiskers extend from the box to show the range of the data to the most extreme, non-outlier data points.

the total number of ligands times 0.95 (to account for potential MS contaminants). For additional information on the EL SA benchmark, refer to Supplementary Table 4.S8.

The results of these benchmarks are shown in Figure 4.3. Here, NetMHCpan-4.1 was compared to NetMHCpan4.0 [162], MixMHCpred [188,211], MHCFlurry [249] and MHCFlurry EL (an unpublished version of MHCFLurry trained with EL SA data, available at GitHub [255]). For this benchmark, because MixMHCpred cannot make predictions for peptides containing 'X' (wildcard amino acid symbol), such peptides were removed from the benchmark dataset. NetMHCIIpan-4.0 was compared in a similar manner to NetMHCIIpan-3.2 [163], MixMHC2pred [245], MHCnuggets [256] and DeepSeqPanII [257].

With the exception of NetMHCpan-4.1 and NetMHCpan-4.0 when tested on the epitope benchmark, all three benchmarks confirmed a significantly superior performance of NetMHCpan-4.1 and NetMHCIIpan-4.0 over all other methods included in the respective benchmarks. For the class I epitope benchmark, NetMHCpan-4.1 and NetMHCpan-4.0 were found to share comparable predictive performance. For NetMHCpan-4.1 a consistent improvement was found for HLA-B and HLA-C molecules for both the epitopes and ligand benchmarks when compared to NetMHCpan-4.0 (consistent with the very large increased coverage of these loci by the EL dataset used for the training of NetMHCpan-4.1). Note, also that in contrast to what was observed when evaluating the performance on eluted ligand data [253], but in line with earlier works [215, 253, 258], a

drop in the performance of NetMHCIIpan-4.0 was observed when including context information (Supplementary Figure 4.S6).

Discussion

Over the last years, large amounts of novel MS-eluted MHC ligand data have become available, enabling a highly enriched characterization of the MHC-presented ligandome. Here, we have benefitted from this data, and combining it with an extensive set of MHC peptide-binding data available in the IEDB, have developed updated versions of the NetMHCpan and NetMHCIIpan tools. Both methods are capable of predicting a peptide's likelihood of antigen presentation (and BA) to MHC class I and class II molecules. Both tools were trained using the NNAlign_MA machine learning framework, which enables the integration of MS ligand datasets obtained from cell lines expressing multiple MHC alleles. The benchmarking of these methods against other available state-of-the-art algorithms exhibited a significantly improved predictive power for the prediction of MHC ligands and T-cell epitopes.

For both NetMHCpan-4.1 and NetMHCIIpan-4.0, the performance gain was found most pronounced for prediction of MS identified MHC ligands. This in particular for class I, where the NetMHCpan-4.1 method on the epitope benchmark was found to perform at par with its most recent ancestor NetMHCpan-4.0. Many possible reasons for this limited impact on the performance for epitope prediction exists, including biases in the epitope data currently available toward past prediction methods and in-vitro experimental validation techniques, and biases in the MS EL data not shared with T-cell epitopes. Future work will resolve the impact and importance of these biases, and allow us to access to what degree the improved power for prediction of MS MHC ligands translates into an improved power also for prediction of T-cell epitopes.

Benchmark evaluation of the tools demonstrated an overall robust power of the NNAlign MA machine learning framework to perform motif deconvolution across all MHC molecules included in the training data. However, results also pointed to a lower performance for MHC molecules characterized by limited ligand datasets such as HLA-C and HLA-DQ. While this low number of ligands annotated to MHC from these two loci in part can be explained from their relative low protein expression, other causes could include differences in the HLA-loci specificities of the antibodies used for immunoprecipitation (IP) when purifying MHC molecules prior to running MS experiments. Future work may tell if working with antibodies with improved HLA-DQ specificities or using engineered cell lines with, for instance, tagged HLA molecules as suggested by [53] can help resolve this.

Even though one of the main contributions to the improved performance of the prediction methods proposed here (and other recently published methods) is the integration of MS derived EL data, MS data itself contains an inherent bias imposed resulting in for instance overrepresentation of 'flyable' [259] and neglecting cysteine-containing peptides [170]. These biases impose limitations on the set of ligands detectable in MS and hence subsequent limitations on the learned binding motifs. Given this, further complementary technological platforms for high throughput detection of MHC peptide interactions might be warranted to complete our understanding of HLA antigen presentation.

Both NetMHCpan and NetMHCIIpan have an easy to use user interface, allowing for simple uploads of query sequence data, and a selection of MHC alleles to be interrogated for binding. As the only current publicly available tools, both methods demonstrate a truly pan-specific capability, allowing users to make predictions for all MHC molecules, including those not previously characterized by binding data. The output from the tools is provided in simple text format with guided information, aiding the user to select relevant epitope/MHC-ligand candidates.

Given the demonstrated high performances and their ease of use, we expect the updated web servers to become relevant tools to guide future rational epitope discovery projects.

Supplementary Material

Training and Test data

Both NetMHCpan-4.1 and NetMHCIIpan-4.0 were trained using data from multiple sources, according to the type of MHC system being modeled. The assembled training sets for these systems consisted of two main data types [162]: Binding Affinity (BA, peptides derived from in-vitro Peptide-MHC binding assays) and Eluted Ligands (EL, peptides derived from Mass Spectrometry experiments). Additionally, EL data is composed of two subtypes: Single-Allele (SA, peptides assigned to single MHCs) and Multi-Allele (MA, peptides with multiple MHC options to be assigned). For more information on these types of data, refer to Alvarez et al. [252].

BA data is, in essence, real-valued and transformed to fall in the [0,1] interval, as described earlier [154]. On the other hand, EL data is binary, meaning that positive instances (both SA and MA) were labeled with a target value of 1 and negatives with a target value of 0.

NetMHCpan-4.1
EL MA training data for this method were extracted from Alvarez et al [252], Bulik-Sullivan et al. [244] and one in-house dataset. EL SA data were collected from Alvarez et al. [252], the IEDB [225] and DeVette et al. [216]; BA data was gathered from Alvarez. et al [252] and the IEDB [225]. An overview of the full training set is presented in Supplementary Table 4.S1.

SA (BA)			SA (EL)			MA (EL)		
Positives	Negatives	#MHCs	Positives	Negatives	#MHCs	Positives	Negatives	#MHCs
54,402	155,691	170	218,962	3,813,877	142	446,53	8,395,021	112

Table 4.S1. NetMHCpan-4.1 training data overview. Columns correspond to each type of training data employed in this work, for which the number of positive and negative training instances is displayed, together with the total amount of unique MHCs. BA: Binding Affinity; EL: Eluted Ligands; SA: Single Allele; MA: Multi Allele. A threshold of 500 nM was used to define positive BA data points.

All peptides employed in the training were filtered to only include 8 to 14 amino acid long peptides. All MHCs present in the BA subset were enriched with 100 random negative sequences (target value of 0.01). On the other hand, positive peptides for each MHC present in the EL subset were enriched, length-wise, with 5 times the amount of peptides of the most abundant peptide length, as described earlier [225]. Random peptides were extracted from the UniProt database.

For independent performance evaluation, a test set of HLA restricted CD8+ epitopes was constructed. This data set consists of the epitope data set from Jurtz et al. [162] combined with multimer validated epitopes obtained from the IEDB (downloaded 11-04-2020). The data set was filtered to only contain epitopes of length 8-14, mapped to fully typed HLA molecules covered by all methods included in the benchmark, and annotated source protein sequence. To remove potential noise in the data, all epitopes with a minimal Frank value across all methods included in the benchmark greater than 0.1 were excluded. Further, additional SA EL datasets were downloaded from [254]. Each dataset SA was enriched, length-wise, with negative decoy peptides of 5 times the amount of ligands of the most abundant peptide length. Also, here were the datasets limited to HLA molecules covered by all methods included in the benchmark. Finally, to ensure the independent test set's orthogonality, positive peptides overlapping with the training data were removed from all test sets. The resulting benchmark datasets consisted of 1,660 epitopes restricted to 52 distinct MHC-I molecules, and 36 SA EL datasets covering a total 45,416 MS MHC eluted ligands.

NetMHCIIpan-4.0
Training data was gathered from Reynisson et. al. [253], which is composed of data from the IEDB [225], 16 publically available datasets [53, 185, 202, 226–232, 245, 260–264] and one in-house data set. BA data was extracted from [163]. Out of the 17 EL datasets, 5 contained exclusively SA data [226,230,231,260,263], 6 contained exclusively MA [227,229,232,245,261,262] and 6 contained a mixture of MA and SA data ([53, 185, 202, 228, 264] and the in-house dataset). Only cell lines with more than 250 measured ligands were included in the final dataset. A summary of the data is provided in Supplementary Table 4.S2.

All data was filtered to only include peptides of length 13-21. Each EL (SA or MA) data set was enriched with random negative peptides as described for NetMHCpan-4.0 by adding negative decoy peptides 5 times the amount of the most represented positive length for each length.

SA (BA)			SA (EL)			MA (EL)		
Positives	Negatives	#MHCs	Positives	Negatives	#MHCs	Positives	Negatives	#MHCs
44,861	64,098	79	66,307	586,118	19	314,759	3,119,046	114

Table 4.S2. NetMHCIIpan-4.0 training data overview. Columns correspond to each type of training data employed in this work, for which the number of positive and negative training instances is displayed, together with the total amount of unique MHCs. BA: Binding Affinity; EL: Eluted Ligands; SA: Single Allele; MA: Multi Allele. A threshold of 500 nM was used to define positive BA data points.

An independent test set was generated from HLA restricted CD4+ epitopes from the IEDB. MHC II Epitopes measured by 'ICS', 'intracellular staining', 'multimer/tetramer' assays were extracted from a set of all T-cell assays from the IEDB (downloaded 27-11-2019) and filtered to include only peptides of length 13-21, removing peptides with unconventional amino acids and post-translationally modified peptides. To ensure orthogonality, all training set peptides that shared a 9 residue motif with this epitope set were removed. Further, and to remove potential noise in the data, all epitopes with a minimal Frank value across all methods included in the benchmark greater than 0.1 were excluded. The final CD4 epitope benchmark contained 917 epitopes restricted to 20 different MHC-II molecules.

All evaluation data sets are available from http://www.cbs.dtu.dk/services/NetMHCpan-4.1.

Neural Network Architectures and Hyperparameters

Both NetMHCpan-4.1 and NetMHCIIpan-4.0 were constructed upon the NNAlign_MA [252] machine learning modeling framework. NNAlign_MA is a neural network method based on NNAlign [159], with the extended capability of deconvoluting MHC binding motifs of Mass Spectrometry derived Immunopeptidomics datasets. Both models were trained with similar hyperparameters as described previously [159, 215, 220, 252] for datasets containing BA, SA EL and MA EL sequences (for more details on the training data, refer to Supplementary Tables 4.S1 and 4.S2). Essentially, the neural network architecture for both models is a Feed Forward Network, with an input layer, a single hidden layer and an output layer with two output neurons (one for binding affinity and other for eluted ligand likelihood). Networks were trained using back-propagation with stochastic gradient descent and a fixed learning rate of 0.05. An ensemble of networks was created for each model according to the amount of chosen hyperparameters (see below). When making predictions using the ensembles, the average over the individual network predictions was used as the final prediction score.

NetMHCpan-4.1
A total of 10 random seeds for weight initialization were used; the hidden layer was populated with 55 and 66 hidden neurons; and the training data was split into 5 partitions for cross-validation using a Hobohm1-based common motif algorithm [156] with a motif length of 8 amino acids. This yielded a final ensemble of 50 networks. All networks in the ensemble were trained using 200 iterations, with a burn-in period of 20 iterations and early stopping.

NetMHCIIpan-4.0
An ensemble of models was trained in a 5-fold cross-validation manner using the common motif algorithm [156] with motif length of 9 residues for splitting the data, with 20, 40 and 60 neurons in the hidden layer, each with 10 seeds for weight initialization. The resulted in a final ensemble of 150 networks, each trained for 400 iterations, with a burn-in period of 20 iterations and no early stopping.

Training Performance Evaluation

Beyond the performance evaluation on the independent T cell epitope benchmarks included in this work, both methods were further evaluated from their cross-validated performance. This evaluation included cross-validated AUC, AUC0.1 (AUC integrated up to a False Positive Rate of 10%), PCC (Pearson Correlation Coefficient), PPV (Positive Predictive Value) and several measures for motif deconvolution consistency. The details of these different performance measures are described in earlier publications [252, 253].

NetMHCpan-4.1
AUC, AUC0.1, and PCC values for the BA (binding affinity) data, SA (single allele EL) data, and MA (multi allele EL) data are included in Supplementary Tables 4.S3, 4.S4, 4.S5, and 4.S6. Here, only MHC molecules/datasets characterized with at least 10 data points and at least 2 binders and

2 non-binders are included. The performance on the SA and MA datasets was found to be overall comparable (median AUC equal to 0.99 and 0.98 for the SA and MA datasets respectively). The binding motif devolutions for the MA data are shown in Supplementary Figure 4.S4. In this plot motifs are represented as sequence logos generated using Seq2Logo [29], generated from the ligands assigned to the MHC molecule in each cell line, excluding ligands with a presented percentile rank score of 20% or higher. Only motifs for molecules characterized by at least 10 ligands are shown. These figures demonstrated the ability of NetMHCpan to identify motifs with, in the vast majority of cases, well-defined anchor positions for the MHC molecules in each dataset. Only datasets and MHC molecules characterized by few deconvoluted ligands share more noisy motifs (exemplified by, for instance, the HLA-C*07:01 motif in the Line.34 dataset, and HLA-C*06:02 motif in the Line.41 dataset). Furthermore, a correlation analysis of the deconvoluted motifs for individual MHC molecules characterized by at least 50 ligands and shared between multiple MA datasets revealed a high motif consistency (see Supplementary Figure 4.S5). Here, the average/median correlation for MHC molecules shared between 3 or more data sets was 0.89/0.90. Further, PPV values for each motif deconvolution in the MA data were found to be generally high (and comparable to the PPV values obtained from the SA data) with a median of 0.82 (the median for SA data was 0.89). PPV was here calculated as the proportion of true-positive predictions within the top N predictions for each allele in each data set, where N is the number of positive ligands deconvoluted to the given MHC molecule with a predicted percentile rank score of 20% of less (20 rank is used to allow for a small proportion of false positive MS ligands). These PPV values should be compared to the expected value of a random predictor of 0.06 (estimated from the number of ligands divided by the total number of peptides assigned to each MHC molecule). Moreover, we found that examples of deconvolutions with reduced PPV values, in the vast majority of cases, correspond to HLA-C motifs deconvoluted with very few ligand examples. By way of example, 70% of the MHC alleles with a PPV of 0.5 or less correspond to HLA-C molecules, and 85% of these examples are characterized by 100 or fewer ligands.

NetMHCIIpan-4.0

The cross-validated performance evaluation of NetMHCIIpan-4.0 -similar to that shown above for NetMHCpan-4.1- is reported in [253]. In such work, conclusions supported that also for MHC II can the NNAlign_MA framework successfully deconvolute MA data, thus extending training data substantially both in terms of MHC molecules and ligands. This was supported by comparing cross-validation AUCs for models trained on MA data (and SA data) to models trained only on SA data. Significant improvements were observed for the model trained on the extended datasets, indicating successful integration of MA data. Further, and in line with earlier publications [215], the work confirmed an improved performance for predicting MS ligands for models trained integrating information about the ligand context from the source protein sequence. Quantification of deconvolution consistency by motif correlation (as described for MHC I above) showed high consistency scores for most MHC molecules across loci. HLA-DQ was however, found to be an exception from this, with fewer molecules characterized by consistent and accurate binding motifs. HLA-DQ molecules also performed generally worse than HLA-DR and HLA-DP in the PPV analyses, where PPV was computed for each cell line MHC deconvolution after positive ligands had been filtered to include only ligands with predicted percentile rank score less then 20. This thresholding was applied to account for noise in EL data, leaving the average ligand/negative ratio as 0.119, which is the exptexted PPV of a random predictor. Median PPV values for loci HLA-DR, DP, DQ and Mouse H-2 were 0.824, 0.738, 0.558 and 0.865, respectively. For a conservative threshold of PPV 0.5 the predictor deconvoluted accurate motifs for 34 HLA-DR, 9 HLA-DP, 11 HLA-DQ and 3 H-3 molecules.

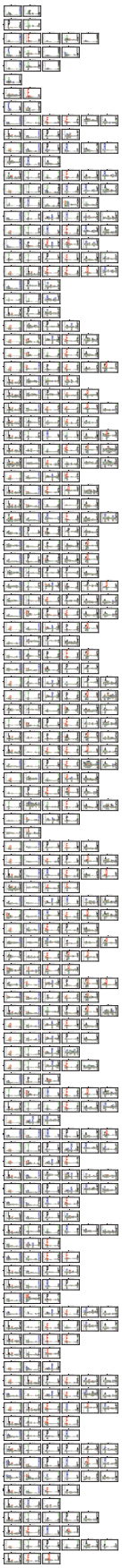

Figure 4.S4. Full motif deconvolution for the Eluted Ligand (EL) Multi Allele (MA) data used to train NetMHCpan-4.1. Each row corresponds to a cell line present in the training data (114 in total; for more details, refer to Supplementary Table 4.S5). Using cross-validation, ligands are assigned to the single most likely MHC molecule expressed by a given cell line. Using this assignment, binding motifs are then generated for each allele in each cell line. To remove potential Mass-Spectrometry related contaminants, only ligands with a Rank score lower than 20 are included. The number of sequences associated to the corresponding MHC is displayed on top of each logo. For more information regarding the performance of this deconvolution, refer to Supplementary Table 4.S6.

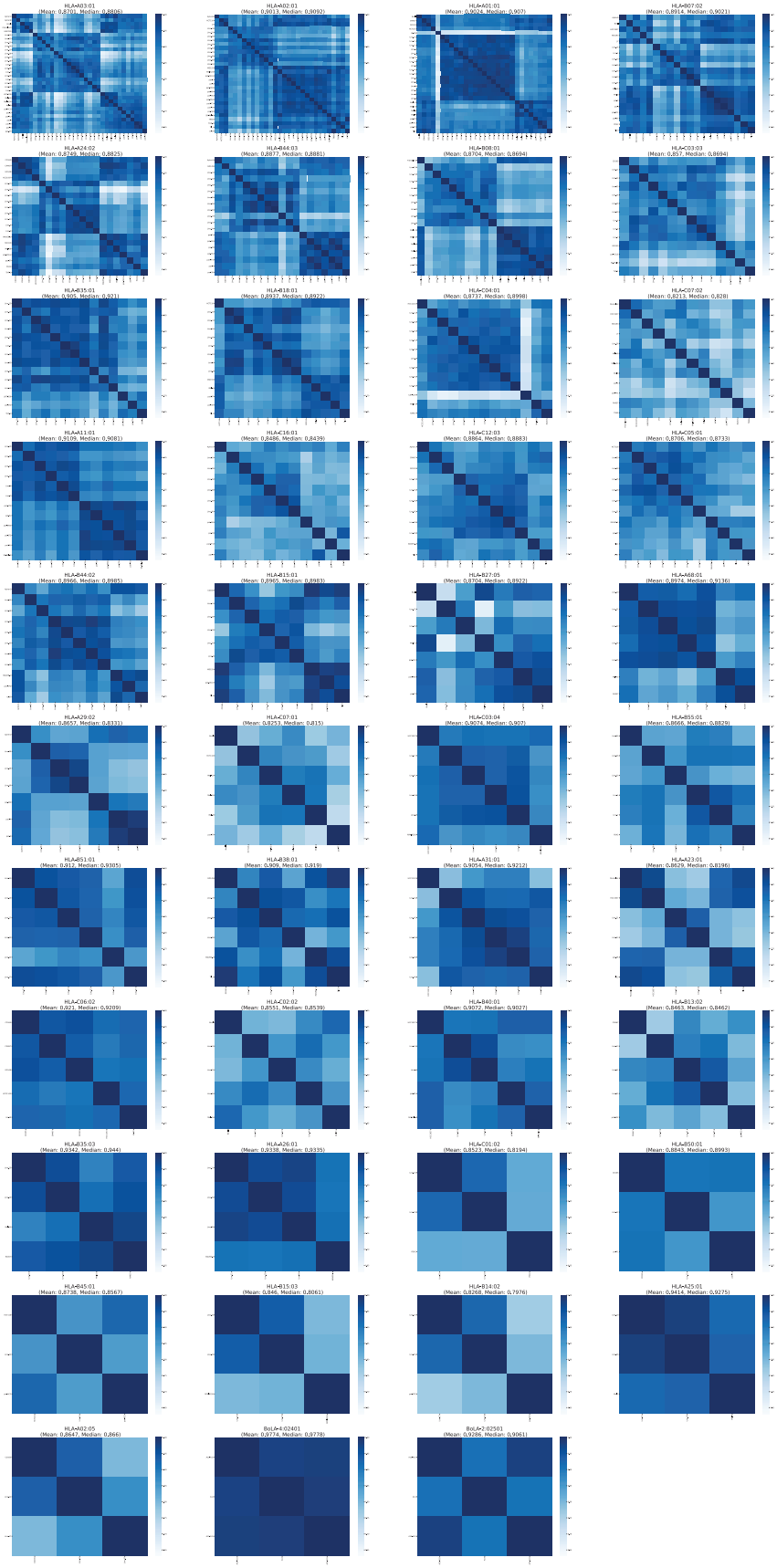

Figure 4.S5. Correlation matrices between deconvoluted MHC motifs across all the Eluted Ligand (EL) Multi Allele (MA) cell lines employed in the training of NetMHCpan-4.1 Each matrix corresponds to an MHC that is shared between three or more cell lines and has more than 50 assigned sequences in the motif deconvolution of the cell line data. To remove potential Mass-Spectrometry related contaminants, only ligands with a Rank lower than 20 are included. For a given MHC, each matrix entry displays the Pearson Correlation Coefficient (PCC) between two motifs for the MHC obtained from the two cell line data sets. For a given matrix, the corresponding MHC allele and the mean/median PCCs are displayed on top.

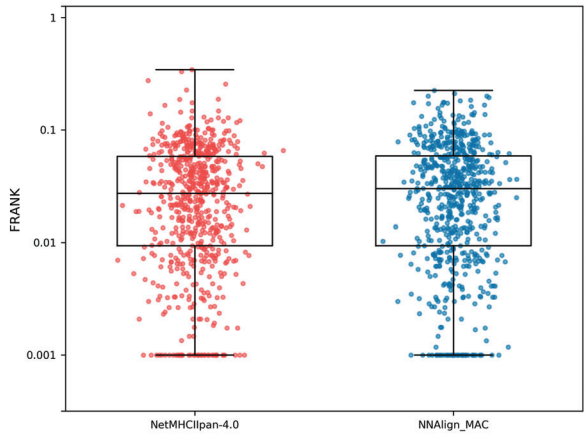

Figure 4.S6. CD4 Epitope benchmark results for NetMHCIIpan-4.0 and the corresponding version trained including ligand context ("NNAlign_MAC"). For NetMHCIIpan-4.0, FRANK values are calculated as described in the text. For NNAlign_MAC, to account for the fact that epitopes are measured from synthetic peptides (that might not reflect length preferences and signatures of antigen processing), another FRANK scoring scheme was implemented, similar to what has been described before [258]. Briefly, the prediction score for each peptide was assigned from the sum of prediction scores of all 13-17-mers with a binding core overlapping the original sequence. NetMHCIIpan-4.0 obtained a significantly lower median F-rank score compared to the context encoding model (2.732 and 3.004, respectively, p-value<0.005 in a binomial test, excluding ties).

MHC	# Peptides	# Positives	# Negatives	AUC	AUC0.1	PCC
BoLA-1:02301	258	182	76	0.80473	0.15856	0.62819
BoLA-2:00801	14	11	3	0.81818	0.45455	0.57290
BoLA-2:01201	177	50	127	0.86394	0.35167	0.70783
BoLA-3:00101	166	8	158	0.82753	0.28333	0.45304
BoLA-3:00201	158	32	126	0.66642	0.16667	0.35007
BoLA-6:01301	269	220	49	0.79889	0.41477	0.58428
BoLA-6:04101	157	38	119	0.81557	0.40431	0.62725
BoLA-T2C	90	84	6	0.85913	0.63095	0.53832
Eqca-100101	678	109	569	0.91461	0.55357	0.76101
Gogo-B0101	15	6	9	0.77778	0.33333	0.66770
H-2-Db	3925	834	3091	0.93235	0.65438	0.83963
H-2-Dd	414	48	366	0.90380	0.72280	0.76855
H-2-Kb	4107	1436	2671	0.91186	0.59947	0.82763
H-2-Kd	905	306	599	0.84330	0.42689	0.67805
H-2-Kk	390	177	213	0.85552	0.47269	0.73447
H-2-Ld	338	121	217	0.95129	0.61590	0.83715
HLA-A01:01	4883	734	4149	0.95378	0.70521	0.82188
HLA-A02:01	12853	4814	8039	0.95092	0.67143	0.86276
HLA-A02:02	4178	1869	2309	0.92074	0.59904	0.83911
HLA-A02:03	6323	2424	3899	0.94549	0.70085	0.86937
HLA-A02:05	121	87	34	0.93442	0.61686	0.87755
HLA-A02:06	5651	2278	3373	0.91175	0.58375	0.81731
HLA-A02:07	81	22	59	0.97612	0.82727	0.90400
HLA-A02:11	1085	404	681	0.95456	0.72186	0.87438
HLA-A02:12	1183	308	875	0.96744	0.80225	0.89320
HLA-A02:16	921	181	740	0.96982	0.79752	0.87274
HLA-A02:17	346	176	170	0.80364	0.38402	0.62691
HLA-A02:19	1246	234	1012	0.98307	0.84357	0.87605
HLA-A02:50	135	88	47	0.98549	0.82955	0.92562
HLA-A03:01	7481	2180	5301	0.93842	0.61631	0.83130
HLA-A03:02	26	13	13	0.63314	0.23077	0.37828
HLA-A03:19	30	14	16	0.90625	0.64286	0.68729
HLA-A11:01	6478	2226	4252	0.95197	0.66884	0.85984
HLA-A11:02	14	10	4	0.90000	0.80000	0.86414
HLA-A23:01	2632	612	2020	0.92827	0.53288	0.80608
HLA-A24:02	3363	753	2610	0.92061	0.51999	0.79013
HLA-A24:03	1381	332	1049	0.97138	0.85279	0.89527
HLA-A25:01	960	71	889	0.99468	0.94622	0.84119
HLA-A26:01	4535	516	4019	0.94501	0.70269	0.81338
HLA-A26:02	643	168	475	0.97445	0.83118	0.89063
HLA-A26:03	538	62	476	0.96730	0.74399	0.81097
HLA-A29:02	2788	726	2062	0.89525	0.54127	0.77588
HLA-A30:01	2926	776	2150	0.93394	0.68716	0.83397
HLA-A30:02	2105	644	1461	0.84999	0.38926	0.68255
HLA-A31:01	5706	1616	4090	0.92998	0.58377	0.81573
HLA-A32:01	1131	410	721	0.92979	0.56426	0.82555
HLA-A32:07	88	79	9	0.71027	0.39241	0.48760
HLA-A32:15	74	59	15	0.74350	0.22034	0.58807
HLA-A33:01	3589	712	2877	0.93335	0.63362	0.80795
HLA-A66:01	220	13	207	0.94314	0.53462	0.51757
HLA-A68:01	3825	1605	2220	0.92525	0.60779	0.83388
HLA-A68:02	5580	1351	4229	0.93940	0.67028	0.83235
HLA-A68:23	82	76	6	0.80263	0.55263	0.50075
HLA-A69:01	2571	249	2322	0.95708	0.73549	0.80405
HLA-A74:01	15	6	9	0.88889	0.00000	0.68405
HLA-A80:01	1173	122	1051	0.96615	0.74934	0.80247
HLA-B07:02	4849	1401	3448	0.95606	0.68289	0.85885
HLA-B08:01	3544	803	2741	0.93814	0.63730	0.80965
HLA-B08:02	1020	36	984	0.99077	0.90731	0.82444
HLA-B08:03	470	18	452	0.95612	0.71728	0.71464
HLA-B14:01	42	17	25	0.77176	0.29412	0.55578
HLA-B14:02	343	43	300	0.91450	0.66202	0.70112
HLA-B15:01	4456	1368	3088	0.93583	0.61544	0.79366
HLA-B15:02	165	124	41	0.83969	0.31653	0.66016
HLA-B15:03	696	384	312	0.93899	0.72236	0.87179
HLA-B15:09	832	45	787	0.96366	0.71681	0.76993
HLA-B15:17	1448	379	1069	0.96235	0.79489	0.87184
HLA-B15:42	364	3	361	0.93167	0.31481	0.17949
HLA-B18:01	2831	310	2521	0.91207	0.61686	0.76070
HLA-B27:05	3747	689	3058	0.94741	0.63879	0.83027
HLA-B35:01	3304	985	2319	0.93380	0.61158	0.82984
HLA-B35:03	278	19	259	0.98069	0.83158	0.87915
HLA-B37:01	66	9	57	0.91813	0.42222	0.68058
HLA-B38:01	531	159	372	0.98406	0.87591	0.89810
HLA-B39:01	1814	304	1510	0.96888	0.77608	0.83283
HLA-B39:06	33	21	12	0.84921	0.52381	0.73497
HLA-B40:01	3439	638	2801	0.96575	0.74914	0.86269
HLA-B40:02	1088	342	746	0.92894	0.53062	0.84119
HLA-B40:13	59	53	6	0.77673	0.66038	0.46828
HLA-B42:01	232	63	169	0.93303	0.76290	0.87096
HLA-B42:02	18	14	4	0.60714	0.14286	0.52671
HLA-B44:02	2337	447	1890	0.95769	0.70149	0.81415
HLA-B44:03	1480	426	1054	0.93865	0.64058	0.85082
HLA-B45:01	1069	192	877	0.93555	0.64494	0.84890
HLA-B45:06	362	4	358	0.75419	0.28571	0.10311
HLA-B46:01	1815	97	1718	0.97139	0.76216	0.73946
HLA-B48:01	883	73	810	0.98519	0.85591	0.84447
HLA-B51:01	2923	313	2610	0.93702	0.60571	0.72146
HLA-B52:01	24	7	17	0.96639	0.71429	0.75269
HLA-B53:01	1818	501	1317	0.92946	0.60828	0.80515
HLA-B54:01	1206	231	975	0.93998	0.66350	0.83464
HLA-B57:01	3000	493	2507	0.95627	0.77234	0.83912
HLA-B57:02	18	14	4	0.94643	0.78571	0.85617
HLA-B57:03	34	27	7	0.98942	0.92593	0.82883
HLA-B58:01	3381	763	2618	0.95816	0.76915	0.87261
HLA-B58:02	68	13	55	0.67832	0.18462	0.42597
HLA-B73:01	129	17	112	0.91071	0.36898	0.71238
HLA-B81:01	26	13	13	1.00000	1.00000	0.92549
HLA-B83:01	339	40	299	0.96204	0.80086	0.78621
HLA-C03:03	165	121	44	0.89350	0.36570	0.77979
HLA-C04:01	552	23	529	0.88962	0.52007	0.39890
HLA-C05:01	173	68	105	0.95210	0.76912	0.84496
HLA-C06:02	408	101	307	0.93640	0.65050	0.82932
HLA-C07:01	261	101	160	0.82494	0.40965	0.65143
HLA-C07:02	143	84	59	0.89205	0.50238	0.64032
HLA-C08:02	87	32	55	0.80511	0.26250	0.60157
HLA-C12:03	172	155	17	0.65996	0.10323	0.38532
HLA-C14:02	259	187	72	0.84314	0.37433	0.71743
HLA-C15:02	253	116	137	0.90454	0.54841	0.75145
HLA-E01:01	96	14	82	0.85714	0.65179	0.74865
HLA-E01:03	67	5	62	0.80323	0.53333	0.56546
Mamu-A1*00101	2590	1114	1476	0.90512	0.59231	0.79740
Mamu-A1*00201	1359	584	775	0.89498	0.53482	0.79891
Mamu-A1*00701	681	215	466	0.97340	0.84591	0.89059
Mamu-A1*01101	1269	405	864	0.90263	0.55515	0.77621
Mamu-A1*02201	689	215	474	0.95557	0.81108	0.90769
Mamu-A1*02601	206	51	155	0.77356	0.39085	0.57513
Mamu-A2*00102	146	30	116	0.87098	0.65455	0.79842
Mamu-A7*00103	102	22	80	0.79318	0.34659	0.56888
Mamu-B*00101	880	83	797	0.95087	0.82919	0.87765
Mamu-B*00301	1145	309	836	0.93320	0.64355	0.84922
Mamu-B*00801	1127	305	822	0.94603	0.69060	0.86936
Mamu-B*01001	159	56	103	0.85160	0.47679	0.70986
Mamu-B*01701	1650	437	1213	0.92759	0.59069	0.84066
Mamu-B*01704	1360	365	995	0.91150	0.51335	0.81255
Mamu-B*03901	456	241	215	0.85857	0.61549	0.67131
Mamu-B*05201	892	547	345	0.92137	0.68131	0.82437
Mamu-B*06502	17	4	13	0.86538	0.25000	0.68186
Mamu-B*06601	101	73	28	0.81115	0.12329	0.62895
Mamu-B*08301	432	219	213	0.91389	0.52881	0.82681
Mamu-B*08701	155	71	84	0.84423	0.37148	0.67994
Patr-A01:01	434	108	326	0.87435	0.39091	0.75151
Patr-A03:01	374	59	315	0.93968	0.67961	0.79542
Patr-A04:01	327	74	253	0.89408	0.42378	0.79753
Patr-A07:01	591	132	459	0.84046	0.36481	0.67350
Patr-A09:01	655	282	373	0.83993	0.50968	0.75295
Patr-B01:01	784	190	594	0.92936	0.60277	0.83881
Patr-B13:01	216	131	85	0.91953	0.63359	0.84441
Patr-B24:01	324	84	240	0.92133	0.66022	0.81440
SLA-104:01	211	142	69	0.84823	0.40141	0.61838
SLA-1107:01	23	14	9	0.76984	0.64286	0.13509
SLA-1107:02	39	30	9	0.60000	0.10000	0.23229
SLA-204:01	130	51	79	0.60660	0.16246	0.08269
SLA-304:01	76	29	47	0.84446	0.18103	0.63609

Table 4.S3. Cross-Validation training performance for the Binding Affinity (BA) data used to trainNetMHCpan-4.1. "# Peptides" refers to the total amount of peptides present for a given MHC ("MHC" column); "# Positives" and "# Negatives" represent the quantity of positive and negative peptides (defined using a threshold of 500 nM), respectively; "AUC" = Area Under the ROC Curve; "AUC0.1" = Area Under the ROC Curve integrated up to a False Positive Rate of 10%; "PCC" = Pearson Correlation Coefficient.

MHC	# Peptides	# Positives	# Negatives	AUC	AUC0.1	PPV
DLA-8803401	1891	316	1575	0.98972	0.90295	0.89333
DLA-8850101	34285	1654	32631	0.97480	0.87100	0.81477
DLA-8850801	409	49	360	0.97500	0.75567	0.78261
H-2-Db	97739	4355	93384	0.99457	0.97316	0.93449
H-2-Dd	23	4	19	1.00000	1.00000	1.00000
H-2-Dq	77165	7547	69618	0.98292	0.87464	0.84698
H-2-Kb	98211	5444	92767	0.94670	0.86721	0.84626
H-2-Kd	21358	772	20586	0.94796	0.88564	0.87449
H-2-Kk	60	10	50	0.97400	0.74000	0.77778
H-2-Kq	4656	404	4252	0.82158	0.55804	0.58747
H-2-Ld	79	9	70	0.98889	0.88889	0.87500
HLA-A01:01	84815	7156	77659	0.99259	0.93937	0.89850
HLA-A01:03	82	7	75	1.00000	1.00000	1.00000
HLA-A02:01	265231	13025	252206	0.97418	0.86266	0.79868
HLA-A02:03	44403	1873	42530	0.99526	0.96660	0.90444
HLA-A02:04	84455	3155	81300	0.98195	0.88547	0.80848
HLA-A02:05	4474	300	4174	0.98664	0.89633	0.84912
HLA-A02:07	107160	4040	103120	0.99343	0.96109	0.90750
HLA-A02:14	99	9	90	0.98519	0.85185	0.75000
HLA-A03:01	80210	4403	75807	0.99318	0.95089	0.89646
HLA-A03:02	197	17	180	0.98725	0.87255	0.81250
HLA-A11:01	53828	2794	51034	0.99434	0.97141	0.93142
HLA-A23:01	2638	121	2517	0.99883	0.98828	0.96491
HLA-A24:02	118725	6007	112718	0.99230	0.94659	0.88959
HLA-A24:06	4341	356	3985	0.95812	0.87112	0.87574
HLA-A24:13	5268	237	5031	0.97266	0.89218	0.88000
HLA-A26:01	3683	155	3528	0.99768	0.97674	0.92517
HLA-A29:02	181129	7342	173787	0.98905	0.94635	0.87669
HLA-A30:01	266	21	245	0.83421	0.29365	0.31579
HLA-A30:02	217	22	195	0.99277	0.92584	0.90000
HLA-A30:03	79	4	75	0.99333	0.92857	1.00000
HLA-A30:04	50	5	45	0.99111	0.90000	0.75000
HLA-A31:01	27537	1922	25615	0.98285	0.88153	0.83343
HLA-A32:01	13667	607	13060	0.99785	0.98214	0.94271
HLA-A66:01	382	44	338	0.99361	0.93664	0.90244
HLA-A66:02	377	27	350	0.98275	0.86032	0.80000
HLA-A68:01	243	21	222	1.00000	1.00000	1.00000
HLA-A68:02	38125	1984	36141	0.93372	0.79015	0.76911
HLA-A69:01	12	2	10	1.00000	1.00000	1.00000
HLA-B07:02	201053	11256	189797	0.99268	0.94858	0.90648
HLA-B08:01	86467	4350	82117	0.98460	0.91397	0.85649
HLA-B13:01	972	74	898	0.93630	0.73110	0.71429
HLA-B13:02	743	43	700	0.99821	0.98206	0.95000
HLA-B14:02	5949	279	5670	0.99843	0.98425	0.93962
HLA-B14:03	260	20	240	0.99104	0.91042	0.84211
HLA-B15:01	145368	7365	138003	0.99280	0.95323	0.89208
HLA-B15:02	47971	2614	45357	0.98457	0.89809	0.81837
HLA-B15:03	34	4	30	1.00000	1.00000	1.00000
HLA-B15:08	153	18	135	0.99095	0.90598	0.82353
HLA-B15:09	100	10	90	1.00000	1.00000	1.00000
HLA-B15:10	276	17	259	0.99955	0.99529	0.93750
HLA-B15:11	3345	129	3216	0.98219	0.90410	0.86066
HLA-B15:13	36	6	30	1.00000	1.00000	1.00000
HLA-B15:16	313	13	300	0.92769	0.82308	0.75000
HLA-B15:17	77	13	64	0.97716	0.75641	0.83333
HLA-B15:18	35	5	30	1.00000	1.00000	1.00000
HLA-B15:42	30	5	25	0.93600	0.20000	0.50000
HLA-B18:01	22988	1255	21733	0.99705	0.98308	0.96644
HLA-B18:03	2289	212	2077	0.99861	0.98605	0.94527
HLA-B27:01	62975	3887	59088	0.99714	0.97723	0.93283
HLA-B27:02	48532	3339	45193	0.99573	0.97316	0.94861
HLA-B27:03	17350	1028	16322	0.99631	0.97456	0.93955
HLA-B27:04	28502	1316	27186	0.99302	0.97078	0.93440
HLA-B27:05	102127	4960	97167	0.98270	0.92837	0.89304
HLA-B27:06	27657	1233	26424	0.99496	0.97294	0.93083
HLA-B27:07	41143	2146	38997	0.99623	0.97587	0.93719
HLA-B27:08	35936	2162	33774	0.99558	0.96465	0.91330
HLA-B27:09	93383	4794	88589	0.98026	0.92422	0.88669
HLA-B27:10	130	10	120	1.00000	1.00000	1.00000
HLA-B35:01	41260	2033	39227	0.94131	0.85197	0.84205
HLA-B35:02	727	37	690	0.98927	0.91892	0.91429
HLA-B35:03	4092	196	3896	0.98899	0.92571	0.89785
HLA-B35:04	132	12	120	0.87847	0.46528	0.45455
HLA-B35:06	115	10	105	0.85619	0.58000	0.66667
HLA-B35:08	3843	207	3636	0.91321	0.71088	0.72449
HLA-B37:01	397	41	356	0.98931	0.90732	0.92105
HLA-B38:01	134	9	125	0.92622	0.78704	0.75000
HLA-B39:01	20810	973	19837	0.99872	0.98714	0.94589
HLA-B39:05	47	7	40	0.98571	0.85714	0.83333
HLA-B39:06	7567	512	7055	0.98799	0.90455	0.83333
HLA-B39:09	95	15	80	0.99750	0.97500	0.92857
HLA-B39:10	56	6	50	0.98333	0.83333	0.80000
HLA-B39:24	3454	260	3194	0.99458	0.94577	0.88259
HLA-B40:01	49871	2803	47068	0.99727	0.97661	0.93201
HLA-B40:02	145809	7596	138213	0.99542	0.96688	0.90909
HLA-B41:01	10157	458	9699	0.99413	0.97208	0.95862
HLA-B41:02	157	17	140	0.98445	0.89496	0.87500
HLA-B41:03	1057	83	974	0.92339	0.74960	0.75641
HLA-B41:04	505	50	455	0.95301	0.66533	0.70213
HLA-B41:05	331	32	299	0.81114	0.55496	0.63333
HLA-B41:06	334	19	315	0.87034	0.54669	0.55556
HLA-B42:01	130	10	120	0.99750	0.97500	1.00000
HLA-B44:02	79191	4908	74283	0.99576	0.97111	0.95260
HLA-B44:03	79380	4523	74857	0.99836	0.98923	0.96113
HLA-B44:05	34	4	30	1.00000	1.00000	1.00000
HLA-B44:08	772	48	724	0.99577	0.95747	0.88889
HLA-B44:09	2337	170	2167	0.95368	0.83322	0.85093
HLA-B44:27	2583	205	2378	0.99818	0.98179	0.96392
HLA-B44:28	1092	119	973	0.99737	0.97358	0.95575
HLA-B45:01	16025	680	15345	0.99166	0.93812	0.91022
HLA-B46:01	47705	1918	45787	0.98484	0.92991	0.84907
HLA-B47:01	366	26	340	0.98699	0.86991	0.79167
HLA-B49:01	4148	203	3945	0.99815	0.98151	0.91667
HLA-B50:01	11209	579	10630	0.99885	0.98845	0.95455
HLA-B50:02	12	2	10	1.00000	1.00000	1.00000
HLA-B51:01	75166	3792	71374	0.98119	0.90564	0.87257
HLA-B51:02	66	7	59	1.00000	1.00000	1.00000
HLA-B51:08	12754	624	12130	0.99830	0.98303	0.93243
HLA-B52:01	48	8	40	0.97500	0.78125	0.85714
HLA-B54:01	21893	1092	20801	0.98862	0.94005	0.88910
HLA-B55:01	12	2	10	1.00000	1.00000	1.00000
HLA-B55:02	30	5	25	1.00000	1.00000	1.00000
HLA-B56:01	14534	723	13811	0.99883	0.98830	0.95627
HLA-B57:01	184807	15505	169302	0.98108	0.91194	0.88764
HLA-B57:02	98	8	90	1.00000	1.00000	1.00000
HLA-B57:03	76418	4804	71614	0.98700	0.92037	0.87092
HLA-B58:01	65867	3916	61951	0.98844	0.94109	0.90108
HLA-B58:02	47	7	40	0.98214	0.82143	0.83333
HLA-B73:01	3295	178	3117	0.99922	0.99221	0.94675
HLA-C01:02	41473	1624	39849	0.98977	0.95807	0.91440
HLA-C02:02	31877	1916	29961	0.99194	0.92953	0.87857
HLA-C03:03	31071	1110	29961	0.99329	0.96675	0.91177
HLA-C03:04	59103	2162	56941	0.99690	0.97425	0.91573
HLA-C04:01	46055	2008	44047	0.99337	0.96817	0.93865
HLA-C05:01	95872	4533	91339	0.98366	0.93355	0.89039
HLA-C06:02	57239	2125	55114	0.93453	0.83491	0.81665
HLA-C07:01	6775	407	6368	0.98871	0.91640	0.85492
HLA-C07:02	28828	1252	27576	0.95817	0.87562	0.83684
HLA-C07:04	12	2	10	0.80000	0.00000	0.00000
HLA-C08:02	47927	3458	44469	0.99823	0.98232	0.94521
HLA-C12:02	78	8	70	0.97321	0.73214	0.85714
HLA-C12:03	24501	1388	23113	0.99478	0.95415	0.90440
HLA-C12:04	18	3	15	1.00000	1.00000	1.00000
HLA-C14:02	34464	2441	32023	0.99854	0.98539	0.94651
HLA-C15:02	34035	1873	32162	0.99555	0.95658	0.89713
HLA-C16:01	42635	2970	39665	0.99236	0.93842	0.89436
HLA-C17:01	8585	602	7983	0.98394	0.87288	0.82487
Mamu-B*00801	17535	851	16684	0.97741	0.91342	0.89109

Table 4.S4. Cross-Validation training performance for the Eluted Ligand (EL) Single Allele (SA) data used to train NetMHCpan-4.1. "# Peptides" refers to the total amount of peptides present for a given MHC ("MHC" column); "# Positives" and "# Negatives" represent the quantity of positive and negative peptides, respectively; "AUC" = Area Under the ROC Curve; "AUC0.1" = Area Under the ROC Curve integrated up to a False Positive Rate of 10%; "PPV" = Positive Predictive Value. For details on how PPV is calculated refer to the manuscript text.

Cell Line	# Peptides	# Positives	# Negatives	AUC	AUC0.1
A10	158826	10188	148638	0.97429	0.85328
A11-A11	141942	7403	134539	0.96556	0.87381
A12-A15	282968	11872	271096	0.97800	0.90023
A14	195380	9509	185871	0.97862	0.89380
A15-A15	292756	12433	280323	0.97670	0.90066
A18	86723	6615	80108	0.98743	0.93720
A19-A19	163032	9582	153450	0.96641	0.88588
A20-A20	251551	11726	239825	0.97866	0.90984
Apher1	129377	6145	123232	0.95430	0.87363
Apher6	41725	1962	39763	0.98065	0.90534
Bcell	232902	12199	220703	0.98188	0.90757
CA46	64924	2324	62600	0.98102	0.92858
CD165	138113	5364	132749	0.98107	0.90699
CM467	191850	7401	184449	0.98592	0.91757
EBL	277396	12915	264481	0.94843	0.81612
Fibroblast	127318	5289	122029	0.97092	0.89343
GD149	217941	9756	208185	0.98240	0.93313
HCC1143	72274	2780	69494	0.97922	0.91435
HCC1937	107220	4976	102244	0.97807	0.92332
HCT116	97309	4174	93135	0.97774	0.91400
HEK293	91543	4972	86571	0.98562	0.92658
HL-60	122202	6607	115595	0.99153	0.94754
JY	63674	2868	60806	0.98250	0.93753
Line.1	6053	467	5586	0.99142	0.96166
Line.10	25670	2140	23530	0.98575	0.93651
Line.11	61813	4720	57093	0.99287	0.94592
Line.12	29002	2199	26803	0.99190	0.94943
Line.13	7636	486	7150	0.97440	0.82162
Line.14	3080	208	2872	0.99402	0.94233
Line.15	12640	921	11719	0.99081	0.94284
Line.16	5860	369	5491	0.99187	0.94570
Line.17	31746	1823	29923	0.96786	0.89025
Line.18	23223	1602	21621	0.98406	0.91819
Line.19	4191	318	3873	0.97294	0.85376
Line.2	11078	896	10182	0.99231	0.95531
Line.20	47348	3680	43668	0.98550	0.91385
Line.21	19898	1394	18504	0.97378	0.86347
Line.22	38131	2372	35759	0.97245	0.85578
Line.23	18442	1366	17076	0.98815	0.92964
Line.24	9393	570	8823	0.98745	0.93783
Line.25	47776	3338	44438	0.98654	0.91317
Line.26	5218	366	4852	0.97721	0.89726
Line.27	8596	575	8021	0.98524	0.90342
Line.28	11632	796	10836	0.98977	0.93602
Line.29	45243	3268	41975	0.98478	0.88924
Line.3	37308	2716	34592	0.98684	0.92107
Line.30	5345	571	4774	0.94481	0.74236
Line.31	41110	2833	38277	0.98730	0.91935
Line.32	8961	908	8053	0.98512	0.89561
Line.33	19918	1282	18636	0.99099	0.93769
Line.34	67373	4861	62512	0.98436	0.92253
Line.35	7803	482	7321	0.99263	0.94049
Line.36	60197	4517	55680	0.98758	0.93595
Line.37	14535	943	13592	0.98215	0.89406
Line.38	47064	3685	43379	0.98654	0.92198
Line.39	38260	2714	35546	0.98324	0.90925
Line.4	24355	1684	22671	0.98356	0.91488
Line.40	45493	3164	42329	0.98821	0.93689
Line.41	25526	1691	23835	0.98081	0.88058
Line.42	9104	662	8442	0.97949	0.91049
Line.43	1477	100	1377	0.98262	0.87314
Line.44	25489	1909	23580	0.98998	0.94100
Line.45	35979	2647	33332	0.98700	0.91891
Line.46	54443	3489	50954	0.98778	0.91856
Line.47	3242	230	3012	0.98487	0.88466
Line.48	44981	3138	41843	0.97845	0.87460
Line.49	23169	1797	21372	0.97639	0.88152
Line.5	29156	2345	26811	0.98974	0.93441
Line.50	5209	315	4894	0.99154	0.93473
Line.51	10097	750	9347	0.99001	0.95275
Line.52	3439	244	3195	0.98964	0.94830
Line.53	33054	2278	30776	0.98052	0.88637
Line.54	7859	651	7208	0.99203	0.94208
Line.55	40559	2784	37775	0.97562	0.87127
Line.6	24908	1916	22992	0.99045	0.94381
Line.7	14151	1077	13074	0.99026	0.94481
Line.8	25449	1818	23631	0.99183	0.95109
Line.9	30369	2461	27908	0.99076	0.95432
LNT-229	188049	10311	177738	0.93425	0.79206
MAVER-1	179017	7403	171614	0.99203	0.94751
MD155	112301	4374	107927	0.98060	0.93363
Mel-12	92087	3758	88329	0.94475	0.86319
Mel-15	416468	21813	394655	0.97028	0.86888
Mel-16	275821	11980	263841	0.97476	0.90230
Mel-5	111537	4749	106788	0.90970	0.77826
Mel-624	51400	2375	49025	0.98790	0.91989
Mel-8	125171	6251	118920	0.97873	0.90506
pat-AC2	33520	1369	32151	0.97590	0.89782
pat-C	52730	2983	49747	0.94198	0.83667
pat-CELG	76094	3814	72280	0.90670	0.77183
pat-CP2	38669	1790	36879	0.98596	0.93735
pat-FL	77971	3629	74342	0.96953	0.87735
pat-J	45017	2552	42465	0.94136	0.83725
pat-JPB3	37208	1937	35271	0.98286	0.90181
pat-JT2	31038	1467	29571	0.98511	0.91892
pat-M	55715	2476	53239	0.99037	0.94648
pat-MA	73523	3682	69841	0.93019	0.81837
pat-ML	58353	3139	55214	0.97329	0.90406
pat-NS2	15841	636	15205	0.97946	0.91569
pat-NT	55388	2190	53198	0.96058	0.88904
pat-PF1	91435	4646	86789	0.95472	0.87410
pat-R	51515	2372	49143	0.99101	0.94739
pat-RT	52339	2537	49802	0.90433	0.77757
pat-SR	60016	2632	57384	0.94528	0.87819
pat-ST	28207	1256	26951	0.99216	0.94359
PD42	51227	2577	48650	0.97716	0.91132
RA957	243345	11037	232308	0.98843	0.92774
RPMI8226	117644	4524	113120	0.96572	0.87106
SK-Mel-5	67772	3293	64479	0.97899	0.89732
T98G	225822	10011	215811	0.91652	0.76603
THP-1	148285	5542	142743	0.97867	0.91325
TIL1	145613	5445	140168	0.98375	0.91860
TIL3	214746	8799	205947	0.99068	0.94019
U-87	252670	11585	241085	0.93549	0.80746

Table 4.S5. Cross-Validation training performance for the Eluted Ligand (EL) Multi Allele (MA) data used to train NetMHCpan-4.1. "# Peptides" refers to the total amount of peptides present for a given cell line ("Cell Line" column); "# Positives" and "# Negatives" represent the quantity of positive and negative peptides, respectively; "AUC" = Area Under the ROC Curve; "AUC0.1" = Area Under the ROC Curve integrated up to a False Positive Rate of 10%;

Table 4.S6. Due to its dimensions, this table is not embedded in this manuscript. Please refer to the online supplementary material to access it.

Table 4.S7. Due to its dimensions, this table is not embedded in this manuscript. Please refer to the online supplementary material to access it.

Table 4.S8. Due to its dimensions, this table is not embedded in this manuscript. Please refer to the online supplementary material to access it.

Chapter 5

Deep Learning for MHC motif discovery: a primer

5.1 Introduction

Up to now, we have seen how different machine learning strategies can be employed to generate peptide-MHC binding predictors. These strategies vary in their degree of complexity, which depends mostly on the datatype at hand. In the case of SA datasets, the usage of NNAlign-2.0 has proven to be quite effective in terms of capturing MHC receptor preferences and aligning their corresponding peptide sequences. For multi-allelic datasets, NNAlign_MA has also shown outstanding conditions to extend this task to also include MA data sets. For both cases, the core FFNN architecture behind these algorithms is able to properly learn meaningful patterns from input data, "crystallizing" such information in the network topology. Thanks to this, and after network training, it is possible to generate MHC binding motifs by means of predicting a certain quantity of random peptides (usually 200.000) and then taking the top (often 0.1-1%) scoring peptides and constructing a sequence logo. As an example, this well-established approach is used to generate the receptor preferences shown in the NetMHCpan-4.1 and NetMHCIIpan-4.0 motif viewers [265, 266], where the resolution of the logos depend, to the highest degree, on the quality of the inner representations learnt by the networks.

In a similar case, Fenoy et al. [267] went beyond the MHC domain, and employed the above technique to generate kinase phosphorylation motif logos. Such motifs were assembled after constructing the NetPhosPan algorithm, a generic deep convolutional neural network for ligand-receptor interaction predictions. Such a network was utilized to train a pan-specific peptide-kinase binding predictor, in the general fashion NNAlign_MA was trained to predict pan-specific peptide-MHC interactions. Architecture-wise, the main difference between NNAlign_MA and NetPhosPan is the implementation of 1-dimensional convolutional layers (CLs) in the latter, which are used to "scan" input kinase-domains for potential binding sites. Briefly, encoded kinase sequences are fed to a convolution module consisting of three parallel CLs of lengths (3, 5, 7) with 40 filters each, whose global max-pooled outputs are concatenated to the encoded ligand sequences, and then fed to a shallow FFNN with a single output neuron that returns the phosphorylation likelihood for submitted ligand-kinase pairs.

Due to its deep learning nature, the above example represents an interesting case for analysis. The rationale behind this is that convolutional layers have the capacity to slide over a given sequence (that may be temporal, semantic, etc.) and adaptively "pay attention" to different regions across the length of the input series (in Fenoy et al., such regions corresponded to the kinase binding domains). For this to happen, convolutional filter weights -characterized by matrices- become optimized following the strategies described in the introduction of this manuscript. This optimization dynamically molds and shapes the space of such filters, in the sense that their matrices capture well-defined preferences for specific amino acid compositions at specific positions. Then, in the presence of such an enriched representation space, a question may arise: could it be possible to extract receptor motifs directly from this space? This would mean to, somehow, construct receptor binding motif logos directly from the network weights rather than from the top predictions of random peptides described previously.

To start exploring possible answers for the above question, a careful dissection and understanding of the 1D convolution operator must be achieved first. From a deep learning perspective, 1D

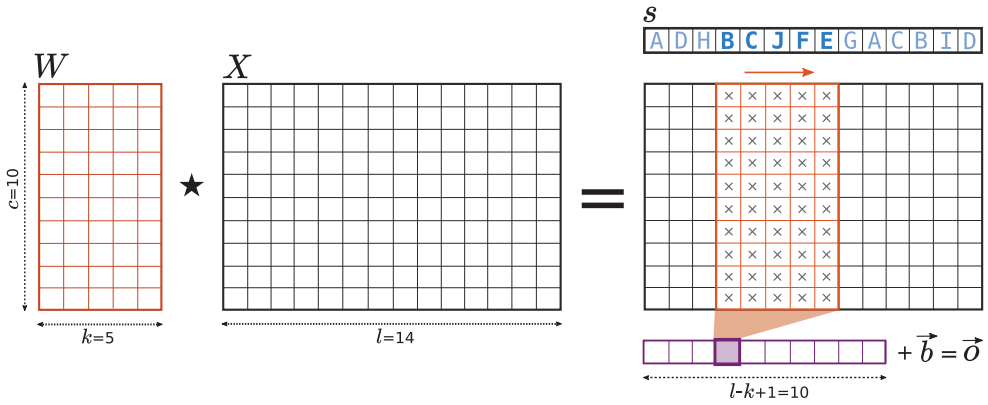

Figure 5.1. Toy example of a 1-dimensional convolution. Here, a filter W with a kernel size k of 5 and 10 channels c is being convolved with the encoding X of the hypothetical sequence s = ADHBCJFEGACBID, of length $l = 14$, and constructed via sampling of an alphabet of length c. W is slided from left to right over the position axis, and for each possible sliding step, the convolution $\star$ (the sum of all pairwise multiplications $\times$) is computed and stored in a vector of maximum length $l - k + 1 = 10$ (in purple). Then, this convolution vector is added to the bias $\vec{b}$, resulting in the final output $\vec{o}$. In this example, the filter is shown located "on top" of X at sliding position 3, with the convolution result stored in the position 3 of $\vec{o}$.

convolutions are, essentially, sliding matrices that are applied to some input data matrix in order to detect repeating patterns. Formally, we can define such operation as [268]:

$$\vec{o} = W \star X + \vec{b} \tag{5.1}$$

where $\vec{o}$ is the output, $W \in \Re^{c \times k}$ is the weight matrix (or filter of kernel size k) of the convolution, $\star$ is the convolution operator, $X \in \Re^{c \times l}$ is the encoding of some input sequence s (i.e. with BLOSUM) of length l (generated from an alphabet A of length c), and $\vec{b}$ is the bias vector (refer to Figure 5.1 for a graphical example). With this, the $\star$ operator is defined to 1) compute the pairwise multiplication of all elements in W and X at a given sliding position, 2) sum all these multiplications (condensing everything into a single scalar), and 3) store this scalar in an output vector at the same index of the current sliding position. As a result, the elements of $\vec{o}$ will carry information regarding the presence or absence of particular encoded characters $a \in A$ at specific positions of s. Moreover, filters of different k values will scan s for specific subsequences of length k, making 1D convolutions a fit candidate for multi-resolution motif recognition.

A rightful question, however, might be raised: why is that $\star$ is able to capture such information? The short answer is that, in essence, the 1D convolution is a type of sliding dot product operator. This means that a filter matrix W convolving X computes, per sliding position, intermediate column-wise dot products, which span an intermediate vector $\vec{u}$, whose position-wise sum yields the final convolution output (see Figure 5.2 for a visual example). Since the dot product between two vectors is a way of measuring the projection of one onto the other, such column-wise dot products represent an explicit way of projecting elements of s onto a filter's space. With this, $\vec{u}$ can be directly interpreted as the projection of sequence s onto a filter f (or, analogously, the projection of X onto W), at a specific sliding position.

Since a W of size k can be placed at $l - k + 1$ different positions over X, a convolution will span a total of $l - k + 1$ possible output values. Among these values, there will be a maximum one, and it will be associated to a specific sliding position p. This position is of great importance, since it conveys the starting position of the subsequence of s which generated the greatest filter response. With this last piece of information, we can now associate this subsequence to a filter's peak activation position p, and as a result recover the best projection vector $\vec{u}_p$, whose i-th component is defined as:

$$\vec{u}_p[i] = W_{i,j} \bullet X_{i,p+j} \tag{5.2}$$

with $0 \leq j \leq k - 1$ and $\bullet$ being the dot product. Notice how j indexes only column positions, meaning that the operation is conducted across rows (in the same way as displayed in Figure 5.2). If we now take into account that $\vec{u}_p$ comes from a subsequence of s, we can remap the elements of this vector into a "projection matrix" whose rows are indexed by alphabet elements a, and columns

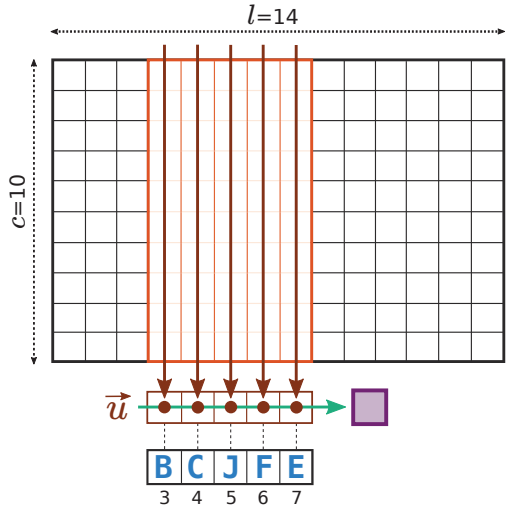

Figure 5.2. Computing a 1D convolution from a sliding dot product viewpoint. Here, the same situation from Figure 5.1 is shown. The convolutional filter W (in orange) is sitting on top of X (in black) at position 3, and extends itself up to position 7. In a column-wise manner, dot products between W and X are calculated and stored in an intermediate projection vector $\vec{u}$ (in brown). Then, by summing the components of $\vec{u}$ (green arrow) the convolution value for position 3 is obtained (purple block). Notice how there is a unique correspondence between elements of $\vec{u}$, letters of the subsequence BCJFE, and the positions of such letters in s. A side note: another possible approach to compute the same convolution value is to do the dot product row-wise, and then summing (this, however, is not going to be covered here).

by kernel positions. Moreover, considering an input space S (with multiple sequences s), summing the projection matrices of each s will serve as a way of sampling a target convolution's projection image. We will refer to this sampled image simply as projection, and denote it with ϕ according the following expression:

$$\phi = \sum_{s \in S} \sum_{j=0}^{k-1} I(s_{p+j} = a) \cdot \vec{u}_p \tag{5.3}$$

where a represents any symbol in alphabet A (in our case, the 20-letter amino acid alphabet), s_{p+j} is the symbol found at position $p+j$ of s, and I is an indicator function [269] of s (defined as 1 if $s_{p+j} = a$, 0 otherwise) used for indexing amino acids in ϕ (refer to Figure 5.3 for an example computation). Given the fact that a filter W will have a preference for specific amino acids at certain positions, ϕ will capture the presence/absence of such a preference in S, and, furthermore, give an estimate of its amplitude (by means of accumulation by summing for all s). Also, thanks to the indexing provided by I, ϕ will have akin characteristics to a PSSM, and thus it will be possible to treat it as a such (i.e. to generate a logo for visualization).

5.2 Materials and Methods

In the case of peptide-MHC interactions, the aforementioned strategy could be used to project binding sequences s and extract possible binding motifs scattered across different positions in the input space. The problem is that, in order for this to work, filter weights need to be tuned to specific values for the detection of such motifs. Now, if we think of convolutional neural networks as a collection of convolutional layers, and convolutional layers as stacks of convolutional filters, we should be able to assemble some kind of CNN architecture, optimize its filter weights using backpropagation, and then compute their projections ϕ over some input peptide space S. To start exploring possible experimental scenarios, we adapted the architecture of NetPhosPan to the peptide-MHC system (Figure 5.4), using the Keras deep learning API [270]. Our approach consists of an allele-specific ANN composed, essentially, of a convolution module with six parallel convolutional layers each of kernel sizes 1, 2, 3, 5, 7, or 9 (in the NetPhosPhan model, kernels of length 3, 5, and 7 were used; we here add a 9-kernel to account for 9-mer binding cores, and 1- and 2-kernels to scan for potentially smaller patterns). The chosen activation function for all CLs was hyperbolic tangent (`tanh`), and the padding was set to valid (this means that, after convolving a peptide of length l with a filter of length k, the resulting vector will not be zero-padded to the right, and thus will have $l-k+1$ positions). Each activated CL is then fed into a `GlobalMaxPool()` operation, which extracts the

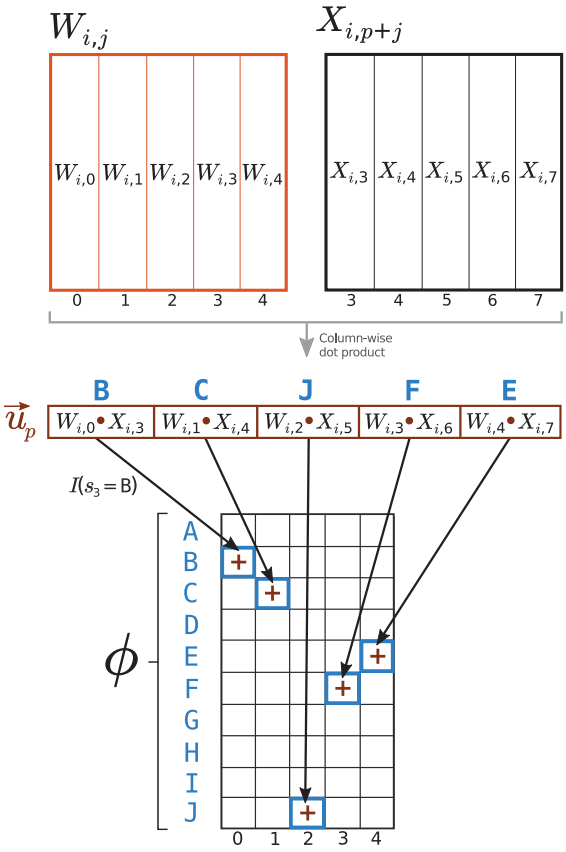

Figure 5.3. Example computation of ϕ for the toy scenario presented in Figure 5.1. Here, Here, we will assume that subsequence BCJFE yielded a maximum convolution value at position $p = 3$. For each overlapping column pairs between W and X, the dot product is calculated and stored in the projection vector $\vec{u}_p$ (in brown). Then, elements of $\vec{u}_p$ are summed into ϕ, a matrix whose coordinates are indexed by (letter, position) pairs, using an indicator function I as row mapper (in this figure, an example for B at s_3 is shown). With this dot product accumulation, the contribution of to the projection of S onto is leveraged, improving its resolution.

maximum value of the activated convolution output. Then, all pooled values become concatenated into a single vector and fed to a single output neuron with sigmoid activation.

Since the proposed architecture has multiple convolutional filters $f \in F$ (with F being the collection of all filters), we will refer to their projections as ϕ_f. Also, since a single output neuron and no hidden layer is present in the proposed model (Figure 5.4), a weighted projection $\hat{\phi}_f$ of S onto f can be computed as:

$$\hat{\phi}_f = \omega_f \cdot \phi_f \tag{5.4}$$

where ω_f is the weight connecting filter f (after `GlobalMaxPool()`) to the output neuron (added to account for the network's assigned importance to). Since in our experimental setup S corresponds to a list of specific MHC-restricted peptides, $\hat{\phi}_f$ shall display a filter's "viewpoint" of such MHC binding preferences. Thanks to this, meaningful information regarding receptor motifs might be extracted from such a unique perspective. Also, with the above expression, different $\hat{\phi}_f$ matrices may be combined together to generate composite projections for any filter combination. For instance, the full network projection can be calculated as the sum of all weighted projections $\sum_f \hat{\phi}_f$.

5.3 Results

Having defined the architecture and projection computation procedures, we next proceeded to the training step. A total of six models were trained, accounting for HLA-A*02:01, HLA-A*01:01 and HLA-B*08:01 (HLA-I); and HLA-DRB1*01:01, HLA-DRB1*03:01 and HLA-DRB1*11:04 (HLA-II). Data for the class I system was extracted from the clustering output of NNAlign_MA training, whileas data for class II was obtained from the clustering output of NetMHCIIpan-4.0 training.

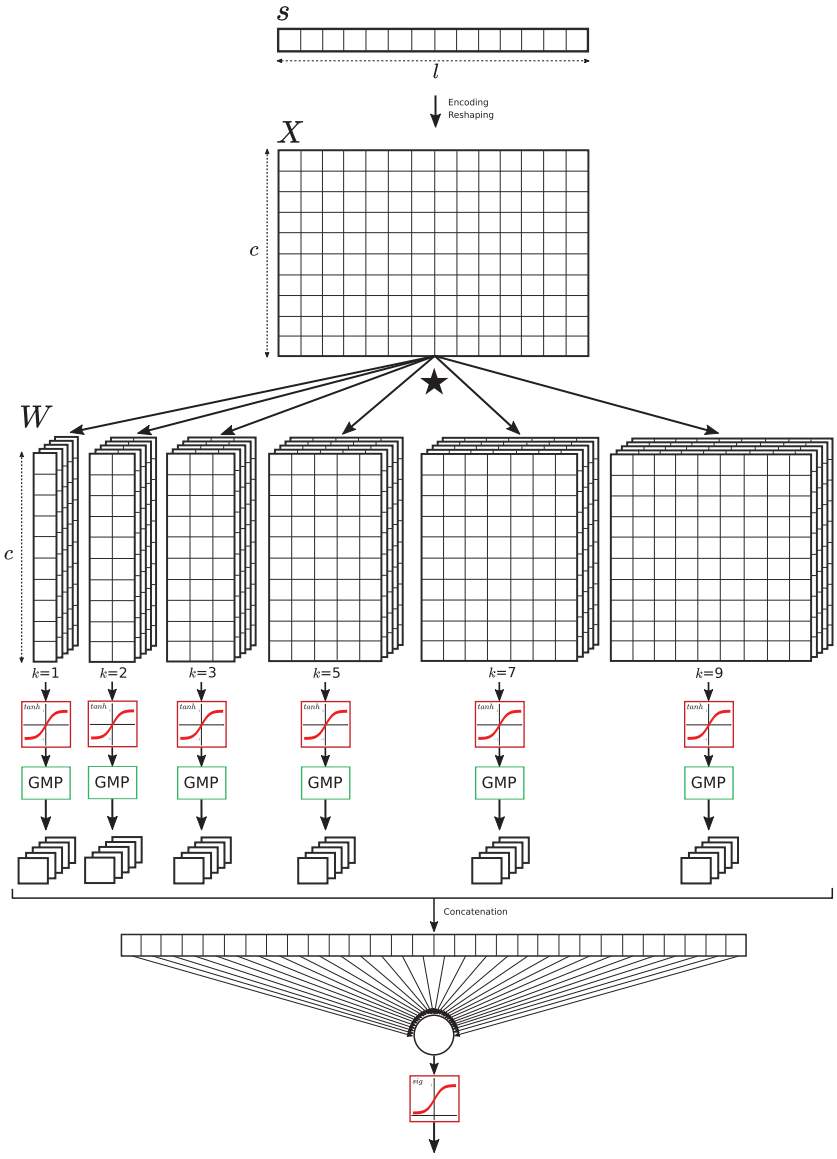

Figure 5.4. Architectural representation of the proposed model. From top to bottom, an input sequence s of length l assembled from an alphabet of length c is encoded and reshaped into a matrix X in order to fit the convolutional layers input shape (c, l). For MHC-I and MHC-II, $l = 14$ and $l = 21$, respectively. Also, BLOSUM encoding implies $c = 20$. Six convolutional layers of kernel sizes 1, 2, 3, 5, 7 and 9 (with 5 filters each) are fed the input . CLs outputs are then fed to `tanh` activation functions, and then into `GlobalMaxPool()` (GMP) blocks. The output of each GMP are 5 scalars (each one associated to an upstream filter), which become concatenated into a unique vector of length 30 (six kernels times five filters each) and fed to the output neuron, whose activation function is a sigmoid.

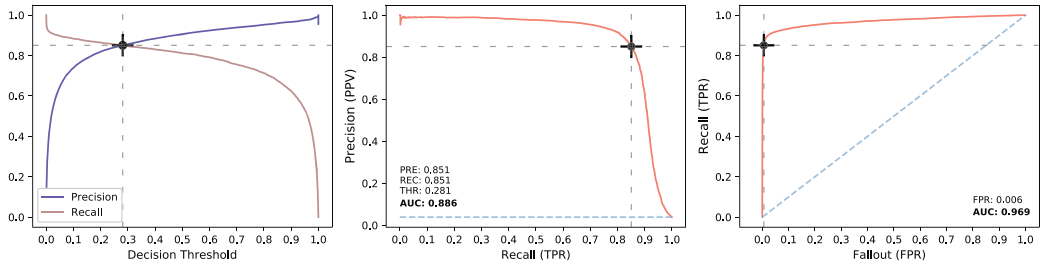

Figure 5.5. Cross-validation metrics for the HLA-B*08:01 model. From left to right: Precision and Recall curves as a function of the decision threshold, Precision-Recall curve and ROC curve. The black cross indicates the chosen operation point for the model (in this case, the intersection of the precision and recall curves). The AUC values for PRC and ROC are shown in bold.

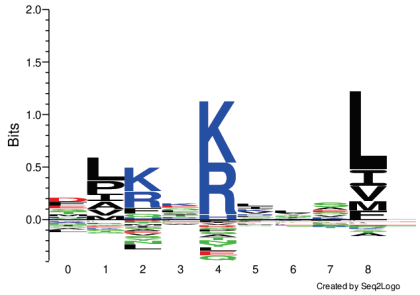

Figure 5.6. Known HLA-B*08:01 binding motif, extracted from the top 0.1% of 200.000 random peptide predictions.

Positive peptides were labeled with a target value of 1, while negative peptides with a target value of 0. Negative enrichment was 5x the quantity of the most abundant positive length, for all lengths as described earlier. Peptides were padded to a maximum length of 14 (in the case of MHC-I), and a maximum length of 21 (in the case of MHC-II). The chosen encoding was BLOSUM50, rescaled by a factor of 5; the padding character "X" (wildcard amino acid) was encoded with a vector of length 20 and values of -1. Optimization was done with gradient descent (learning rate of 0.05), using mean squared error as loss function. Training was conducted for 200 epochs with an early stopping of patience 40, following a 5-fold cross validation schema with homology reduction, and a batch size of 64. A total quantity of 5 filters per convolutional kernel was used for all the models, except HLA-DRB1*11:04, for which we used 10.

All models exhibited overall high performance values, with an average cross-validated ROC AUC of 0.941 and PRC AUC of 0.793 (see Supplementary Figure 5.S12 for performance details of all models). In particular, for HLA-B*08:01, the reported CV metrics were as displayed in Figure 5.5. Having corroborated the model's successful training, the goal now was to define an optimal projection of the input space (HLA-B*08:01 peptides) onto the network's filter space. For the approach to succeed, such projection should display similar characteristics to the known HLA-B*08:01 binding motif (shown in Figure 5.6), which will be our target PSSM.

To do this, four different cumulative projections alternatives were computed for all filters (Figure 5.7). From this figure it is apparent that not using ω_f at all (first panel from the left) results in a highly divergent pattern with no clear similarities to the target logo (Figure 5.6), and an almost monotonically decreasing information content is observed as a function of the position. Then, the inclusion of the absolute value of ω_f (second panel) helps dilute such monotony, with the P5 anchor slightly emerging as well. Things look much better when using the sign of ω_f as weight (third panel), with a sharper P5, a first appearance of P3 and a correct positioning of the P9 anchor. Finally, employing ω_f (fourth panel) yields the best result of all, with overall more informative and crisp anchors, but also with correct, depleted enrichments at unimportant positions as well. Refer to Supplementary Figure 5.S13 for similar results for the remaining models.

With these results, we can observe how important a filter's associated output weight is for the architecture's inner workings. If this were not the case, the summing of projections displayed in Figure 5.7 would have shown similar characteristics for all four weighting cases. This suggests that, to some degree, the network training assigns an importance level to each filter, and does this by

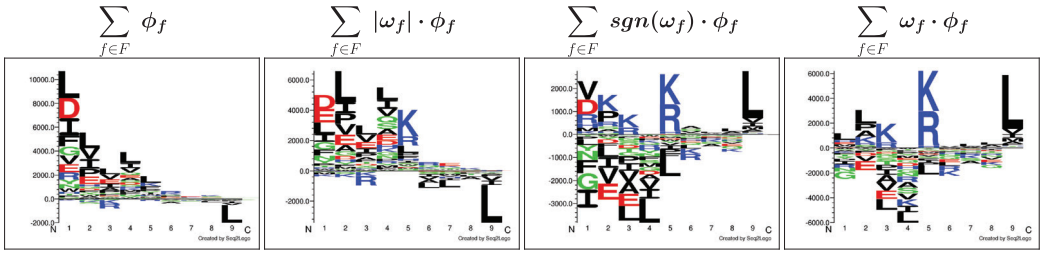

Figure 5.7. Four different cumulative projections for the model trained on HLA-B*08:01 data. From left to right: no weighting, weighting using the absolute value of the output neuron weight, weighting using the sign of the output neuron weight, and weighting using the output neuron weight.

increasing or decreasing the magnitude of its corresponding weight (higher valued filters become more important, and vice versa). In addition to the magnitude, the sign seems to play a crucial role as well. We believe this can be explained, initially, by looking at the chosen activation function (hyperbolic tangent) outputs. Since the image of tanh lies in the [-1, 1] interval, the separation between positive and negative classes might be done by squishing positive predictions towards 1 and negatives to -1, or by doing the exact opposite. This works because, essentially, the network only needs to allocate different classes to different regions in its internal mapping to achieve separation. Then, to be consistent with the preferences dictated by the real MHC binding motif, the learning process will enforce a negative or positive sign on each weight, and thus filter outputs will become flipped accordingly when summed by the output neuron (refer to Figure 5.8 for an example).

Having stated the importance of w_f, a side-to-side comparison between the logo of the raw input data, the sum of all $\hat{\phi}$ and the logo obtained from top predictions is shown in Figure 5.9. Here, we can observe a great resemblance between our cumulative weighted projection and the NetMHCpan-4.1 logo. In addition, the logo generated from raw peptides displays more dissimilar properties in comparison to the projection, hinting that, indeed, we are recovering valid information from this technique. Also, it is important to clarify that, in contrast to Equation 1.14, any $\hat{\phi}$ (and, in particular, the blend $\sum_f \hat{\phi}_f$) represents a pseudo-PSSM that encodes the network's internal abstraction for particular combinations of amino acids, and as such cannot be read as information content in the classical way. In practical terms, this means that the y-axis of the logo representation of any projection combination will display dimensionless quantities (in contrast to NetMHCpan ones, which will display information units). Because of this, logo comparisons will be made qualitatively, by means of visual inspection (a proper objective quantification, such as Spearman's rank correlation is possible, but exceeds the scope of this chapter).

Next, we repeated the projection extraction procedure with the model trained on HLA-A*01:01 binding data, and computed the corresponding logo (Figure 5.10). As can be seen, the scenario now looks different. First and foremost, the raw data logo for this allele exhibits a highly repeating Tyrosine enrichment towards C-terminus. This is a direct consequence of the chosen padding, which leaves the N-terminus at a fixed position, and as a result the P9 anchors across input peptides become misaligned. Virtually, this is bad news, because such misalignments are now transferred to the cumulative projection, which now displays large differences at P9 in comparison to the NetMHCpan logo.

As kitschy as it sounds, a misalignment problem calls for an alignment solution, for which we developed a small pipeline. First, the projection with highest amplitude value (or with more "pseudo-information" content) is selected from the convolutional layer with the longest kernel size. This will become the alignment template $\hat{\phi}_t$. Afterwards, all remaining projections $\hat{\phi}$ are sorted from higher to lower maximum positive amplitude within each subgroup of kernel size k. Starting from the highest k value, a 2-dimensional cross-correlation is computed between $\hat{\phi}_t$ and the first $\hat{\phi}$ of the sorted list; as a result, an offset o -such that correlation is maximum- will be extracted. Following, o is used to apply an offset correction to $\hat{\phi}$, phasing it with the template. Next, offset-corrected $\hat{\phi}$ becomes summed to $\hat{\phi}_t$, updating it. Then, this whole correlation-correction schema is repeated, but now using the next projection matrix from the sorted list. The process continues until all elements of such list are consumed, which results in $\hat{\phi}_t$ becoming the final alignment. The output of applying this pipeline, for all trained models, is shown in Figure 5.11.

As seen in Figure 5.11, the final outcome of projecting S onto the filter space of trained models varies drastically depending on the HLA molecule being analyzed. The easiest case turned out to be HLA-B*08:01, whose projection looks good before alignment (just some minor errors are present,

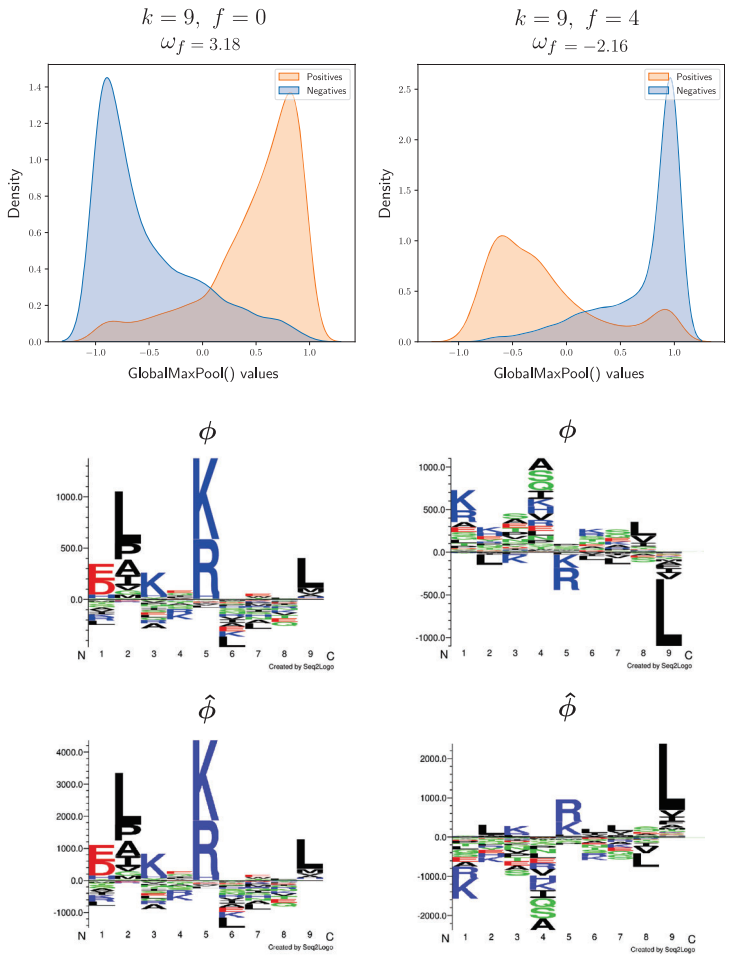

Figure 5.8. Role of the output weight ω_f in generating consistent internal projections. On the top row, density estimations for `GlobalMaxPool()` outputs are shown for the first filter ($f = 0$, first column) and last filter ($f = 4$, second column) of the convolutional layer of kernel size $k = 9$ (the corresponding output weight is shown on top of the plots). Densities corresponding to the positive and negative training sets are plotted separately. Given that GMP receives the output of a `tanh` activation function as input, such values will be bound between -1 and 1, guiding the network to squish class separation on these extremes. Notice how such separation can be obtained disregarding the tanh sign: for $f = 0$, positives are squished towards 1, whereas for $f = 4$, towards -1. As a direct consequence of this, projections ϕ for both filters (middle row) will be x-axis mirrored (both will display similar anchors, but on opposite sides). When applying the weighting $\hat{\phi} = \omega_f \cdot \phi$ (bottom row), the mirroring becomes fixed, and now important anchors point towards the same direction (amplitudes become corrected too). With this, the summation of $\hat{\phi}_f$ over all $f \in F$ will yield a more sound PSSM, as seen on the rightmost logo of Figure 5.7. Note: density plots display values outside the [-1,1] interval. This is an artifact of the visualization; these do not exist in reality.

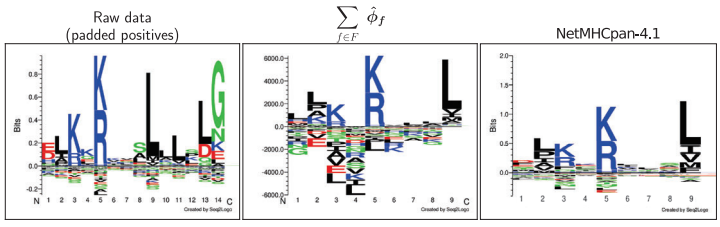

Figure 5.9. HLA-B*08:01 logo comparison between raw input data (left), summation of all weighted projections (middle) and NetMHCpan-4.1 top 0.1% predictions (right). Raw data logo has 14 positions in total because all input peptides are padded to such length.

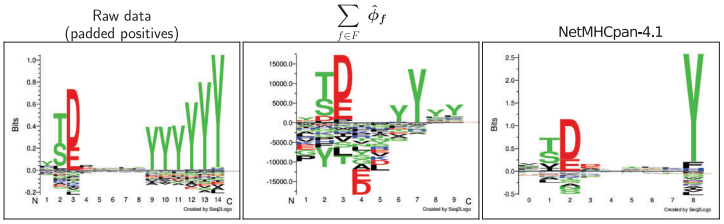

Figure 5.10. HLA-A*01:01 logo comparison between raw input data (left), summation of all weighted projections (middle) and NetMHCpan-4.1 top 0.1% predictions (right). Raw data logo has 14 positions in total because all input peptides are padded to such length.

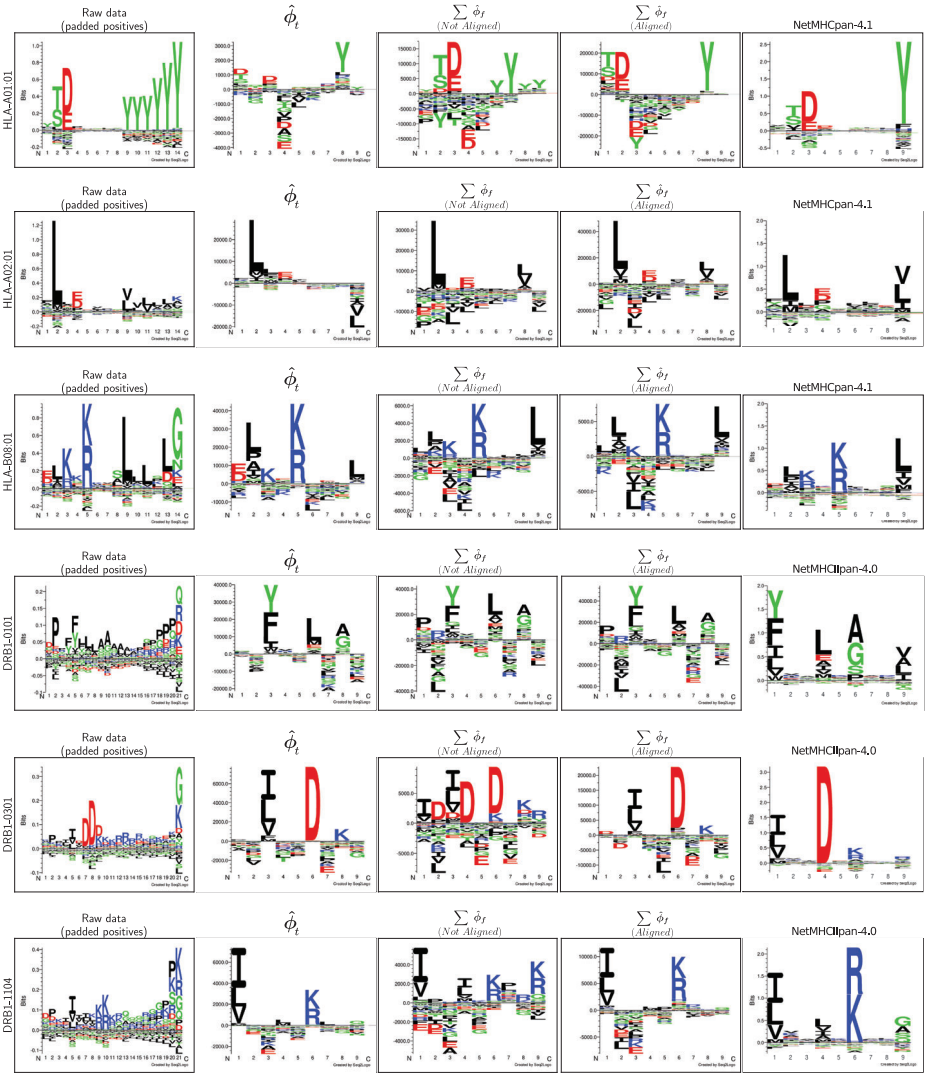

Figure 5.11. Comparison between logos for raw input data (first column), alignment template (second column), weighted projection accumulation (third column), aligned weighted projection accumulation (fourth column), and known NetMHCpan logo (fifth column), for all employed models (one per row).

i.e. K/R enrichment at P2, or P9 anchor slightly appearing at P7 and P8). When aligned, these errors dissipate, but on the other hand, R from P3 is lost. For HLA-A*01:01, the existence of misalignments in the algorithm's solution was evident, and as can be observed, P9 "ghost anchors" become unified after aligning. One thing to observe is that the sum of aligned projections starts from NetMHCpan logo's P2, but positional relationships are conserved. The scenario for HLA-A*02:01 is an interesting one, in the sense that there is almost no difference when aligning (besides amplitude gain), and moreover the malposition of P9 could not be rescued. Moving to the MHC-II system, both pre- and post-alignment projection logos for HLA-DRB1*01:01 look rather similar, and no qualitative gain can be seen. However, it is noticeable how both projections start two positions to the left of NetMHCIIpan's logo P1, leaving the final motif two positions short towards C-terminus. We believe this happens because of a consistent Proline (P) signal present upstreams of P1, which is in concordance to previously characterized MHC-II antigen processing signals [215]. Moving on, HLA-DRB1*03:01 exhibited a clear optimization thanks to the alignment pipeline, that helped unify scattered P1, P4 and P6 anchors. Also, as observed in the previous molecule, the two position shift is present. The last allele, HLA-DRB1*11:04, benefitted as well from the alignment, resulting in sharp P1 and P6 anchors. On the downside, less important anchors such as P4 and P9 became "diluted", since these are visibly present in the aligned $\hat{\phi}$, but have small magnitudes. Additionally, and in contrast to HLA-DRB1*03:01 and HLA-DRB1*11:04, we do not observe the two position shift, and all anchors are in phase with the NetMHCIIpan logo. Finally, it is important to state that, independently of individual model gains, losses, and caveats, $\hat{\phi}$ always represents a more accurate binding motif estimation when compared to the raw input data itself. For a more comprehensive display on how alignments were generated from summing offset-corrected projections, refer to Supplementary Figures 5.S14, 5.S15, 5.S16, 5.S17, 5.S18 and 5.S19.

5.4 Discussion

With all the results exposed above, it becomes evident that neural networks generate complex internal representations of the outer world we present to them. Given the sparse nature of this abstraction space (where information is stored in weights and connections), looking under the hood to understand such a language is not trivial. In this chapter, we have made a humble attempt to do so, focusing on the particular case of 1-dimensional convolutional neural networks.

Since this type of deep learning approach is an ideal candidate for the analysis of peptide-MHC binding interaction data, we assembled a primordial architecture to study the problem in a controlled setting. We did this by utilizing the convolution operator as a way of projecting the input set onto the weight space of the architecture's convolutional filters. Such an approach enabled us to construct and then visualize inner network constructs using the well known sequence logo representations. In total, we trained six different peptide-MHC binding predictions models. Half of these models were dedicated to HLA-I, a relatively simpler molecule that served as the first benchmark for our projection pipeline. Results looked promising, since calculated projections had acceptable resemblances to known experimental binding motifs. However, an offset correction step was introduced in order to solve a clear misalignment issue present in some anchors. Such correction was indeed a success, but also a much-needed intervention to make projections for HLA-II (a more complex system) look correctly aligned with experimental data as well.

Thanks to the above, we believe such novel ways of looking at 1D convolutions might serve to expand the current understanding on how these ANNs construct, combine and exploit inner representations of input data. We consider the work presented here as a first approach on dealing with such conundrum, leaving a lot of room for improvement and scaling. For instance, and from what can be observed in Supplementary Figures 5.S14, 5.S15, 5.S16, 5.S17, 5.S18 and 5.S19., plenty of individual projections seem to not display informative content (no clear patterns can be distinguished, and/or amplitude is low), hinting that not all filters become optimized equally, and opening the question of why this happens (is this a real issue, or just how the network normally behaves?). This could be further explored using regularization techniques (L1/L2 penalties, dropout layers, etc.) and measuring how these impact in the projection shapes. Since output weights are of central importance to this approach, and such weights properties depend directly on how filter activations are calculated, another interesting question is the role of such activation functions in generating projections. In our case, hyperbolic tangent was employed, and this forced output weights to become positive or negative in order to flip filter responses to render them useful. The usage of different activations may impact this behaviour, and even reveal different network strategies for organizing information. On another take, projection motifs are currently constructed capping the position axis to the max value of k, imposing a limitation to motif discovery, since longer motis may be present in our data. Enabling the offset correction step to align similar projections at any position shall help creating longer representations, overcoming the limitations imposed by fixed convolutional kernel sizes.

Finally, expanding the current architecture to a pan-specific framework is also of great relevance. Doing so, a single convolutional filter will encounter the challenge of having to capture information from multiple MHC molecules. With this, filter-MHC exclusivity should be lost, and thus new patterns of abstraction might emerge. Directly using the current architecture to train a pan-specific model may, however, incur in performance problems. This, mainly, due to the fact that a single output neuron is currently employed, coming up short in the task of dealing with multiple MHC information. So, expanding the architecture to leverage multiple output neurons (or even more, multiple hidden layers) seems a good deed. When doing this, however, one shall consider an adaptation of the weighting schema for $\widehat{\phi}$ computation, since at the moment it only contemplates a unique output neuron.

In conclusion, the horizon of such deep learning approaches appears quite vast, but seems promising and worth exploring.

5.5 Supplementary Material

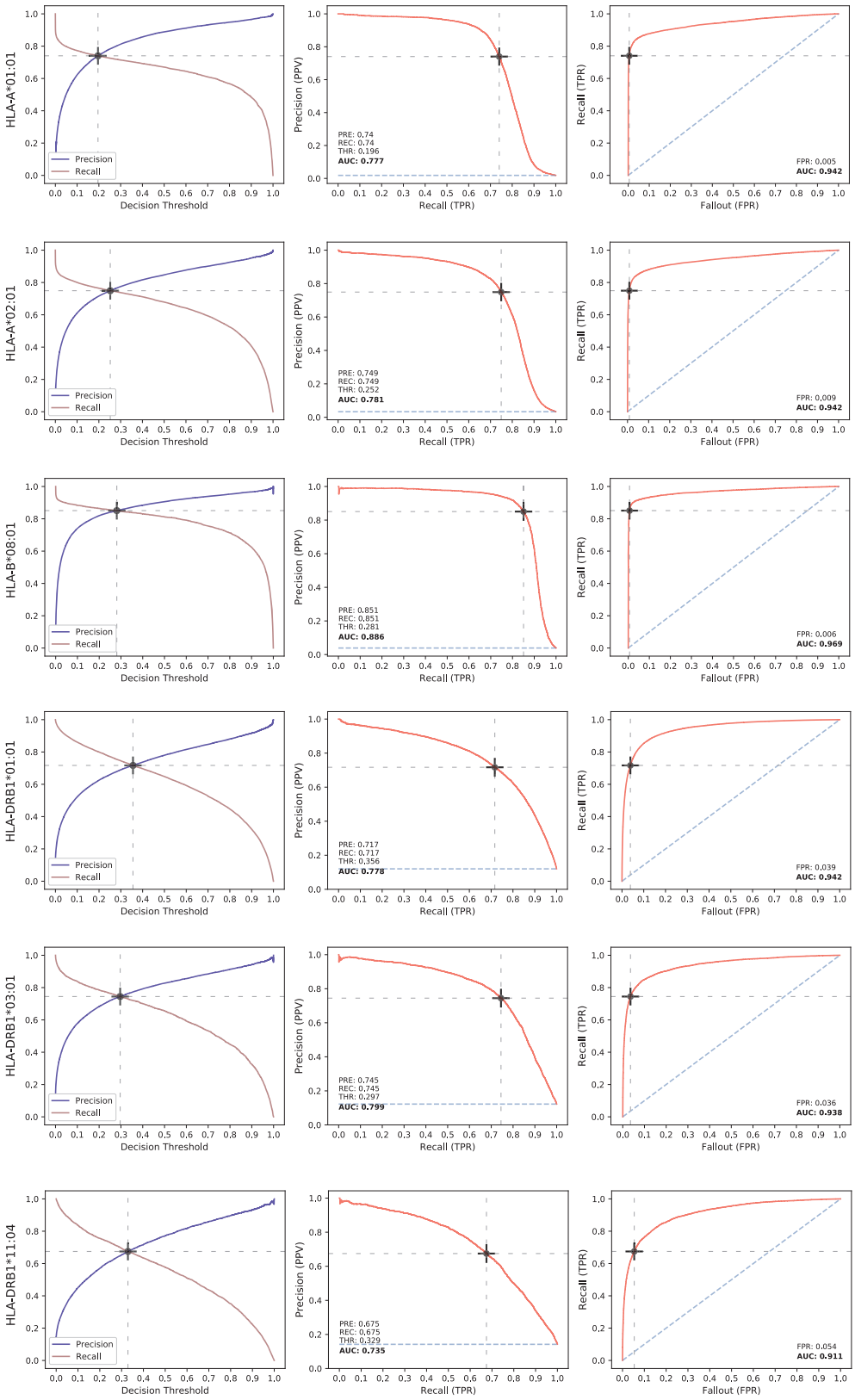

Figure 5.S12. Cross-validation metrics for all trained models (one per row). From left to right columns: Precision and Recall curves as a function of the decision threshold, Precision-Recall curve and ROC curve. The black cross indicates the chosen operation point for the model (in this case, the intersection of the precision and recall curves). The AUC values for PRC and ROC are shown in bold.

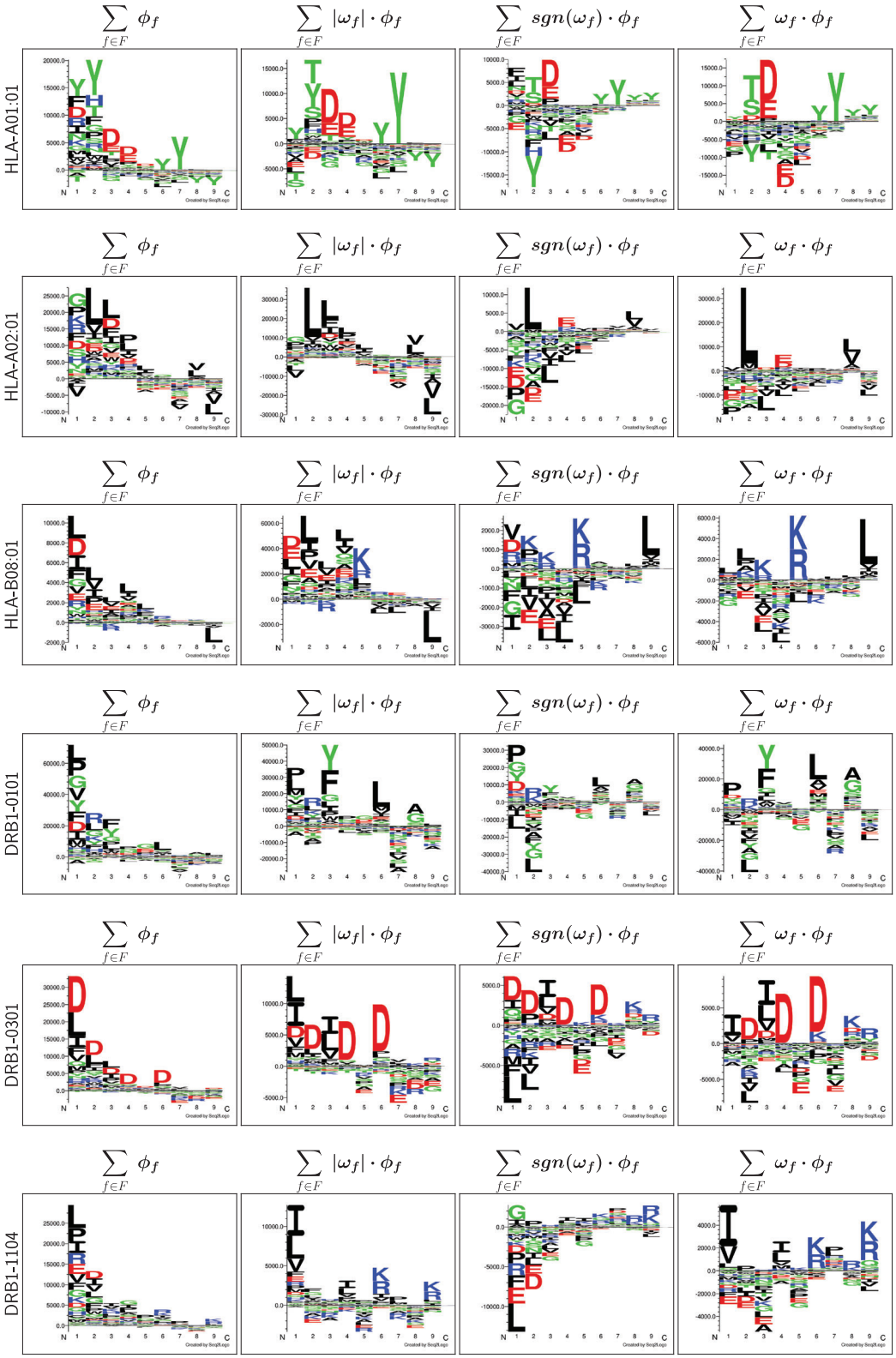

Figure 5.S13. Logos resulting from different ϕ_f weighting approaches, for each one of the trained models (indexed by rows). From left to right columns: no weighting, weighting using the absolute value of the output neuron weight, weighting using the sign of the output neuron weight, and weighting using the output neuron weight.

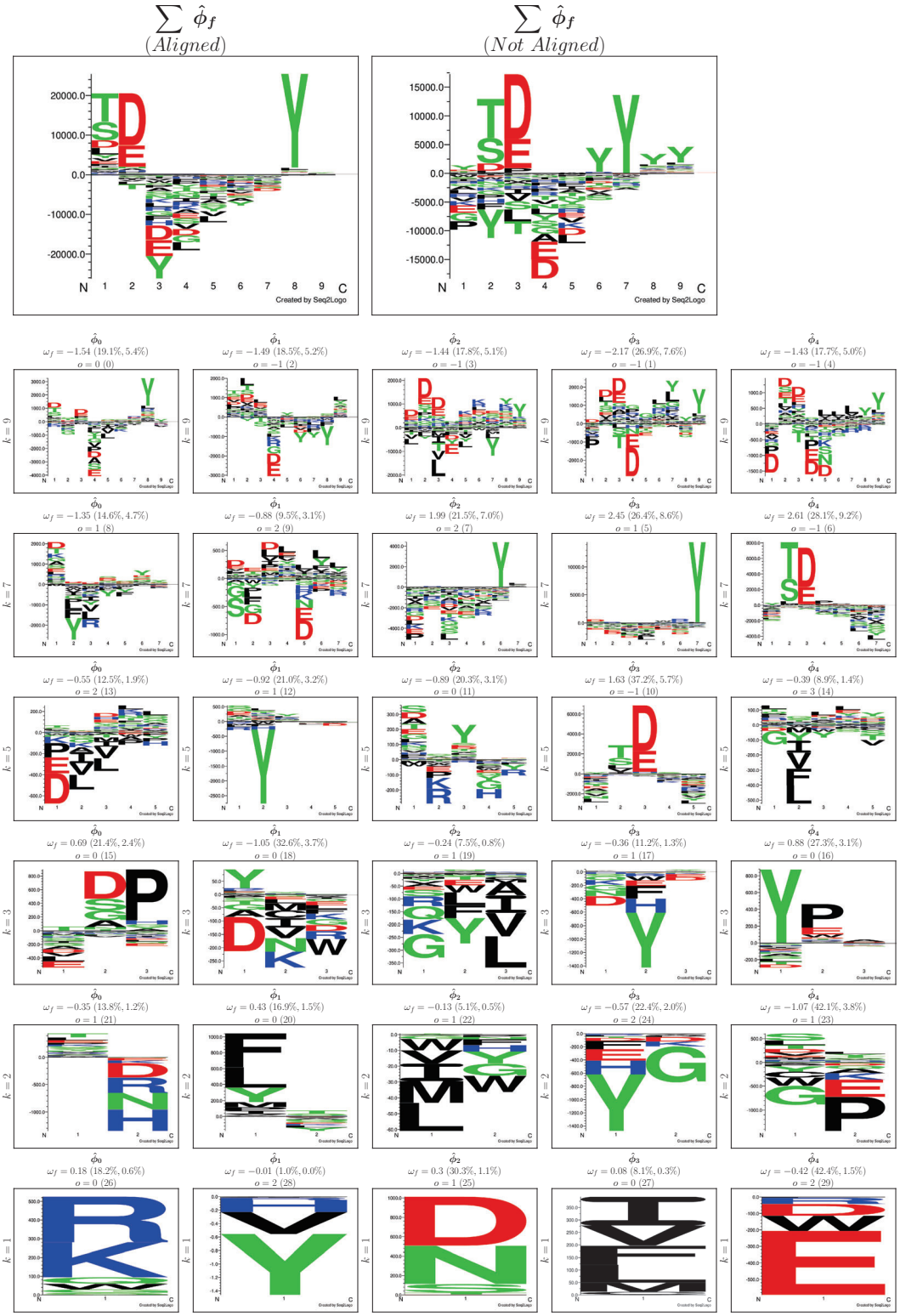

Figure 5.S14. Full projection panel for the HLA-A*01:01 molecule. On top, the aligned and not aligned accumulated weighted projections are shown in the left and in the right, respectively. Below, all the individual weighted projections $\hat{\phi}$ for all filters (indexed by columns) in each convolutional layer of kernel size k (indexed by rows) are shown. On top of each projection, the value of ω_f (weight connecting the corresponding filter to the output neuron) is shown. Next to this, and in between parentheses, the contribution percentage of ω_f to the convolutional layer and to the full network are displayed. Below, the offset correction value o is exhibited; next to it, in between parentheses, the order in which the projection was added to the cumulative projection is shown.

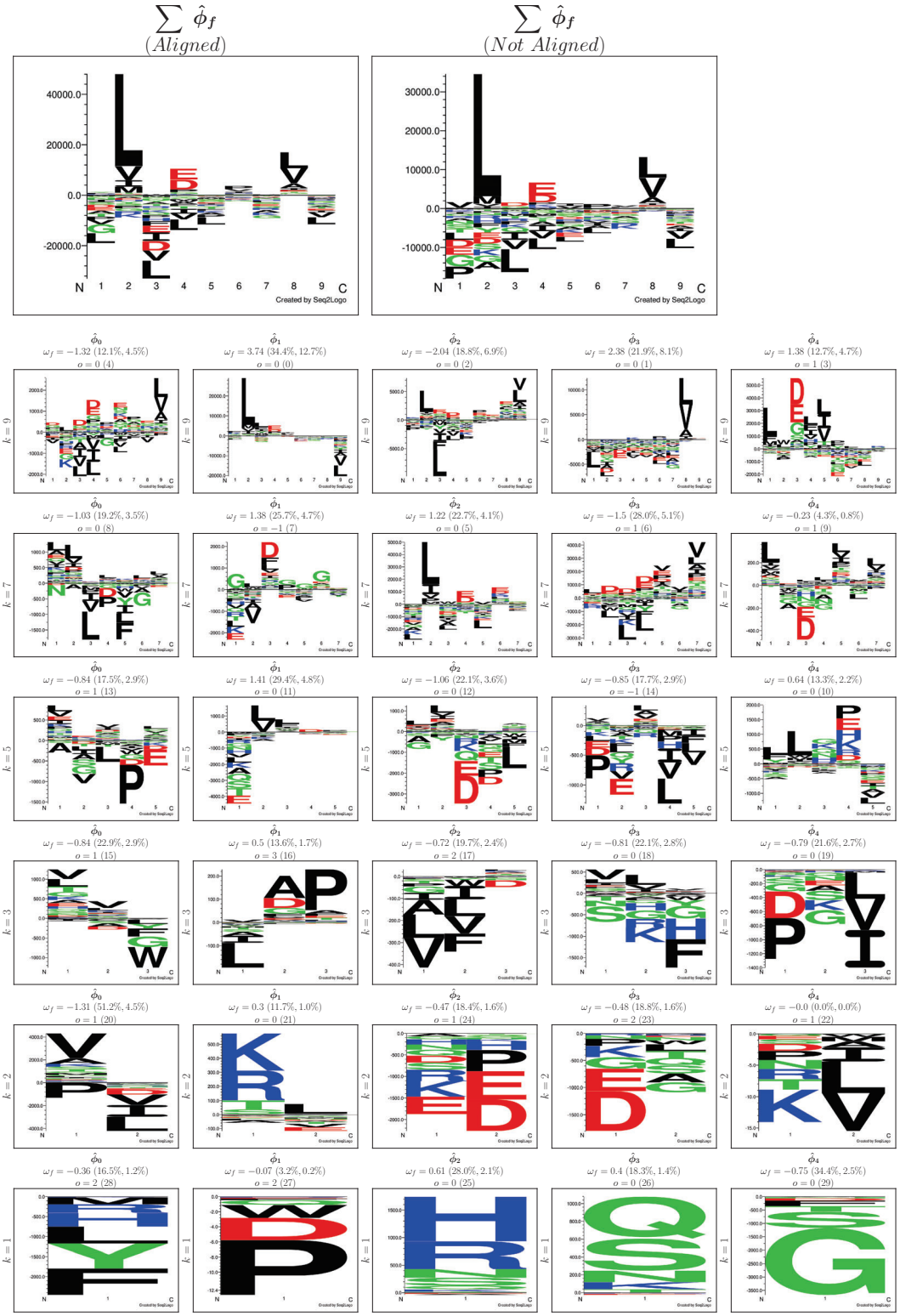

Figure 5.S15. Full projection panel for the HLA-A*02:01 molecule. On top, the aligned and not aligned accumulated weighted projections are shown in the left and in the right, respectively. Below, all the individual weighted projections $\hat{\phi}$ for all filters (indexed by columns) in each convolutional layer of kernel size k (indexed by rows) are shown. On top of each projection, the value of ω_f (weight connecting the corresponding filter to the output neuron) is shown. Next to this, and in between parentheses, the contribution percentage of ω_f to the convolutional layer and to the full network are displayed. Below, the offset correction value o is exhibited; next to it, in between parentheses, the order in which the projection was added to the cumulative projection is shown.

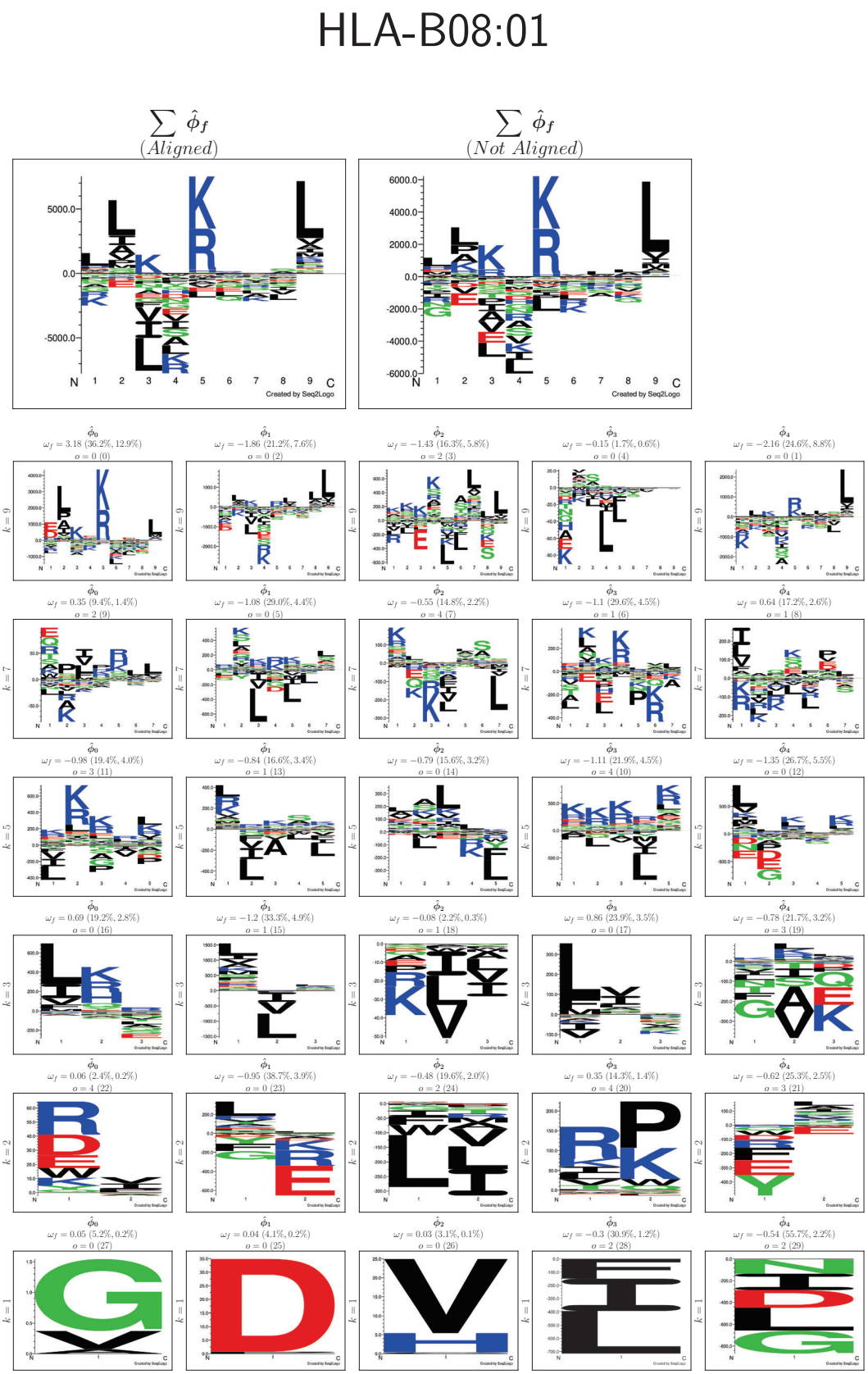

Figure 5.S16. Full projection panel for the HLA-B*08:01 molecule. On top, the aligned and not aligned accumulated weighted projections are shown in the left and in the right, respectively. Below, all the individual weighted projections $\hat{\phi}$ for all filters (indexed by columns) in each convolutional layer of kernel size k (indexed by rows) are shown. On top of each projection, the value of ω_f (weight connecting the corresponding filter to the output neuron) is shown. Next to this, and in between parentheses, the contribution percentage of ω_f to the convolutional layer and to the full network are displayed. Below, the offset correction value o is exhibited; next to it, in between parentheses, the order in which the projection was added to the cumulative projection is shown.

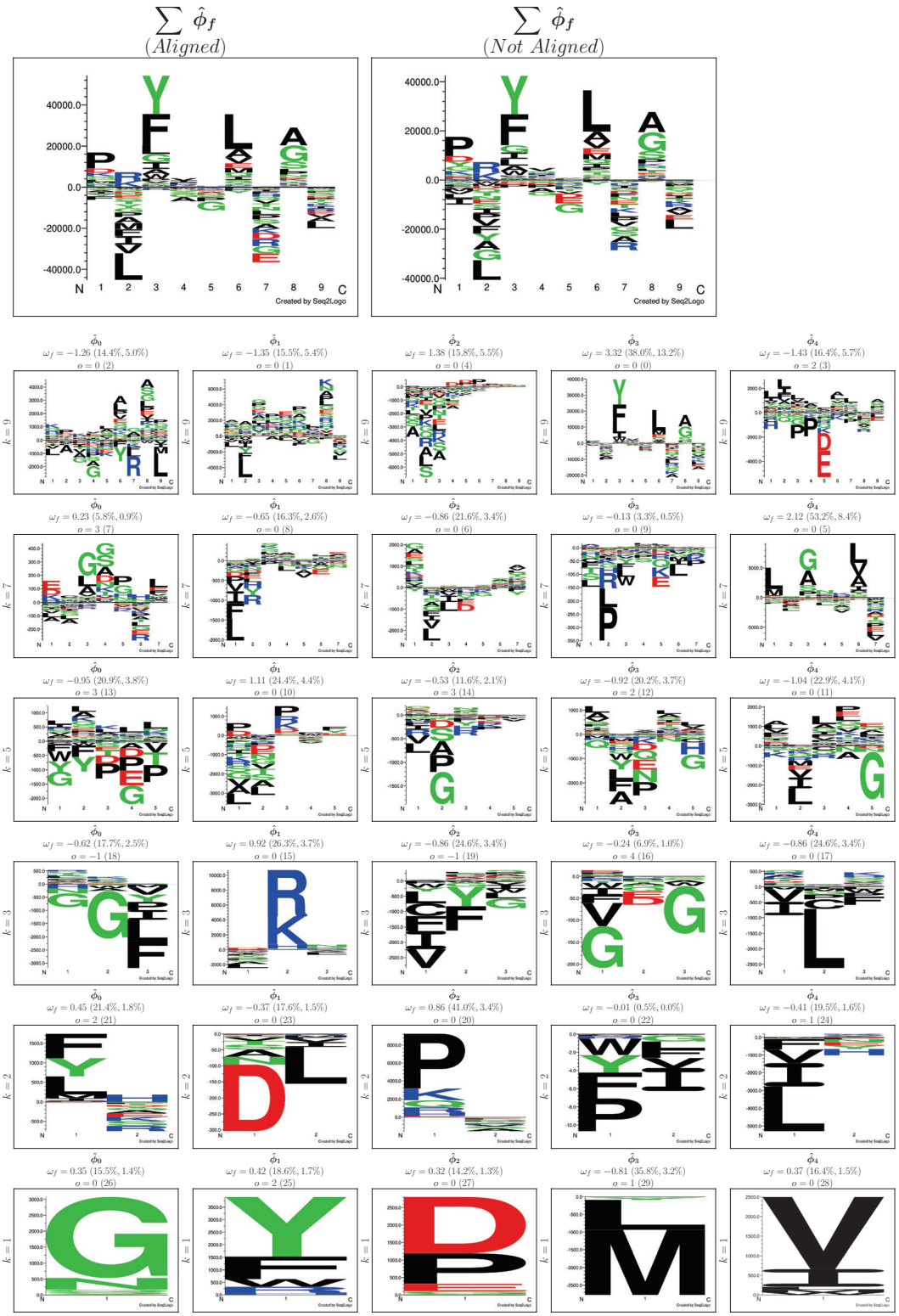

Figure 5.S17. Full projection panel for the HLA-DRB1*01:01 molecule. On top, the aligned and not aligned accumulated weighted projections are shown in the left and in the right, respectively. Below, all the individual weighted projections $\hat{\phi}$ for all filters (indexed by columns) in each convolutional layer of kernel size k (indexed by rows) are shown. On top of each projection, the value of ω_f (weight connecting the corresponding filter to the output neuron) is shown. Next to this, and in between parentheses, the contribution percentage of ω_f to the convolutional layer and to the full network are displayed. Below, the offset correction value o is exhibited; next to it, in between parentheses, the order in which the projection was added to the cumulative projection is shown.

DRB1-0301

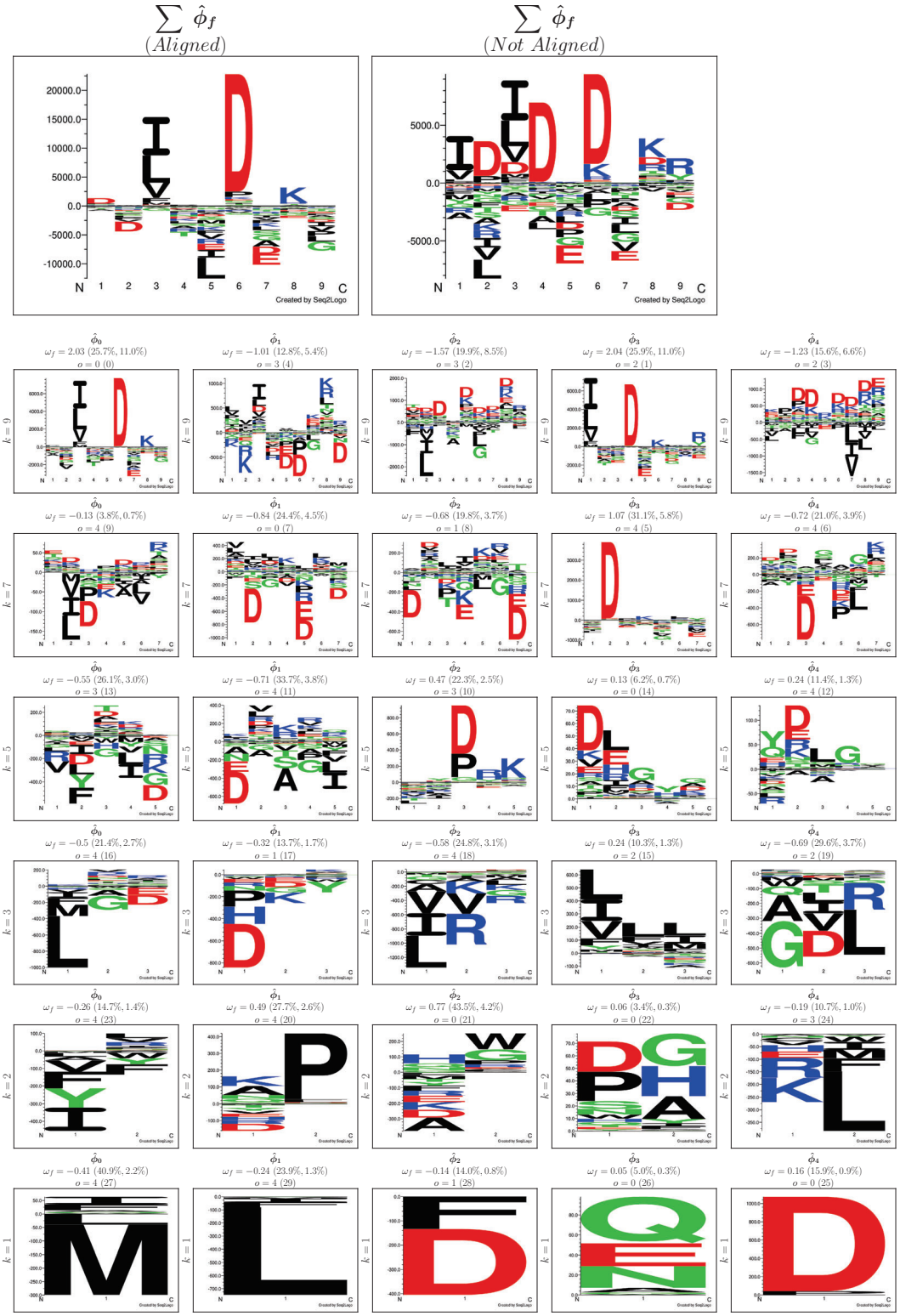

Figure 5.S18. Full projection panel for the HLA-DRB1*03:01 molecule. On top, the aligned and not aligned accumulated weighted projections are shown in the left and in the right, respectively. Below, all the individual weighted projections $\hat{\phi}$ for all filters (indexed by columns) in each convolutional layer of kernel size k (indexed by rows) are shown. On top of each projection, the value of ω_f (weight connecting the corresponding filter to the output neuron) is shown. Next to this, and in between parentheses, the contribution percentage of ω_f to the convolutional layer and to the full network are displayed. Below, the offset correction value o is exhibited; next to it, in between parentheses, the order in which the projection was added to the cumulative projection is shown.

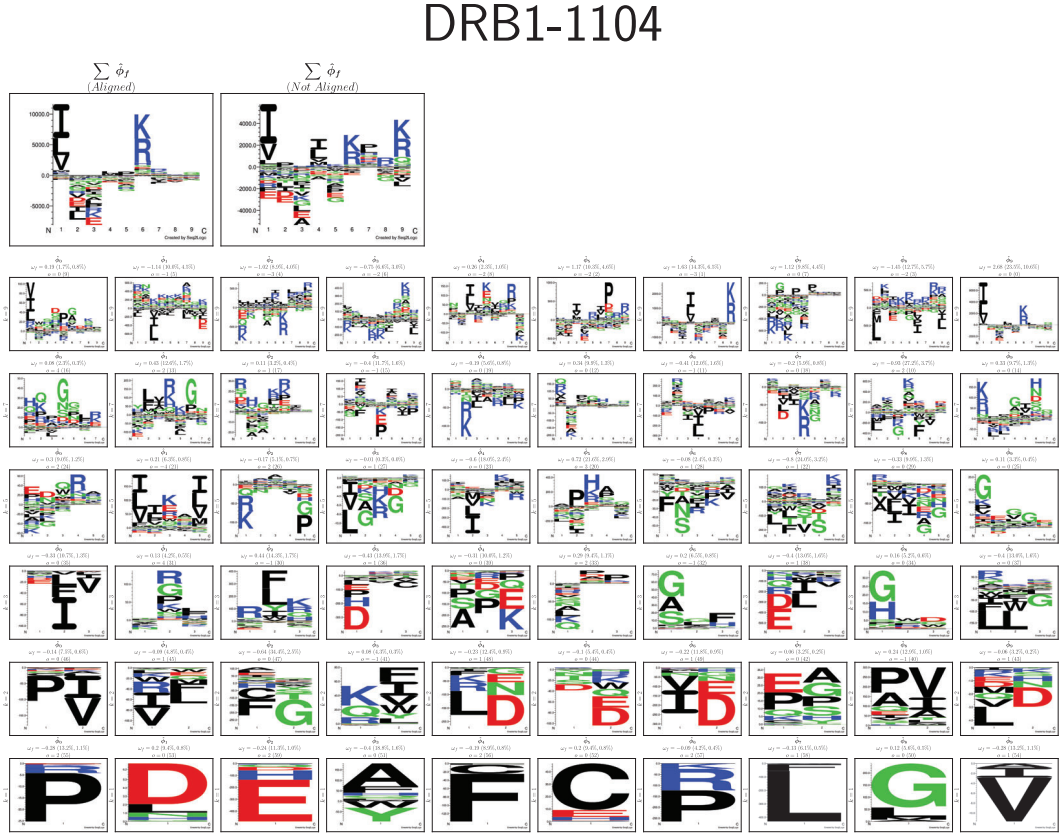

Figure 5.S19. Full projection panel for the HLA-DRB1*11:04 molecule. On top, the aligned and not aligned accumulated weighted projections are shown in the left and in the right, respectively. Below, all the individual weighted projections $\hat{\phi}$ for all filters (indexed by columns) in each convolutional layer of kernel size k (indexed by rows) are shown. On top of each projection, the value of ω_f (weight connecting the corresponding filter to the output neuron) is shown. Next to this, and in between parentheses, the contribution percentage of ω_f to the convolutional layer and to the full network are displayed. Below, the offset correction value o is exhibited; next to it, in between parentheses, the order in which the projection was added to the cumulative projection is shown.

Chapter 6

Epilogue

From a systems biology standpoint, the immune system is a heavily interconnected network of effectors, whose individual actions continuously reshape the topology of the network itself. Additionally, on top of this layer of emergent complexity, evolution has favored graph divergence across individuals, meaning that no immune system is identical to others. Taken together, these characteristics make navigating the immune state of an organism a labyrinthine -and maybe impracticable- task. Nonetheless, such limitations may be leveraged to push the limits of what can be done, with what is available.

Presently, immunopeptidomic techniques have enabled the sampling of a cell's immune state in the form of an immunopeptidome, a more amicable and navigable object. With this, all the work presented in this manuscript was conducted with the drive of pushing forward current capabilities of exploiting such immunopeptidomes, from three angles alike: facilitate the deconvolution of MHC preferences, improve epitope and ligand benchmarks by means of assembling better peptide-MHC binding predictors, and provide such algorithms for the scientific community to use.

With the above in mind, the second chapter of this work dives into 1) deconvoluting immunopeptidomes and 2) training models to predict peptide-MHC binding, as consecutive steps. To do this, we employed data for both the HLA-I and HLA-II systems, from genetically engineered (mono-allelic) and wild type (multi-allelic) cell lines. Considering that binding specificities of MHC molecules are historically defined using data from binding affinity experiments, this chapter provides a first step into what utilizing EL SA and EL MA datasets looks like. For the mono-allelic specimens (only one MHC expressed), the main challenge was to filter spurious sequences (related to MS artifacts), and then generate the corresponding MHC binding motifs from such variable length peptide inputs. For this, GibbsCluster-2.0 was employed, yielding motif logos with increased information content when such filtering was activated (trash cluster enabled). Moreover, for several alleles, sequences found in the trash cluster exhibited terminal lysine/arginine preferences at C-terminus, pointing towards a possible source of wetlab-related, trypsin digestion contamination. The effectiveness of the trash cluster was also explored by means of predicting such spurious sequences against their corresponding MHC restrictions (using the NetMHCpan suite available at the time), where predictions scored low values in comparison to true ligands. In the case of MA cell lines (where a proper deconvolution is needed), GibbsCluster-2.0 also exhibited overall good results, with properly formed deconvoluted motifs. Moreover, in the case of the DR15-DR51 cell line (Class II), only by enabling trash cluster filtering the algorithm correctly converged, showing the importance of not overlooking the effect of noise in our input data. In particular, for the deconvoluted HLA-DRB1*15:01 allele, the GibbsCluster logo exhibited closer similarities to known experimental epitopes than BA-derived motifs. These results hint that MS-derived data has, indeed, the potential of complementing our understanding of peptide properties required for MHC antigen presentation. On the other hand, GibbsCluster failed at fully clustering the HLAs expressed by the HCC1143 cell line (Class I), missing the HLA-C*04:01 allele. Since HLA-C molecules have lower expression values in comparison to their HLA-A and HLA-B counterparts, this imposes a major limitation that should be addressed in future versions of the algorithm. Finally, after GibbsCluster filtering and deconvolution, HLA-I and HLA-II multi-allelic cell lines were employed as training sets to generate peptide-MHC binding predictors, using the NNAlign-2.0 framework. With this, not only the trained models exhibited good cross-validated consistencies (AUC > 0.95 for all six predictors), but binding motifs for deconvoluted alleles and their corresponding peptide length distributions were also successfully recovered from such models.

On the gray side of things, and despite the overall good results shown in chapter 2, all the comparative analyses between deconvoluted motifs (our algorithmic output) and already characterized motifs (ground truth) were done qualitatively by means of visual inspection. This occurs, in principle, because GibbsCluster lacks the capability of automatically annotating the corresponding MHC restrictions to its output, leaving the user with the task of recognizing which MHC corresponds to a given binding preference logo. This limitation is highly detrimental for alleles with unknown binding motifs, and/or projects with large data yields that cannot be processed by humans in a feasible time or without committing several mistakes. Moreover, since a machine learning algorithm cannot be better than the data it is trained on, using GibbsCluster's output to train NNAlign may result, undoubtedly, in subpar predictive models. Because of this, such shortcomings significantly truncate the true potential of MHC motif discovery pipelines.

To circumvent the aforementioned limitations, in the third chapter of this thesis we introduced NNAlign_MA, an augmented version of NNAlign capable of automatically deconvoluting and annotating immunopeptidomes of known MHC typing, while also training a pan-specific model to predict peptide-MHC interactions. This was achieved by means of developing a custom training loop over a two output neuron (accounting for BA and EL data) FFNN architecture. NNAlign_MA was trained on multi-allelic cell lines from the HLA-I, BoLA (bovine MHC-I) and HLA-II systems, and tested on independent ligands and epitopes, in diverse scenarios. In the HLA-I case, all EL MA datasets were completely deconvoluted and annotated to all their HLA restrictions, exhibiting overall good correlation values to known SA motifs, for alleles both characterized and not characterized by SA EL training data. This latter observation shows how co-occurrence and exclusion principle, both exploited by the algorithm, can indeed work together to fish out binding motifs and annotations from MA EL data. This strategy, however, succeeded partially for HLAs present only on single MA EL datasets and for alleles with low quantity of associated peptides (in both cases, binding motifs displayed less informative enrichments). Furthermore, motif consistency for alleles shared between multiple cell lines was also measured, yielding on average PCC values of 0.9. This depicts that, in addition to good correlation with experimental motifs, the method manifests good internal consistency across reported clustering solutions. Clusters of shared alleles also displayed an average PPV of 0.75, meaning a threefold increased likelihood of making a true positive call than a false positive one. With this, NNAlign_MA demonstrates solid conditions for complete, consistent and accurate immunopeptidome deconvolution. Moving on, and to measure the impact of training using EL MA data, performance comparisons were made against a model trained solely on SA data. Here, NNAlign_MA displayed significantly higher performance values when evaluated on EL MA and BA sequences, and a comparable performance on EL SA data. Independent HLA-I epitopes were also tested, with results consistently in favor of NNAlign_MA; in particular, FRANK values were substantially better for epitopes whose alleles were characterized only by EL MA data, confirming again the power of MA data deconvolution. Epitope benchmark was also done with external predictors, displaying small but significant improvements against MHCFlurry_EL and MixMHCpred, and comparable performances against NetMHCpan-4.0 and MHCFlurry. As a final acid test on the class I dataset, single-allelic peptides for specific HLA-I supertypes were removed from the training data, forcing the algorithm to leverage these absent HLA-I motifs purely from multi-allelic data. Results were encouraging, displaying average AUC0.1 values of 0.85 in cross-validation, and one order of magnitude median predictive boost for epitope predictions. After this extensive benchmark on HLA-I data, we moved onto cattle immunity, which represents a more challenging task (data is less abundant, and MHC expression has higher variability). BoLA-I datasets were then employed to train NNAlign_MA, and showed overall good deconvolution shapes. Moreover (and thanks to the score rescaling applied during training) binding motifs reported by the algorithm helped discover a wet lab MHC annotation mistake of a previously published work [167], which was afterwards corrected. A similar case happened with the BoLA-1*00901 molecule, which exhibited a considerably different binding preference in comparison to previous findings. After observing this, collaborators carried out in-vitro binding affinity assays to characterize this molecule, and found a very high similarity with the NNAlign_MA reported motif. These last two results demonstrate, to a high degree, the framework's real capacity of challenging the literature and pushing forward MHC motif discovery. Afterwards, we moved onto evaluating the trained model using a set of experimentally validated BoLA restricted CD8 epitopes, which showed overall comparable performance to NetBoLApan, while also boosting predictive power for BoLA-1*00901 and BoLA-3:01701 epitopes. Having finished with bovine samples, benchmarking the HLA-II system was ensued. For this, we repeated similar analyses to the HLA-I case, with comparable achieved results. Sharp deconvoluted motifs were found, but, as for the class I benchmark, accuracy of identified motifs, also here, depended on the number of ligands assigned to a given HLA. This observation is crucial, as it underlines the dependence of NNAlign_MA on the quality and quantity of input data. Moving on, SA+MA and SA-only models exhibited excellent MA data cross-validation performances, with the main difference being the model trained only on SA modestly (but not significantly) outperforming NNAlign_MA when predicting SA sequences. However, this turned out not to be a major limitation, since CD4 epitope predictions were significantly better for NNAlign_MA in comparison

to the SA model and NetMHCIIpan-3.2.

Considering all the above reported results and observations, we concluded that the NNAlign_MA framework exhibited state-of-the-art performances for MHC motif deconvolution, annotation and epitope predictions. Having culminated the design, train and validation cycles of such a framework, we moved onto deployment and serving. The results of this stage were reported in the fourth chapter of this manuscript, where NNAlign_MA was utilized as training engine for the new versions of NetMHCpan (promoted to 4.1) and NetMHCIIpan (promoted to 4.0), two well established peptide-MHC binding predictors. In the case of MHC-I, and in comparison to the NNAlign_MA benchmark, absolute quantity of training data was extended in 25% (approximately 13.1 million data points), accounting for an allelic coverage expansion of 53%, 78% and 34% for BA, EL SA and EL MA datasets, respectively. NetMHCpan-4.1 was benchmarked against its former version and rival softwares, using independent CD8 epitopes and SA eluted ligands. All these benchmarks resulted in significantly superior performance of NetMHCpan-4.1 versus other methods, with the exception of NetMHCpan-4.0, which resulted in a comparable performance when tested on epitopes. Despite this, consistent improvement was found for HLA-B and HLA-C restricted epitopes. Compared to the NNAlign_MA publication, the sheer amount of training data for MHC-II was increased by 526% (approximately 4.2 million data points), representing an augmented allelic coverage of 34%, 137% and 612% for BA, EL SA and EL MA datasets, respectively. NetMHCIIpan-4.0 was tested on CD4 epitopes and benchmarked against its previous version and other competing algorithms, exhibiting significant performance improvements in all cases. After this benchmarking, both NetMHCpan-4.1 and NetMHCIIpan-4.0 were released to the public in the form of web-servers, whose main characteristics and usage are also described throughout the chapter.

Since such release in May 2020, a lot has happened in the world surrounding SARS-CoV-2. In particular, scientific research played (and still plays) a key role in unpuzzling the novel coronavirus and pushing forward new treatments. Regarding this, in the early stages of the pandemic, the recently updated NetMHCpan suite helped assessing if T cell immunity was indeed achievable in individuals with asymptomatic or mild cases [271], and if a successful vaccine formulation was possible for circulating variants of that moment [272], by means of prioritizing candidate epitopes for tetramer formulation and generating protein hotspot maps, respectively. It followed a proposal for in-silico optimized MHC-I and MHC-II vaccines, with coverages of more than 93% of the world population, where the suite took part in the peptide ranking algorithm core [273]. In the same line of action, a particular case of vaccine for the Colombian population was also formulated [274]. NetMHCpan was also involved in some interesting descriptive studies, such as the association between disease severity and HLA-I genotypes [275] (employed to generate a risk score metric), the assessment of CD8+ T cell immune evasion [276] (used to analyze differences in MHC presentation of ancestral vs. mutated epitopes), and showing how previously exposed individuals develop enhanced immunity against B.1.1.7 and B.1.351 variants with a single BNT162b2 vaccine dose, in comparison to those who were not previously exposed [277] (prioritized HLA-II targets in preliminary studies). Useful online tools have also been created on top of NetMHCpan, such as "neoCOVID Explorer" [278] (accessible through [279]) which enables exploration of predicted Covid-19 epitope landscape for the HLA proteins and prioritization of peptide-MHC pairs for specific populations, and "SARS-CoV-2 T cell epitopes" [280] (accessible through [281]) a curated database of experimentally validated SARS-CoV-2 T cell epitopes compiled from 18 studies of cohorts of recovered patients. An interesting observation on this database is that, from the total 1209 epitopes included, 1017 were pre-selected using NetMHCpan. As good as this may sound for us as developers, it is fair to state that this is also concerning. As of today, there exist several unknown holes and errors in the peptide-MHC mapping space of NetMHCpan, which are still in hiding due to the fact that its predictions tend to be blindly trusted. This lack of inquisitive behaviour perpetuates a cycle which can only be broken through careful examination of cases where the algorithm does not succeed or where different predictors do not arrive at a consensus, with a strong foundation on unbiased experimental validation, as seen in [282]. Such studies will eventually provide the necessary information to compensate for such misbehaviours, leading to better and more comprehensive peptide-MC binding predictions.

Moving on, and having completed the deployment of improved state-of-the-art methods, we left the realm of shallow learning behind and set sail to the uncharted territory of Deep Learning for immunopeptidomics. The results of such a voyage were described in the fifth chapter of this manuscript, where we explored different techniques to navigate peptide-MHC binding data in a completely original way. For this, a novel framework to exploit the sliding dot product of 1D convolutional neural networks was introduced, enabling us to project a given MHC ligand space onto the network's filter space. This resulted in the generation of interesting mathematical objects, termed by us "projections", which were revealed to carry valuable information regarding inner data representations of convolutional filters. The posterior analysis of such projections suggested that the employed CNN was capable of abstracting the input space into smaller, sparse representations of

MHC binding motifs. In addition, output neuron weights acted as a magnitude and sign correction mechanism for such filter representations. Results for six different HLAs were computed, spanning Class I and Class II alleles. For HLA-I, projections exhibited overall high concordances to known experimental binding motifs. However, the HLA-A*01:01 molecule revealed that, in order to fully recover all anchors, an alignment step was required. After developing a concordant alignment procedure, logo representations became sharper and clearer. This procedure proved to be essential to successfully recover the projections of HLA-II molecules HLA-DRB1*03:01 and HLA-DRB1*11:04, whose unaligned projections were considerably noisy. Looking at the full projection panels for the six trained models, it is evident that different levels of length resolution can be achieved by the usage of various convolutional kernel sizes. With longer kernels detecting positional relationships between amino acids, and smaller kernels spiking at uncorrelated positions (but helping to gain information magnitude), the proposed CNN architecture exhibited an exceptional capacity to generate composite, multi-resolution abstractions of sequence patterns, which are ideal for motif discovery.

In closing, there are several ways in which the work presented in this thesis could be continued. First and foremost, incorporation of the T cell side of the problem is paramount. Up to today, peptide-MHC binding predictors are trained purely on MHC binding data, which is necessary for peptide immunogenicity, but not sufficient for it. As new technologies emerge and consolidate, iteration speeds should go up, while prices go down. With this, immunoinformatic pipelines for the high-throughput mining of peptide-MHC-Tcell binding (the golden triad) shall become more abundant and affordable. The result of such technological milestones will be, most likely, an unprecedented capacity to understand and predict immune responses against several pathogens, but also currently impenetrable illnesses such as cancer. Also, in combination with the ever-growing availability of patient-specific therapeutics, all the aforementioned shines a light of hope on disease treatment and prevention. Besides the far-fetched goal of T cell data exploitation, there may exist several other, short-term ways to better draw the boundary separating MHC ligands from non-ligands. Since peptide presentation is the last step of a long antigen processing pathway, further exploration of such pathways for MHC-I and MHC-II may increase the quantity of discriminatory training features (this was explored already for the Class I system in O'Donnell et al. [283], and for Cass II in Barra et al. [215] and Reynisson et al. [253], but still there may be room for further investigation). Other ways of increasing available features may be related to utilizing protein expression measurements (i.e. derived from RNAseq or proteomics data), integrating post-translational modifications, incorporating information associated to the self proteome, or by (somehow) cleverly utilizing epitopes also during the model's training phase. From a machine learning perspective, expanding current approaches to deeper alternatives may be desirable, moreover if we count on the genomical data yields the future seems to offer. Several models have already been proposed in the last years [283–292], but these lack basic rigurosity such as proper data partitioning to reduce homology overlapping, and thus results cannot be trusted. However, a recent publication by Cheng et al. [293] correctly addressed the issue of partition redundancy and implemented BERT (a well known deep ANN architecture from the transformers family) together with multiple instance learning to boost MHC-II binding predictions and fully deconvolute/annotate EL MA data, with promising results that point interesting directions.

Lastly, and by virtue of my advisor's astounding commitment, all the work presented in this thesis is now a running cog inside a big machine that, without a doubt, will keep on getting bigger and bigger. In the meanwhile, I am left with a satisfying, content feeling of having participated in this small contribution to human health and disease research.

✦ ✦ ✦

Bibliography

[1] E. Duan, L. and Mukherjee, *Janeway's Immunobiology*. Yale Journal of Biology and Medicine, 2016. 1, 3, 78

[2] S. C, G. EA, and H. Y, "The Immune System: Basis of so much Health and Disease: 4. Immunocytes," *Dental update*, vol. 44, pp. 436–442, may 2017. 1

[3] R. DB, "V(D)J Recombination: Mechanism, Errors, and Fidelity," *Microbiology spectrum*, vol. 2, nov 2014. 1

[4] D. MM, K. M, H. M, H. J, L. BF, and L. QJ, "T cells as a self-referential, sensory organ," *Annual review of immunology*, vol. 25, pp. 681–695, 2007. 1

[5] B. JA, B. JL, and C. MD, "Differential activation requirements for virgin and memory T cells," *Journal of immunology (Baltimore, Md. : 1950)*, vol. 141, no. 10, pp. 3249–3257, 1988. 1

[6] A. CL, S. L, E. DM, W. MD, and G. JK, "Phagocytosis mediated by three distinct Fc gamma receptor classes on human leukocytes," *The Journal of experimental medicine*, vol. 171, pp. 1333–1345, apr 1990. 2

[7] M. S, T. A, A. B, K. L, and S. F, "Introduction to the Immune System," *Methods in molecular biology (Clifton, N.J.)*, vol. 2024, pp. 1–24, 2019. 2

[8] S. S, Y. T, N. T, and O. M, "Regulatory T cells and immune tolerance," *Cell*, vol. 133, pp. 775–787, may 2008. 2

[9] J. Charles A Janeway, P. Travers, M. Walport, and M. J. Shlomchik, "The major histocompatibility complex and its functions," 2001. 2

[10] T. E, "Immunology," *International forum of allergy & rhinology*, vol. 4 Suppl 2, no. SUPPL.2, 2014. 2

[11] T. N and D. RK, "A brief outline of the immune system," *Methods in molecular biology (Clifton, N.J.)*, vol. 1184, pp. 3–12, 2014. 3

[12] G. JR and W. SM, "Human MHC class III and IV genes and disease associations," *Frontiers in bioscience : a journal and virtual library*, vol. 6, no. 1, p. d960, 2001. 3

[13] L.-B. B and T. R, "The transporter associated with antigen processing TAP: structure and function," *FEBS letters*, vol. 464, pp. 108–112, dec 1999. 3

[14] N. J, J. ML, P. P, and B. O, "Towards a systems understanding of MHC class I and MHC class II antigen presentation," *Nature reviews. Immunology*, vol. 11, pp. 823–836, dec 2011. 3

[15] C. E, V. K, F. MH, L. JP, B. A, H. MP, V. G, R. PP, L. S, T. P, and P. C, "The MHC I immunopeptidome conveys to the cell surface an integrative view of cellular regulation," *Molecular systems biology*, vol. 7, 2011. 3

[16] S. RM, "The dendritic cell system and its role in immunogenicity," *Annual review of immunology*, vol. 9, no. 1, pp. 271–296, 1991. 3

[17] J. Owen, J. Punt, and S. Stranford, *Kuby Immunology 7th edition*. ISBN 13: 9780004236797, 2013. 3

[18] X. H and R. D, "Lysosomal physiology," *Annual review of physiology*, vol. 77, pp. 57–80, feb 2015. 3

[19] W. M, A. ET, S. J, Á.-B. M, S. S, N. F, and F. C, "Major Histocompatibility Complex (MHC) Class I and MHC Class II Proteins: Conformational Plasticity in Antigen Presentation," *Frontiers in immunology*, vol. 8, mar 2017. 3

[20] H. DF, H. RA, S. J, S. K, M. H, S. N, C. AL, A. E, and E. VH, "Characterization of peptides bound to the class I MHC molecule HLA-A2.1 by mass spectrometry," *Science (New York, N.Y.)*, vol. 255, no. 5049, pp. 1261–1263, 1992. 3

[21] B. JH, J. TS, G. JC, S. LJ, U. RG, S. JL, and W. DC, "Three-dimensional structure of the human class II histocompatibility antigen HLA-DR1," *Nature*, vol. 364, no. 6432, pp. 33–39, 1993. 3

[22] D. Sehnal, S. Bittrich, M. Deshpande, R. Svobodová, K. Berka, V. Bazgier, S. Velankar, S. K. Burley, J. Koča, and A. S. Rose, "Mol* Viewer: modern web app for 3D visualization and analysis of large biomolecular structures," *Nucleic Acids Research*, vol. 49, pp. W431–W437, jul 2021. 5

[23] M. M, F. DH, P. PA, and W. IA, "Emerging principles for the recognition of peptide antigens by MHC class I molecules," *Science (New York, N.Y.)*, vol. 257, no. 5072, pp. 927–934, 1992. 3

[24] B. M and W. DC, "Importance of peptide amino and carboxyl termini to the stability of MHC class I molecules," *Science (New York, N.Y.)*, vol. 265, no. 5170, pp. 398–402, 1994. 3

[25] Z. M and S. S, "Conformational flexibility of the MHC class I alpha1-alpha2 domain in peptide bound and free states: a molecular dynamics simulation study," *Biophysical journal*, vol. 87, no. 4, pp. 2203–2214, 2004. 3

[26] C. RM, U. RG, L. WS, G. JC, S. LJ, V. DA, and S. JL, "Predominant naturally processed peptides bound to HLA-DR1 are derived from MHC-related molecules and are heterogeneous in size," *Nature*, vol. 358, no. 6389, pp. 764–768, 1992. 3

[27] F. K, R. O, S. S, J. G, and R. HG, "Allele-specific motifs revealed by sequencing of self-peptides eluted from MHC molecules. 1991," *Journal of immunology (Baltimore, Md. : 1950)*, vol. 177, pp. 2741–7, sep 2006. 3

[28] S. TD and S. RM, "Sequence logos: a new way to display consensus sequences," *Nucleic acids research*, vol. 18, pp. 6097–6100, oct 1990. 3

[29] M. C. F. Thomsen and M. Nielsen, "Seq2Logo: A method for construction and visualization of amino acid binding motifs and sequence profiles including sequence weighting, pseudo counts and two-sided representation of amino acid enrichment and depletion," *Nucleic Acids Research*, vol. 40, jul 2012. 5, 21, 49, 85

[30] R. J, H. JA, H. JD, F. P, P. P, and M. SG, "The IPD and IMGT/HLA database: allele variant databases," *Nucleic acids research*, vol. 43, pp. D423–D431, jan 2015. 4

[31] EMBL-EBI, "IPD-IMGT/HLA database statistics." 4

[32] P. P and O. T, "Population biology of antigen presentation by MHC class I molecules," *Science (New York, N.Y.)*, vol. 272, pp. 67–74, apr 1996. 4

[33] A. A and B.-S. M, "The Human Immunopeptidome Project, a suggestion for yet another postgenome next big thing," *Molecular & cellular proteomics : MCP*, vol. 10, oct 2011. 6

[34] S. Buus, A. Sette, S. M. Colon, C. Miles, and H. M. Grey, "The relation between major histocompatibility complex (MHC) restriction and the capacity of ia to bind immunogenic peptides," *Science*, vol. 235, no. 4794, pp. 1353–1358, 1987. 6, 44

[35] A. Townsend, T. Elliott, V. Cerundolo, L. Foster, B. Barber, and A. Tse, "Assembly of MHC class I molecules analyzed in vitro," *Cell*, vol. 62, pp. 285–295, jul 1990. 6, 44

[36] S. A, S. J, d. G. MF, S. S, R. J, D. C, G. HM, and K. RT, "Peptide binding to the most frequent HLA-A class I alleles measured by quantitative molecular binding assays," *Molecular immunology*, vol. 31, no. 11, pp. 813–822, 1994. 6

[37] R. Aebersold and M. Mann, "Mass spectrometry-based proteomics," *Nature 2003 422:6928*, vol. 422, pp. 198–207, mar 2003. 6

[38] C. C, M. F, P. H, R. J, D. RT, M. M, G. D, C. G, and B.-S. M, "High-throughput and Sensitive Immunopeptidomics Platform Reveals Profound Interferonγ-Mediated Remodeling of the Human Leukocyte Antigen (HLA) Ligandome," *Molecular & cellular proteomics : MCP*, vol. 17, pp. 533–548, mar 2018. 6

[39] A. W. Purcell, S. H. Ramarathinam, and N. Ternette, "Mass spectrometry–based identification of MHC-bound peptides for immunopeptidomics," *Nature Protocols 2019 14:6*, vol. 14, pp. 1687–1707, may 2019. 6

[40] M. Bassani-Sternberg, S. Pletscher-Frankild, L. J. Jensen, and M. Mann, "Mass spectrometry of human leukocyte antigen class i peptidomes reveals strong effects of protein abundance and turnover on antigen presentation," *Molecular and Cellular Proteomics*, vol. 14, pp. 658–673, mar 2015. 6, 29, 31, 33, 44, 45, 46

[41] G. L. Glish and R. W. Vachet, "The basics of mass spectrometry in the twenty-first century," *Nature Reviews Drug Discovery 2003 2:2*, vol. 2, pp. 140–150, feb 2003. 6

[42] Wikipedia, "Lorentz force," *https://en.wikipedia.org/wiki/Lorentz_force*. 6

[43] J. Fenn, M. Mann, C. Meng, S. Wong, and C. Whitehouse, "Electrospray ionization for mass spectrometry of large biomolecules," *Science*, vol. 246, pp. 64–71, oct 1989. 6

[44] M. Bassani-Sternberg and G. Coukos, "Mass spectrometry-based antigen discovery for cancer immunotherapy," aug 2016. 7, 28

[45] C. Ranque, "Electrospray Ionization (ESI) Mass Spectrometry." 7

[46] M. FW, "Tandem mass spectrometry," *Science (New York, N.Y.)*, vol. 214, no. 4518, pp. 280–287, 1981. 7

[47] E. JK, M. AL, and Y. JR, "An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database," *Journal of the American Society for Mass Spectrometry*, vol. 5, no. 11, pp. 976–989, 1994. 7

[48] B. Ma, K. Zhang, C. Hendrie, C. Liang, M. Li, A. Doherty-Kirby, and G. Lajoie, "PEAKS: Powerful software for peptide de novo sequencing by tandem mass spectrometry," *Rapid Communications in Mass Spectrometry*, vol. 17, no. 20, pp. 2337–2342, 2003. 7, 28

[49] K. L, S. JD, M. MJ, and N. WS, "Assigning significance to peptides identified by tandem mass spectrometry using decoy databases," *Journal of proteome research*, vol. 7, pp. 29–34, jan 2008. 7, 8

[50] R. E. Moore, M. K. Young, and T. D. Lee, "Qscore: An algorithm for evaluating SEQUEST database search results," *Journal of the American Society for Mass Spectrometry 2002 13:4*, vol. 13, no. 4, pp. 378–386, 2002. 7

[51] K. AA and M. MJ, "Effects of modified digestion schemes on the identification of proteins from complex mixtures," *Journal of proteome research*, vol. 5, pp. 695–700, mar 2006. 7

[52] J. Colinge, A. Masselot, M. Giron, T. Dessingy, and J. Magnin, "OLAV: Towards high-throughput tandem mass spectrometry data identification," *PROTEOMICS*, vol. 3, pp. 1454–1463, aug 2003. 7

[53] J. G. Abelin, D. Harjanto, M. Malloy, P. Suri, T. Colson, S. P. Goulding, A. L. Creech, L. R. Serrano, G. Nasir, Y. Nasrullah, C. D. McGann, D. Velez, Y. S. Ting, A. Poran, D. A. Rothenberg, S. Chhangawala, A. Rubinsteyn, J. Hammerbacher, R. B. Gaynor, E. F. Fritsch, J. Greshock, R. C. Oslund, D. Barthelme, T. A. Addona, C. M. Arieta, and M. S. Rooney, "Defining HLA-II Ligand Processing and Binding Rules with Mass Spectrometry Enhances Cancer Epitope Prediction," *Immunity*, vol. 51, pp. 766–779.e17, oct 2019. 8, 78, 82, 83

[54] T. M. Mitchell, *Machine Learning: A multistrategy approach*. ISBN: 0-07-042807-7, 1997. 9

[55] A. L. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," *IBM Journal of Research and Development*, vol. 3, pp. 210–229, jul 1959. 9

[56] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to Linear Regression Analysis, 5th Edition*. ISBN: 978-0-470-54281-1, 2012. 9

[57] E. Ostertagová, "Modelling using Polynomial Regression," *Procedia Engineering*, vol. 48, pp. 500–506, jan 2012. 9

[58] D. W. Hosmer, S. Lemeshow, and R. X. Sturdivant, "Applied Logistic Regression: Third Edition," *Applied Logistic Regression: Third Edition*, pp. 1–510, aug 2013. 9

[59] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, pp. 386–408, nov 1958. 9, 10

[60] A. J. Izenman, "Linear Discriminant Analysis," pp. 237–280, 2013. 9

[61] E. Fix and J. L. Hodges, "Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties," *International Statistical Review / Revue Internationale de Statistique*, vol. 57, p. 238, dec 1989. 9

[62] Scribd, "K-Means Clustering," *https://es.scribd.com/document/481282724/K-means-clustering*. 9

[63] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological Cybernetics 1982 43:1*, vol. 43, pp. 59–69, jan 1982. 9

[64] M. Ester, H. Kriegel, J. Sander, and X. Xu, *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*. ISBN: 1-57735-004-9, 1996. 9

[65] F. Nielsen, "Parallel Linear Algebra," pp. 121–145, 2016. 9

[66] S. Russel and P. Norvig, *Artificial Intelligence: A Modern Approach, 3rd Edition*. ISBN: 9780136042594, 2010. 10

[67] M. Mohri, A. Rostamizadeh, and A. Talwalkar, "Foundations in Machine learning," *SpringerBriefs in Computer Science*, vol. 0, no. 9783319056050, pp. 39–44, 2014. 10

[68] G. E. Hinton and T. J. T. J. Sejnowski, "Unsupervised learning : foundations of neural computation," p. 398, 1999. 10

[69] X. Zhu and A. B. Goldberg, "Introduction to Semi-Supervised Learning," *https://doi.org/10.2200/S00196ED1V01Y200906AIM006*, vol. 6, pp. 1–116, jun 2009. 10

[70] B. Baker, I. Kanitscheider, T. Markov, Y. Wu, G. Powell, B. McGrew, and I. Mordatch, "Emergent Tool Use From Multi-Agent Autocurricula," sep 2019. 10

[71] OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang, "Solving Rubik's Cube with a Robot Hand," oct 2019. 10

[72] OpenAI, :, C. Berner, G. Brockman, B. Chan, V. Cheung, P. D biak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. Pachocki, M. Petrov, H. P. d. O. Pinto, J. Raiman, T. Salimans, J. Schlatter, J. Schneider, S. Sidor, I. Sutskever, J. Tang, F. Wolski, and S. Zhang, "Dota 2 with Large Scale Deep Reinforcement Learning," dec 2019. 10, 11

[73] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba, "Mastering Atari with Discrete World Models," oct 2020. 10

[74] N. Tomašev, U. Paquet, D. Hassabis, and V. Kramnik, "Assessing Game Balance with AlphaZero: Exploring Alternative Rule Sets in Chess," ** Equal contribution § Classical*, sep 2020. 10

[75] OpenAI *https://openai.com/*. 10

[76] DeepMind *https://deepmind.com/*. 10

[77] Richard S. Sutton. and Andrew G. Barto., "Reinforcement Learning: An Introduction (Adaptive computation and machine learning)," 1998. 10

[78] K. Zsolnai-Fehér, "OpenAI Plays Hide and Seek…and Breaks The Game!," *https://www.youtube.com/watch?v=Lu56xVlZ40M*. 10

[79] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics 1943 5:4*, vol. 5, pp. 115–133, dec 1943. 10

[80] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, apr 2017. 10

[81] C. Olah, "What are neural nets?," *https://www.youtube.com/watch?v=vdqu6fvjc5c*. 10

[82] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function," *Neural Networks*, vol. 6, pp. 861–867, jan 1993. 10

[83] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for Activation Functions," *6th International Conference on Learning Representations, ICLR 2018 - Workshop Track Proceedings*, oct 2017. 10

[84] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature 2015 521:7553*, vol. 521, pp. 436–444, may 2015. 11, 13

[85] C. Olah, A. Mordvintsev, and L. Schubert, "Feature visualization," *Distill*, 2017. https://distill.pub/2017/feature-visualization. 11

[86] C. Olah, A. Satyanarayan, I. Johnson, S. Carter, L. Schubert, K. Ye, and A. Mordvintsev, "The building blocks of interpretability," *Distill*, 2018. https://distill.pub/2018/building-blocks. 11

[87] G. Goh, N. C. †, C. V. †, S. Carter, M. Petrov, L. Schubert, A. Radford, and C. Olah, "Multimodal neurons in artificial neural networks," *Distill*, 2021. https://distill.pub/2021/multimodal-neurons. 11

[88] R. Q. Quiroga, L. Reddy, G. Kreiman, C. Koch, and I. Fried, "Invariant visual representation by single neurons in the human brain," *Nature 2005 435:7045*, vol. 435, pp. 1102–1107, jun 2005. 11

[89] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, pp. 84–90, jun 2017. 11, 12, 13

[90] C. Farabet, C. Couprie, L. Najman, and Y. Lecun, "Learning hierarchical features for scene labeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1915–1929, 2013. 11

[91] J. Tompson, A. Jain, Y. LeCun, and C. Bregler, "Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation," *Advances in Neural Information Processing Systems*, vol. 2, pp. 1799–1807, jun 2014. 11

[92] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 07-12-June-2015, pp. 1–9, oct 2015. 11, 12

[93] T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. Černocký, "Strategies for training large scale neural network language models," *2011 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2011, Proceedings*, pp. 196–201, 2011. 11

[94] G. Hinton, L. Deng, D. Yu, G. Dahl, A. R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012. 11

[95] T. N. Sainath, A. R. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pp. 8614–8618, oct 2013. 11

[96] A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, A. W. R. Nelson, A. Bridgland, H. Penedones, S. Petersen, K. Simonyan, S. Crossan, P. Kohli, D. T. Jones, D. Silver, K. Kavukcuoglu, and D. Hassabis, "Improved protein structure prediction using potentials from deep learning," *Nature 2020 577:7792*, vol. 577, pp. 706–710, jan 2020. 11

[97] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis, "Highly accurate protein structure prediction with AlphaFold," *Nature 2021*, pp. 1–11, jul 2021. 11

[98] J. Ma, R. P. Sheridan, A. Liaw, G. E. Dahl, and V. Svetnik, "Deep Neural Nets as a Method for Quantitative Structure–Activity Relationships," *Journal of Chemical Information and Modeling*, vol. 55, pp. 263–274, feb 2015. 11

[99] T. Ciodaro, D. Deva, J. M. de Seixas, and D. Damazio, "Online particle detection with Neural Networks based on topological calorimetry information," *Journal of Physics: Conference Series*, vol. 368, p. 012030, jun 2012. 11

[100] Kaggle, "Higgs Boson Machine Learning Challenge," *https://www.kaggle.com/c/higgs-boson*. 11

[101] M. Helmstaedter, K. L. Briggman, S. C. Turaga, V. Jain, H. S. Seung, and W. Denk, "Connectomic reconstruction of the inner plexiform layer in the mouse retina," *Nature 2013 500:7461*, vol. 500, pp. 168–174, aug 2013. 11

[102] D. C. Cires¸ancires¸an, A. Giusti, L. M. Gambardella, and J. . Urgen Schmidhuber, "Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images," 11

[103] C. DC, G. A, G. LM, and S. J, "Mitosis detection in breast cancer histology images with deep neural networks," *Medical image computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention*, vol. 16, no. Pt 2, pp. 411–418, 2013. 11

[104] P. Di Lena, K. Nagata, and P. Baldi, "Deep architectures for protein contact map prediction," *Bioinformatics*, vol. 28, pp. 2449–2457, oct 2012. 11

[105] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, T. Lillicrap, and D. Silver, "Mastering Atari, Go, chess and shogi by planning with a learned model," *Nature 2020 588:7839*, vol. 588, pp. 604–609, dec 2020. 11

[106] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *IEEE Transactions on Neural Networks*, vol. 8, no. 1, pp. 98–113, 1997. 11

[107] C. S. Burrus and T. W. Parks, *DFT/FFT and Convolution Algorithms and Implementation, 1st Edition*. ISBN-13: 978-0824714994, 1985. 12

[108] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very Deep Convolutional Networks for Text Classification," *Nature*, pp. 1–11, jun 2016. 12

[109] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, vol. 1, pp. 541–551, dec 1989. 12

[110] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-December, pp. 770–778, dec 2015. 12

[111] M. Babaee, D. T. Dinh, and G. Rigoll, "A deep convolutional neural network for video sequence background subtraction," *Pattern Recognition*, vol. 76, pp. 635–649, apr 2018. 12

[112] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and F. F. Li, "Large-scale video classification with convolutional neural networks," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1725–1732, sep 2014. 12

[113] C. Angermueller, T. Pärnamaa, L. Parts, and O. Stegle, "Deep learning for computational biology," *Molecular Systems Biology*, vol. 12, p. 878, jul 2016. 12

[114] D. R. Kelley, J. Snoek, and J. L. Rinn, "Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks," *Genome Research*, vol. 26, pp. 990–999, jul 2016. 12

[115] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey, "Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning," *Nature Biotechnology 2015 33:8*, vol. 33, pp. 831–838, jul 2015. 12

[116] J. Zhou and O. G. Troyanskaya, "Predicting effects of noncoding variants with deep learning–based sequence model," *Nature Methods 2015 12:10*, vol. 12, pp. 931–934, aug 2015. 12

[117] C. Angermueller, H. J. Lee, W. Reik, and O. Stegle, "DeepCpG: accurate prediction of single-cell DNA methylation states using deep learning," *Genome Biology 2017 18:1*, vol. 18, pp. 1–13, apr 2017. 12

[118] M. K. K. Leung, H. Y. Xiong, L. J. Lee, and B. J. Frey, "Deep learning of the tissue-regulated splicing code," *Bioinformatics*, vol. 30, pp. i121–i129, 06 2014. 12

[119] X. HY, A. B, L. LJ, B. H, M. D, Y. RK, H. Y, G. S, N. HS, H. TR, M. Q, B. Y, K. AR, J. N, S. SW, B. BJ, and F. BJ, "RNA splicing. The human splicing code reveals new insights into the genetic determinants of disease," *Science (New York, N.Y.)*, vol. 347, jan 2015. 12

[120] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998. 13

[121] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, B. C. Van Esesn, A. A. S. Awwal, and V. K. Asari, "The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches," mar 2018. 13

[122] J. E. Marsden and A. Tromba, *Vector Calculus*. ISBN-13: 978-1429215084, 2011. 13

[123] L. Eon Bottou, "Stochastic Gradient Learning in Neural Networks," 13

[124] S. Ruder, "An overview of gradient descent optimization algorithms," sep 2016. 13

[125] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature 1986 323:6088*, vol. 323, no. 6088, pp. 533–536, 1986. 13

[126] Y. Le Cun, "A Theoretical Framework for Back-Propagation," 1988. 13

[127] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: a survey," *Journal of Machine Learning Research*, vol. 18, pp. 1–43, feb 2015. 13

[128] S. Arlot and A. Celisse, "A survey of cross-validation procedures for model selection," *https://doi.org/10.1214/09-SS054*, vol. 4, pp. 40–79, jan 2010. 14

[129] J. Shao, "Linear model selection by cross-validation," *Journal of the American Statistical Association*, vol. 88, no. 422, pp. 486–494, 1993. 14

[130] DietterichTom, "Overfitting and undercomputing in machine learning," *ACM Computing Surveys (CSUR)*, vol. 27, pp. 326–327, sep 1995. 14

[131] J. Brownlee, "How to use Learning Curves to Diagnose Machine Learning Model Performance." 15

[132] L. Prechelt, "Automatic early stopping using cross validation: quantifying the criteria," *Neural Networks*, vol. 11, pp. 761–767, jun 1998. 15

[133] O. Demir-Kavuk, M. Kamada, T. Akutsu, and E.-W. Knapp, "Prediction using step-wise L1, L2 regularization and feature selection for small data sets with large number of features," *BMC Bioinformatics 2011 12:1*, vol. 12, pp. 1–10, oct 2011. 15

[134] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, pp. 301–320, apr 2005. 15

[135] A. Hernández-García and P. König, "Data augmentation instead of explicit regularization," jun 2018. 15

[136] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," jul 2012. 15

[137] N. Srivastava, G. Hinton, A. Krizhevsky, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014. 15

[138] J. Wainer and G. Cawley, "Nested cross-validation when selecting classifiers is overzealous for most practical applications," *Expert Systems with Applications*, vol. 182, p. 115222, nov 2021. 15

[139] H. Zhang, L. Chen, Y. Qu, G. Zhao, and Z. Guo, "Support vector regression based on grid-search method for short-term wind power forecasting," *Journal of Applied Mathematics*, vol. 2014, 2014. 15

[140] J. Bergstra, J. Bergstra, and Y. Bengio, "Random search for hyper-parameter optimization," *JMLR*, p. 305, 2012. 15

[141] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian Optimization of Machine Learning Algorithms," *Advances in Neural Information Processing Systems*, vol. 4, pp. 2951–2959, jun 2012. 15

[142] M. Claesen and B. De Moor, "Hyperparameter Search in Machine Learning," feb 2015. 15

[143] S. Varma and R. Simon, "Bias in error estimation when using cross-validation for model selection," *BMC Bioinformatics 2006 7:1*, vol. 7, pp. 1–8, feb 2006. 16

[144] C. J. Willmott, "ON THE VALIDATION OF MODELS," *http://dx.doi.org/10.1080/02723646.1981.10642213*, vol. 2, no. 2, pp. 184–194, 2013. 16

[145] K. Pearson, "VII. Mathematical contributions to the theory of evolution.—III. Regression, heredity, and panmixia," *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, vol. 187, pp. 253–318, dec 1896. 16

[146] C. Spearman, "The proof and measurement of association between two things. By C. Spearman, 1904.," *The American journal of psychology*, vol. 100, no. 3-4, pp. 441–471, 1987. 16

[147] I. Rish, "An empirical study of the naive bayes classifier," 2001. 18

[148] W. W. Peterson, T. G. Birdsall, and W. C. Fox, "The theory of signal dectectability," *IRE Professional Group on Information Theory*, vol. 4, no. 4, pp. 171–212, 1954. 18

[149] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," *ACM International Conference Proceeding Series*, vol. 148, pp. 233–240, 2006. 18

[150] J. Huang and C. X. Ling, "Using AUC and accuracy in evaluating learning algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, pp. 299–310, mar 2005. 18

[151] O. Lund, M. Nielsen, C. Lundegaard, C. Kesmir, and S. Brunak, "Immunological Bioinformatics," *Immunological Bioinformatics*, dec 2005. 19

[152] S. Henikoff and J. G. Henikoff, "Amino acid substitution matrices," *Advances in Protein Chemistry*, vol. 54, pp. 73–97, jan 2000. 20

[153] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, jul 1948. 20

[154] N. M, L. C, W. P, L. SL, L. K, B. S, B. S, and L. O, "Reliable prediction of T-cell epitopes using neural networks with novel sequence representations," *Protein science : a publication of the Protein Society*, vol. 12, pp. 1007–1017, may 2003. 20, 83

[155] S. Henikoff and J. G. Henikoff, "Amino acid substitution matrices from protein blocks," *Proceedings of the National Academy of Sciences*, vol. 89, pp. 10915–10919, nov 1992. 20

[156] M. Nielsen, C. Lundegaard, and O. Lund, "Prediction of MHC class II binding affinity using SMM-align, a novel stabilization matrix alignment method," *BMC Bioinformatics*, vol. 8, apr 2007. 21, 48, 84

[157] H. U, S. M, S. R, and S. C, "Selection of representative protein data sets," *Protein science : a publication of the Protein Society*, vol. 1, no. 3, pp. 409–417, 1992. 21

[158] M. Nielsen and O. Lund, "NN-align. An artificial neural network-based alignment algorithm for MHC class II peptide binding prediction," *BMC Bioinformatics*, vol. 10, p. 296, sep 2009. 21, 45

[159] M. Nielsen and M. Andreatta, "NNAlign: A platform to construct and evaluate artificial neural network models of receptor-ligand interactions," *Nucleic Acids Research*, vol. 45, pp. W344–W349, jul 2017. 21, 29, 34, 37, 44, 45, 48, 62, 84

[160] M. Andreatta and M. Nielsen, "Gapped sequence alignment using artificial neural networks: application to the MHC class I system," *Bioinformatics*, vol. 32, pp. 511–517, feb 2016. 21

[161] M. Andreatta, V. I. Jurtz, T. Kaever, A. Sette, B. Peters, and M. Nielsen, "Machine learning reveals a non-canonical mode of peptide binding to MHC class II molecules," *Immunology*, vol. 152, pp. 255–264, oct 2017. 21, 32

[162] V. Jurtz, S. Paul, M. Andreatta, P. Marcatili, B. Peters, and M. Nielsen, "NetMHCpan 4.0: Improved peptide-MHC class I interaction predictions integrating eluted ligand and peptide binding affinity data," jun 2017. 22, 28, 29, 33, 34, 37, 44, 45, 46, 47, 48, 49, 50, 78, 79, 80, 81, 83

[163] K. K. Jensen, M. Andreatta, P. Marcatili, S. Buus, J. A. Greenbaum, Z. Yan, A. Sette, B. Peters, and M. Nielsen, "Improved methods for predicting peptide binding affinity to MHC class II molecules," *Immunology*, vol. 154, pp. 394–406, jul 2018. 22, 46, 81, 83

[164] I. Hoof, B. Peters, J. Sidney, L. E. Pedersen, A. Sette, O. Lund, S. Buus, and M. Nielsen, "NetMHCpan, a method for MHC class i binding prediction beyond humans," *Immunogenetics*, vol. 61, pp. 1–13, jan 2009. 22, 49, 50

[165] M. Andreatta, O. Lund, and M. Nielsen, "Simultaneous alignment and clustering of peptide data using a Gibbs sampling approach," *Bioinformatics*, vol. 29, pp. 8–14, jan 2013. 22, 29, 45

[166] M. Andreatta, B. Alvarez, and M. Nielsen, "GibbsCluster: Unsupervised clustering and alignment of peptide sequences," *Nucleic Acids Research*, vol. 45, pp. W458–W463, jul 2017. 22, 29, 37, 45

[167] M. Nielsen, T. Connelley, and N. Ternette, "Improved Prediction of Bovine Leucocyte Antigens (BoLA) Presented Ligands by Use of Mass-Spectrometry-Determined Ligand and in Vitro Binding Data," *Journal of Proteome Research*, vol. 17, pp. 559–567, jan 2018. 23, 31, 45, 46, 47, 48, 57, 58, 59, 78, 114

[168] E. Caron, D. J. Kowalewski, C. C. Koh, T. Sturm, H. Schuster, and R. Aebersold, "Analysis of major histocompatibility complex (MHC) immunopeptidomes using mass spectrometry," dec 2015. 28, 44

[169] M. Bassani-Sternberg and D. Gfeller, "Unsupervised HLA Peptidome Deconvolution Improves Ligand Prediction Accuracy and Predicts Cooperative Effects in Peptide–HLA Interactions," *The Journal of Immunology*, vol. 197, pp. 2492–2499, sep 2016. 28, 29, 33, 45, 78

[170] J. G. Abelin, D. B. Keskin, S. Sarkizova, C. R. Hartigan, W. Zhang, J. Sidney, J. Stevens, W. Lane, G. L. Zhang, T. M. Eisenhaure, K. R. Clauser, N. Hacohen, M. S. Rooney, S. A. Carr, and C. J. Wu, "Mass Spectrometry Profiling of HLA-Associated Peptidomes in Monoallelic Cells Enables More Accurate Epitope Prediction," *Immunity*, vol. 46, pp. 315–326, feb 2017. 28, 29, 30, 33, 37, 44, 45, 78, 82

[171] N. Ternette, H. Yang, T. Partridge, A. Llano, S. Cedeño, R. Fischer, P. D. Charles, N. L. Dudek, B. Mothe, M. Crespo, W. M. Fischer, B. T. Korber, M. Nielsen, P. Borrow, A. W. Purcell, C. Brander, L. Dorrell, B. M. Kessler, and T. Hanke, "Defining the HLA class I-associated viral antigen repertoire from HIV-1-infected human cells," *European Journal of Immunology*, vol. 46, pp. 60–69, jan 2016. 28

[172] J. C. Yaciuk, M. Skaley, W. Bardet, F. Schafer, D. Mojsilovic, S. Cate, C. J. Stewart, C. McMurtrey, K. W. Jackson, R. Buchli, A. Olvera, S. Cedeno, M. Plana, B. Mothe, C. Brander, J. T. West, and W. H. Hildebrand, "Direct Interrogation of Viral Peptides Presented by the Class I HLA of HIV-Infected T Cells," *Journal of Virology*, vol. 88, pp. 12992–13004, nov 2014. 28

[173] C. Berlin, D. J. Kowalewski, H. Schuster, N. Mirza, S. Walz, M. Handel, B. Schmid-Horch, H. R. Salih, L. Kanz, H. G. Rammensee, S. Stevanović, and J. S. Stickel, "Mapping the HLA ligandome landscape of acute myeloid leukemia: a targeted approach toward peptide-based immunotherapy," *Leukemia*, vol. 29, pp. 647–659, mar 2015. 28

[174] S. Kalaora, E. Barnea, E. Merhavi-Shoham, N. Qutob, J. K. Teer, N. Shimony, J. Schachter, S. A. Rosenberg, M. J. Besser, A. Admon, and Y. Samuels, "Use of HLA peptidomics and whole exome sequencing to identify human immunogenic neo-antigens," *Oncotarget*, vol. 7, no. 5, pp. 5110–5117, 2016. 28

[175] M. Bassani-Sternberg, E. Bräunlein, R. Klar, T. Engleitner, P. Sinitcyn, S. Audehm, M. Straub, J. Weber, J. Slotta-Huspenina, K. Specht, M. E. Martignoni, A. Werner, R. Hein, D. H. Busch, C. Peschel, R. Rad, J. Cox, M. Mann, and A. M. Krackhardt, "Direct identification of clinically relevant neoepitopes presented on native human melanoma tissue by mass spectrometry," *Nature Communications*, vol. 7, nov 2016. 28, 29, 45, 46

[176] J. Cox and M. Mann, "MaxQuant enables high peptide identification rates, individualized p.p.b.-range mass accuracies and proteome-wide protein quantification," *Nature Biotechnology*, vol. 26, pp. 1367–1372, dec 2008. 28

[177] J. Zhang, L. Xin, B. Shan, W. Chen, M. Xie, D. Yuen, W. Zhang, Z. Zhang, G. A. Lajoie, and B. Ma, "PEAKS DB: De novo sequencing assisted database search for sensitive and accurate peptide identification," *Molecular and Cellular Proteomics*, vol. 11, apr 2012. 28

[178] M. Brosch, L. Yu, T. Hubbard, and J. Choudhary, "Accurate and sensitive peptide identification with mascot percolator," *Journal of Proteome Research*, vol. 8, pp. 3176–3181, jun 2009. 28

[179] J. P. Murphy, P. Konda, D. J. Kowalewski, H. Schuster, D. Clements, Y. Kim, A. M. Cohen, T. Sharif, M. Nielsen, S. Stevanovic, P. W. Lee, and S. Gujar, "MHC-I Ligand Discovery Using Targeted Database Searches of Mass Spectrometry Data: Implications for T-Cell Immunotherapies," *Journal of Proteome Research*, vol. 16, pp. 1806–1816, apr 2017. 28, 45

[180] T. Delong, T. A. Wiles, R. L. Baker, B. Bradley, G. Barbour, R. Reisdorph, M. Armstrong, R. L. Powell, N. Reisdorph, N. Kumar, C. M. Elso, M. DeNicola, R. Bottino, A. C. Powers, D. M. Harlan, S. C. Kent, S. I. Mannering, and K. Haskins, "Pathogenic CD4 T cells in type 1 diabetes recognize epitopes formed by peptide fusion," *Science*, vol. 351, pp. 711–714, feb 2016. 28

[181] J. Liepe, F. Marino, J. Sidney, A. Jeko, D. E. Bunting, A. Sette, P. M. Kloetzel, M. P. Stumpf, A. J. Heck, and M. Mishto, "A large fraction of HLA class I ligands are proteasome-generated spliced peptides," *Science*, vol. 354, pp. 354–358, oct 2016. 28

[182] H. G. Rammensee, T. Friede, and S. Stevanović, "MHC ligands and peptide motifs: first listing," feb 1995. 28

[183] P. Probst, J. Kopp, A. Oxenius, M. P. Colombo, D. Ritz, T. Fugmann, and D. Neri, "Sarcoma eradication by doxorubicin and targeted TNF relies upon CD8+ T-cell recognition of a retroviral antigen," *Cancer Research*, vol. 77, pp. 3644–3654, jul 2017. 29

[184] D. Ritz, A. Gloger, B. Weide, C. Garbe, D. Neri, and T. Fugmann, "High-sensitivity HLA class I peptidome analysis enables a precise definition of peptide motifs and the identification of peptides from cell lines and patients' sera," *Proteomics*, vol. 16, pp. 1570–1580, may 2016. 29, 33, 45, 46

[185] A. Sofron, D. Ritz, D. Neri, and T. Fugmann, "High-resolution analysis of the murine MHC class II immunopeptidome," *European Journal of Immunology*, vol. 46, pp. 319–328, feb 2016. 29, 45, 46, 83

[186] T. Fugmann, A. Sofron, D. Ritz, F. Bootz, and D. Neri, "The MHC Class II Immunopeptidome of Lymph Nodes in Health and in Chemically Induced Colitis," *The Journal of Immunology*, vol. 198, pp. 1357–1364, feb 2017. 29

[187] G. P. Mommen, F. Marino, H. D. Meiring, M. C. Poelen, J. A. Van Gaans-van Den Brink, S. Mohammed, A. J. Heck, and C. A. Van Els, "Sampling from the proteome to the human leukocyte antigen-DR (HLA-DR) ligandome proceeds via high specificity," *Molecular and Cellular Proteomics*, vol. 15, pp. 1412–1423, apr 2016. 29, 44, 45

[188] M. Bassani-Sternberg, C. Chong, P. Guillaume, M. Solleder, H. S. Pak, P. O. Gannon, L. E. Kandalaft, G. Coukos, and D. Gfeller, "Deciphering HLA-I motifs across HLA peptidomes improves neo-antigen predictions and identifies allostery regulating HLA specificity," *PLoS computational biology*, vol. 13, p. e1005725, aug 2017. 29, 31, 37, 45, 46, 50, 52, 78, 81

[189] M. Nielsen, T. Connelley, and N. Ternette, "Improved prediction of Bovine Leucocyte Antigens (BoLA) presented ligands by use of MS eluted ligands and in-vitro binding data; impact for the identification T cell epitopes," sep 2017. 29

[190] S. Tenzer, B. Peters, S. Bulik, O. Schoor, C. Lemmel, M. M. Schatz, P. M. Kloetzel, H. G. Rammensee, H. Schild, and H. G. Holzhütter, "Modeling the MHC class I pathway by combining predictions of proteasomal cleavage, TAP transport and MHC class I binding," *Cellular and Molecular Life Sciences*, vol. 62, pp. 1025–1037, may 2005. 29

[191] M. Harndahl, M. Rasmussen, G. Roder, I. Dalgaard Pedersen, M. Sørensen, M. Nielsen, and S. Buus, "Peptide-MHC class I stability is a better predictor than peptide affinity of CTL immunogenicity," *European Journal of Immunology*, vol. 42, pp. 1405–1416, jun 2012. 29

[192] M. Andreatta, C. Schafer-Nielsen, O. Lund, S. Buus, and M. Nielsen, "NNAlign: A web-based prediction method allowing non-expert end-user discovery of sequence motifs in quantitative peptide data," *PLoS ONE*, vol. 6, nov 2011. 29, 33, 45

[193] E. E. Sercarz and E. Maverakis, "MHC-guided processing: Binding of large antigen fragments," 2003. 30

[194] K. Hodge, S. T. Have, L. Hutton, and A. I. Lamond, "Cleaning up the masses: Exclusion lists to reduce contamination with HPLC-MS/MS," *Journal of Proteomics*, vol. 88, pp. 92–103, nov 2013. 30, 37

[195] D. Mellacheruvu, Z. Wright, A. L. Couzens, J. P. Lambert, N. A. St-Denis, T. Li, Y. V. Miteva, S. Hauri, M. E. Sardiu, T. Y. Low, V. A. Halim, R. D. Bagshaw, N. C. Hubner, A. Al-Hakim, A. Bouchard, D. Faubert, D. Fermin, W. H. Dunham, M. Goudreault, Z. Y. Lin, B. G. Badillo, T. Pawson, D. Durocher, B. Coulombe, R. Aebersold, G. Superti-Furga, J. Colinge, A. J. Heck, H. Choi, M. Gstaiger, S. Mohammed, I. M. Cristea, K. L. Bennett, M. P. Washburn, B. Raught, R. M. Ewing, A. C. Gingras, and A. I. Nesvizhskii, "The CRAPome: A contaminant repository for affinity purification-mass spectrometry data," *Nature Methods*, vol. 10, pp. 730–736, aug 2013. 30, 37

[196] M. Nielsen and M. Andreatta, "NetMHCpan-3.0; improved prediction of binding to MHC class I molecules integrating information from multiple receptor and peptide length datasets," *Genome Medicine*, vol. 8, mar 2016. 30, 31, 78

[197] M. Rasmussen, M. Harndahl, A. Stryhn, R. Boucherma, L. L. Nielsen, F. A. Lemonnier, M. Nielsen, and S. Buus, "Uncovering the Peptide-Binding Specificities of HLA-C: A General Strategy To Determine the Specificity of Any MHC Class I Molecule," *The Journal of Immunology*, vol. 193, pp. 4790–4802, nov 2014. 31, 47, 53

[198] M. Nielsen, O. Lund, S. Buus, and C. Lundegaard, "MHC Class II epitope predictive algorithms," jul 2010. 32

[199] M. Nielsen, C. Lundegaard, P. Worning, C. Sylvester Hvid, K. Lamberth, S. Buus, S. Brunak, and O. Lund, "Improved prediction of MHC class I and class II epitopes using a novel Gibbs sampling approach," *Bioinformatics*, vol. 20, pp. 1388–1397, jun 2004. 32

[200] T. Sturniolo, E. Bono, J. Ding, L. Raddrizzani, O. Tuereci, U. Sahin, M. Braxenthaler, F. Gallazzi, M. P. Protti, F. Sinigaglia, and J. Hammer, "Generation of tissue-specific and promiscuous HLA ligand databases using DNA microarrays and virtual HLA class II matrices," *Nature Biotechnology*, vol. 17, pp. 555–561, jun 1999. 32

[201] M. Andreatta and M. Nielsen, "Characterizing the binding motifs of 11 common human HLA-DP and HLA-DQ molecules using NNAlign," *Immunology*, vol. 136, pp. 306–311, jul 2012. 32

[202] J. D. Ooi, J. Petersen, Y. H. Tan, M. Huynh, Z. J. Willett, S. H. Ramarathinam, P. J. Eggenhuizen, K. L. Loh, K. A. Watson, P. Y. Gan, M. A. Alikhan, N. L. Dudek, A. Handel, B. G. Hudson, L. Fugger, D. A. Power, S. G. Holt, P. T. Coates, J. W. Gregersen, A. W. Purcell, S. R. Holdsworth, N. L. La Gruta, H. H. Reid, J. Rossjohn, and A. R. Kitching, "Dominant protection from HLA-linked autoimmunity by antigen-specific regulatory T cells," *Nature*, vol. 545, pp. 243–247, may 2017. 32, 35, 46, 83

[203] A. B. Vogt, H. Kropshofer, H. Kalbacher, M. Kalbus, H. G. Rammensee, J. E. Coligan, and R. Martin, "Ligand motifs of HLA-DRB5*0101 and DRB1*1501 molecules delineated from self-peptides.," *The Journal of Immunology*, vol. 153, no. 4, 1994. 33

[204] E. M. Scholz, M. Marcilla, X. Daura, D. Arribas-Layton, E. A. James, and I. Alvarez, "Human leukocyte antigen (hla)-DrB1*15:01 and hla-DrB5*01:01 present complementary peptide repertoires," *Frontiers in Immunology*, vol. 8, aug 2017. 33

[205] M. Andreatta, E. Karosiene, M. Rasmussen, A. Stryhn, S. Buus, and M. Nielsen, "Accurate pan-specific prediction of peptide-MHC class II binding affinity with improved binding core identification," *Immunogenetics*, vol. 67, pp. 641–650, sep 2015. 33

[206] R. Vita, J. A. Overton, J. A. Greenbaum, J. Ponomarenko, J. D. Clark, J. R. Cantrell, D. K. Wheeler, J. L. Gabbard, D. Hix, A. Sette, and B. Peters, "The immune epitope database (IEDB) 3.0," *Nucleic Acids Research*, vol. 43, pp. D405–D412, jan 2015. 36

[207] C. A. van Els, V. Corbière, K. Smits, J. A. van Gaans-van den Brink, M. C. Poelen, F. Mascart, H. D. Meiring, and C. Locht, "Toward understanding the essence of post-translational modifications for the Mycobacterium tuberculosis immunoproteome," *Frontiers in Immunology*, vol. 5, no. AUG, pp. 1–10, 2014. 37

[208] S. Giguère, A. Drouin, A. Lacoste, M. Marchand, J. Corbeil, and F. Laviolette, "MHC-NP: Predicting peptides naturally processed by the MHC," *Journal of Immunological Methods*, vol. 400-401, no. 1, pp. 30–36, 2013. 37

[209] E. Caron, R. Aebersold, A. Banaei-Esfahani, C. Chong, and M. Bassani-Sternberg, "A Case for a Human Immuno-Peptidome Project Consortium," in *Immunity*, vol. 47, pp. 203–208, Cell Press, aug 2017. 44

[210] T. Trolle, C. P. McMurtrey, J. Sidney, W. Bardet, S. C. Osborn, T. Kaever, A. Sette, W. H. Hildebrand, M. Nielsen, and B. Peters, "The Length Distribution of Class I–Restricted T Cell Epitopes Is Determined by Both Peptide Supply and MHC Allele–Specific Binding Preference," *The Journal of Immunology*, vol. 196, pp. 1480–1487, feb 2016. 44

[211] D. Gfeller, P. Guillaume, J. Michaux, H.-S. Pak, R. T. Daniel, J. Racle, G. Coukos, and M. Bassani-Sternberg, "The Length Distribution and Multiple Specificity of Naturally Presented HLA-I Ligands," *The Journal of Immunology*, vol. 201, pp. 3705–3716, dec 2018. 44, 81

[212] A. Sette, L. Adorini, S. M. Colon, S. Buus, and H. M. Grey, "Capacity of intact proteins to bind to MHC class II molecules.," *The Journal of Immunology*, vol. 143, no. 4, 1989. 44

[213] S. Sadegh-Nasseri and A. R. Kim, "MHC class II auto-antigen presentation is unconventional," 2015. 44

[214] D. B. Graham, C. Luo, D. J. O'Connell, A. Lefkovith, E. M. Brown, M. Yassour, M. Varma, J. G. Abelin, K. L. Conway, G. J. Jasso, C. G. Matar, S. A. Carr, and R. J. Xavier, "Antigen discovery and specification of immunodominance hierarchies for MHCII-restricted epitopes," *Nature Medicine*, vol. 24, pp. 1762–1772, nov 2018. 44

[215] C. Barra, B. Alvarez, S. Paul, A. Sette, B. Peters, M. Andreatta, S. Buus, and M. Nielsen, "Footprints of antigen processing boost MHC class II natural ligand predictions," *Genome Medicine*, vol. 10, nov 2018. 44, 47, 48, 62, 78, 79, 81, 84, 85, 102, 116

[216] C. I. DeVette, M. Andreatta, W. Bardet, S. J. Cate, V. I. Jurtz, K. W. Jackson, A. L. Welm, M. Nielsen, and W. H. Hildebrand, "NetH2pan: A computational tool to guide MHC peptide prediction on murine tumors," *Cancer Immunology Research*, vol. 6, pp. 636–644, jun 2018. 45, 83

[217] K. Prilliman, M. Lindsey, Y. Zuo, K. W. Jackson, Y. Zhang, and W. Hildebrand, "Large-scale production of class I bound peptides: Assigning a signature to HLA-B(*)1501," *Immunogenetics*, vol. 45, no. 6, pp. 379–385, 1997. 45

[218] K. Falk, O. Rötzschke, S. Stevanović, G. Jung, and H. G. Rammensee, "Allele-specific motifs revealed by sequencing of self-peptides eluted from MHC molecules," *Nature*, vol. 351, no. 6324, pp. 290–296, 1991. 45

[219] R. B. Schittenhelm, N. L. Dudek, N. P. Croft, S. H. Ramarathinam, and A. W. Purcell, "A comprehensive analysis of constitutive naturally processed and presented HLA-C*04:01 (Cw4) - specific peptides," *Tissue Antigens*, vol. 83, pp. 174–179, mar 2014. 45

[220] B. Alvarez, C. Barra, M. Nielsen, and M. Andreatta, "Computational Tools for the Identification and Interpretation of Sequence Motifs in Immunopeptidomes," jun 2018. 45, 48, 49, 78, 84

[221] H. Pearson, T. Daouda, D. P. Granados, C. Durette, E. Bonneil, M. Courcelles, A. Rodenbrock, J. P. Laverdure, C. Côté, S. Mader, S. Lemieux, P. Thibault, and C. Perreault, "MHC class I-associated peptides derive from selective regions of the human genome," *Journal of Clinical Investigation*, vol. 126, pp. 4690–4701, dec 2016. 46

[222] B. Shraibman, D. M. Kadosh, E. Barnea, and A. Admon, "Human leukocyte antigen (HLA) peptides derived from tumor antigens induced by inhibition of DNA methylation for development of drug-facilitated immunotherapy," *Molecular and Cellular Proteomics*, vol. 15, pp. 3058–3070, sep 2016. 46, 47

[223] G. P. Mommen, C. K. Frese, H. D. Meiring, J. Gaans-van Den Brink, A. P. De Jong, C. A. Van Els, and A. J. Heck, "Expanding the detectable HLA peptide repertoire using electron-transfer/ higher-energy collision dissociation (EThcD)," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 111, pp. 4507–4512, mar 2014. 46

[224] A. Gloger, D. Ritz, T. Fugmann, and D. Neri, "Mass spectrometric analysis of the HLA class I peptidome of melanoma cell lines as a promising tool for the identification of putative tumor-associated HLA epitopes," *Cancer Immunology, Immunotherapy*, vol. 65, pp. 1377–1393, nov 2016. 46

[225] R. Vita, S. Mahajan, J. A. Overton, S. K. Dhanda, S. Martini, J. R. Cantrell, D. K. Wheeler, A. Sette, and B. Peters, "The Immune Epitope Database (IEDB): 2018 update," *Nucleic Acids Research*, vol. 47, no. D1, pp. D339–D343, 2019. 46, 52, 60, 64, 83

[226] C. C. Clement, A. Becerra, L. Yin, V. Zolla, L. Huang, S. Merlin, A. Follenzi, S. A. Shaffer, L. J. Stern, and L. Santambrogio, "The dendritic cell Major Histocompatibility Complex II (MHC II) peptidome derives from a variety of processing pathways and includes peptides with a broad spectrum of HLA-DM sensitivity," mar 2016. 46, 83

[227] E. Bergseng, S. Dørum, M. Arntzen, M. Nielsen, S. Nygård, S. Buus, G. A. De Souz, and L. M. Sollid, "Different binding motifs of the celiac disease-associated HLA molecules DQ2.5, DQ2.2, and DQ7.5 revealed by relative quantitative proteomics of endogenous peptide repertoires," *Immunogenetics*, vol. 67, pp. 73–84, feb 2015. 46, 83

[228] T. Heyder, M. Kohler, N. K. Tarasova, S. Haag, D. Rutishauser, N. V. Rivera, C. Sandin, S. Mia, V. Malmström, Å. M. Wheelock, J. Wahlström, R. Holmdahl, A. Eklund, R. A. Zubarev, J. Grunewald, and A. Jimmy Ytterberg, "Approach for identifying Human Leukocyte Antigen (HLA)-DR bound peptides from scarce clinical samples," *Molecular and Cellular Proteomics*, vol. 15, pp. 3017–3029, sep 2016. 46, 83

[229] A. Nelde, D. J. Kowalewski, L. Backert, H. Schuster, J. O. Werner, R. Klein, O. Kohlbacher, L. Kanz, H. R. Salih, H. G. Rammensee, S. Stevanović, and J. S. Walz, "HLA ligandome analysis of primary chronic lymphocytic leukemia (CLL) cells under lenalidomide treatment confirms the suitability of lenalidomide for combination with T-cell-based immunotherapy," *OncoImmunology*, vol. 7, apr 2018. 46, 83

[230] K. P. Karunakaran, H. Yu, X. Jiang, Q. Chan, M. F. Goldberg, M. K. Jenkins, L. J. Foster, and R. C. Brunham, "Identification of MHC-Bound Peptides from Dendritic Cells Infected with Salmonella enterica Strain SL1344: Implications for a Nontyphoidal Salmonella Vaccine," *Journal of Proteome Research*, vol. 16, pp. 298–306, jan 2017. 46, 83

[231] Y. T. Ting, J. Petersen, S. H. Ramarathinam, S. W. Scally, K. L. Loh, R. Thomas, A. Suri, D. G. Baker, A. W. Purcell, H. H. Reid, and J. Rossjohn, "The interplay between citrullination and HLA-DRB1 polymorphism in shaping peptide binding hierarchies in rheumatoid arthritis," *Journal of Biological Chemistry*, vol. 293, pp. 3236–3251, mar 2018. 46, 83

[232] Q. Wang, E. E. Drouin, C. Yao, J. Zhang, Y. Huang, D. R. Leon, A. C. Steere, and C. E. Costello, "Immunogenic HLA-DR-Presented Self-Peptides Identified Directly from Clinical Samples of Synovial Tissue, Synovial Fluid, or Peripheral Blood in Patients with Rheumatoid Arthritis or Lyme Arthritis," *Journal of Proteome Research*, vol. 16, pp. 122–136, jan 2017. 46, 83

[233] Y. Perez-Riverol, A. Csordas, J. Bai, M. Bernal-Llinares, S. Hewapathirana, D. J. Kundu, A. Inuganti, J. Griss, G. Mayer, M. Eisenacher, E. Pérez, J. Uszkoreit, J. Pfeuffer, T. Sachsenberg, . Yilmaz, S. Tiwary, J. Cox, E. Audain, M. Walzer, A. F. Jarnuczak, T. Ternent,

A. Brazma, and J. A. Vizcaíno, "The PRIDE database and related tools and resources in 2019: Improving support for quantification data," *Nucleic Acids Research*, vol. 47, pp. D442–D450, jan 2019. 46

[234] A. M. Hansen, M. Rasmussen, N. Svitek, M. Harndahl, W. T. Golde, J. Barlow, V. Nene, S. Buus, and M. Nielsen, "Characterization of binding specificities of bovine leucocyte class I molecules: impacts for rational epitope discovery," *Immunogenetics*, vol. 66, pp. 705–718, dec 2014. 47

[235] L. E. Pedersen, M. Harndahl, M. Rasmussen, K. Lamberth, W. T. Golde, O. Lund, M. Nielsen, and S. Buus, "Porcine major histocompatibility complex (MHC) class i molecules and analysis of their peptide-binding specificities," *Immunogenetics*, vol. 63, pp. 821–834, dec 2011. 47

[236] M. Harndahl, M. Rasmussen, G. Roder, and S. Buus, "Real-time, high-throughput measurements of peptide-MHC-I dissociation using a scintillation proximity assay," *Journal of Immunological Methods*, vol. 374, pp. 5–12, nov 2011. 47

[237] O. Lund, M. Nielsen, C. Kesmir, A. G. Petersen, C. Lundegaard, P. Worning, C. Sylvester-Hvid, K. Lamberth, G. Røder, S. Justesen, S. Buus, and S. Brunak, "Definition of supertypes for HLA molecules using clustering of specificity matrices," *Immunogenetics*, vol. 55, pp. 797–810, mar 2004. 49, 55

[238] E. Karosiene, C. Lundegaard, O. Lund, and M. Nielsen, "NetMHCcons: A consensus method for the major histocompatibility complex class i predictions," *Immunogenetics*, vol. 64, pp. 177–186, mar 2012. 55, 61

[239] M. Thomsen, C. Lundegaard, S. Buus, O. Lund, and M. Nielsen, "MHCcluster, a method for functional clustering of MHC molecules," *Immunogenetics*, vol. 65, pp. 655–665, sep 2013. 56

[240] D. Vasoya, A. Law, P. Motta, M. Yu, A. Muwonge, E. Cook, X. Li, K. Bryson, A. MacCallam, T. Sitt, P. Toye, B. Bronsvoort, M. Watson, W. I. Morrison, and T. Connelley, "Rapid identification of bovine MHCI haplotypes in genetically divergent cattle populations using next-generation sequencing," *Immunogenetics*, vol. 68, pp. 765–781, nov 2016. 57, 71

[241] V. Nene, N. Svitek, P. Toye, W. T. Golde, J. Barlow, M. Harndahl, S. Buus, and M. Nielsen, "Designing bovine T cell vaccines via reverse immunology," jun 2012. 59

[242] J. Hart, N. D. MacHugh, T. Sheldrake, M. Nielsen, and W. Ivan Morrison, "Identification of immediate early gene products of bovine herpes virus 1 (BHV-1) as dominant antigens recognized by CD8 T cells in immune cattle," *Journal of General Virology*, vol. 98, pp. 1843–1854, jul 2017. 59

[243] J. Robinson, A. R. Soormally, J. D. Hayhurst, and S. G. Marsh, "The IPD-IMGT/HLA Database - New developments in reporting HLA variation," *Human Immunology*, vol. 77, pp. 233–237, mar 2016. 61

[244] B. Bulik-Sullivan, J. Busby, C. D. Palmer, M. J. Davis, T. Murphy, A. Clark, M. Busby, F. Duke, A. Yang, L. Young, N. C. Ojo, K. Caldwell, J. Abhyankar, T. Boucher, M. G. Hart, V. Makarov, V. T. De Montpreville, O. Mercier, T. A. Chan, G. Scagliotti, P. Bironzo, S. Novello, N. Karachaliou, R. Rosell, I. Anderson, N. Gabrail, J. Hrom, C. Limvarapuss, K. Choquette, A. Spira, R. Rousseau, C. Voong, N. A. Rizvi, E. Fadel, M. Frattini, K. Jooss, M. Skoberne, J. Francis, and R. Yelensky, "Deep learning using tumor HLA peptide mass spectrometry datasets improves neoantigen identification," *Nature Biotechnology*, vol. 37, pp. 55–71, jan 2019. 61, 62, 78, 83

[245] J. Racle, J. Michaux, G. A. Rockinger, M. Arnaud, S. Bobisse, C. Chong, P. Guillaume, G. Coukos, A. Harari, C. Jandus, M. Bassani-Sternberg, and D. Gfeller, "Robust prediction of HLA class II epitopes by deep motif deconvolution of immunopeptidomes," *Nature Biotechnology*, vol. 37, pp. 1283–1286, nov 2019. 62, 78, 81, 83

[246] J. Sidney, S. Becart, M. Zhou, K. Duffy, M. Lindvall, E. C. Moore, E. L. Moore, T. Rao, N. Rao, M. Nielsen, B. Peters, and A. Sette, "Citrullination only infrequently impacts peptide binding to HLA class II MHC," *PLoS ONE*, vol. 12, may 2017. 62

[247] B. Peters, M. Nielsen, and A. Sette, "T Cell Epitope Predictions," *Annual Review of Immunology*, vol. 38, pp. 123–145, apr 2020. 78

[248] E. Karosiene, M. Rasmussen, T. Blicher, O. Lund, S. Buus, and M. Nielsen, "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, pp. 711–724, oct 2013. 78

[249] T. J. O'Donnell, A. Rubinsteyn, M. Bonsack, A. B. Riemer, U. Laserson, and J. Hammerbacher, "MHCflurry: Open-Source Class I MHC Binding Affinity Prediction," *Cell Systems*, vol. 7, pp. 129–132.e4, jul 2018. 78, 81

[250] Y. Kim, J. Sidney, C. Pinilla, A. Sette, and B. Peters, "Derivation of an amino acid similarity matrix for peptide:MHC binding and its application as a Bayesian prior," *BMC Bioinformatics*, vol. 10, nov 2009. 78

[251] C. Garde, S. H. Ramarathinam, E. C. Jappe, M. Nielsen, J. V. Kringelum, T. Trolle, and A. W. Purcell, "Improved peptide-MHC class II interaction prediction through integration of eluted ligand and peptide affinity data," *Immunogenetics*, vol. 71, pp. 445–454, jul 2019. 78

[252] B. Alvarez, B. Reynisson, C. Barra, S. Buus, N. Ternette, T. Connelley, M. Andreatta, and M. Nielsen, "NNAlign-MA; MHC peptidome deconvolution for accurate MHC binding motif characterization and improved t-cell epitope predictions," *Molecular and Cellular Proteomics*, vol. 18, no. 12, pp. 2459–2477, 2019. 78, 79, 83, 84

[253] B. Reynisson, C. Barra, S. Kaabinejadian, W. H. Hildebrand, B. Peters, B. Peters, M. Nielsen, and M. Nielsen, "Improved Prediction of MHC II Antigen Presentation through Integration and Motif Deconvolution of Mass Spectrometry MHC Eluted Ligand Data," *Journal of Proteome Research*, vol. 19, pp. 2304–2315, jun 2020. 80, 81, 83, 84, 85, 116

[254] S. Sarkizova, S. Klaeger, P. M. Le, L. W. Li, G. Oliveira, H. Keshishian, C. R. Hartigan, W. Zhang, D. A. Braun, K. L. Ligon, P. Bachireddy, I. K. Zervantonakis, J. M. Rosenbluth, T. Ouspenskaia, T. Law, S. Justesen, J. Stevens, W. J. Lane, T. Eisenhaure, G. Lan Zhang, K. R. Clauser, N. Hacohen, S. A. Carr, C. J. Wu, and D. B. Keskin, "A large peptidome dataset improves HLA class I epitope prediction across most of the human population," *Nature Biotechnology*, vol. 38, pp. 199–209, feb 2020. 80, 83

[255] A. O'Donnell, T.J., Rubinsteyn, A., Bonsack, M., Riemer and J. Laserson, U. and Hammerbacher, "MHCFlurry," 2020. 81

[256] X. M. Shao, R. Bhattacharya, J. Huang, I. K. Sivakumar, C. Tokheim, L. Zheng, D. Hirsch, B. Kaminow, A. Omdahl, M. Bonsack, A. B. Riemer, V. E. Velculescu, V. Anagnostou, K. A. Pagel, and R. Karchin, "High-throughput prediction of MHC Class i and II neoantigens with MHCnuggets," *Cancer Immunology Research*, vol. 8, pp. 396–408, mar 2020. 81

[257] J. Liu, Z., Jin, J., Cui, Y., Xiong, Z., Nasiri, A., Zhao, Y. and Hu, "DeepSeqPanII: an interpretable recurrent neural network model with attention mechanism for peptide-HLA class II binding prediction," *bioRxiv*, 2019. 81

[258] S. Paul, E. Karosiene, S. K. Dhanda, V. Jurtz, L. Edwards, M. Nielsen, A. Sette, and B. Peters, "Determination of a predictive cleavage motif for eluted major histocompatibility complex class II ligands," *Frontiers in Immunology*, vol. 9, aug 2018. 81, 88

[259] A. F. Jarnuczak, D. C. Lee, C. Lawless, S. W. Holman, C. E. Eyers, and S. J. Hubbard, "Analysis of Intrinsic Peptide Detectability via Integrated Label-Free and SRM-Based Absolute Quantitative Proteomics," *Journal of Proteome Research*, vol. 15, pp. 2945–2959, sep 2016. 82

[260] S. W. Scally, J. Petersen, S. C. Law, N. L. Dudek, H. J. Nel, K. L. Loh, L. C. Wijeyewickrema, S. B. Eckle, J. van Heemst, R. N. Pike, J. McCluskey, R. E. Toes, N. L. La Gruta, A. W. Purcell, H. H. Reid, R. Thomas, and J. Rossjohn, "A molecular basis for the association of the HLA-DRB1 locus, citrullination, and rheumatoid arthritis," *Journal of Experimental Medicine*, vol. 210, pp. 2569–2582, nov 2013. 83

[261] M. S. Khodadoust, N. Olsson, L. E. Wagar, O. A. W. Haabeth, B. Chen, K. Swaminathan, K. Rawson, C. L. Liu, D. Steiner, P. Lund, S. Rao, L. Zhang, C. Marceau, H. Stehr, A. M. Newman, D. K. Czerwinski, V. E. H. Carlton, M. Moorhead, M. Faham, H. E. Kohrt, J. Carette, M. R. Green, M. M. Davis, R. Levy, J. E. Elias, and A. A. Alizadeh, "Antigen presentation profiling reveals recognition of lymphoma immunoglobulin neoantigens," *Nature 2017 543:7647*, vol. 543, pp. 723–727, mar 2017. 83

[262] D. Ritz, E. Sani, H. Debiec, P. Ronco, D. Neri, and T. Fugmann, "Membranal and Blood-Soluble HLA Class II Peptidome Analyses Using Data-Dependent and Independent Acquisition," *PROTEOMICS*, vol. 18, p. 1700246, jun 2018. 83

[263] M. Álvaro-Benito, E. Morrison, E. T. Abualrous, B. Kuropka, and C. Freund, "Quantification of HLA-DM-Dependent Major Histocompatibility Complex of Class II Immunopeptidomes by the Peptide Landscape Antigenic Epitope Alignment Utility," *Frontiers in Immunology*, vol. 0, p. 872, may 2018. 83

[264] P. P. Nanaware, M. M. Jurewicz, J. D. Leszyk, S. A. Shaffer, and L. J. Stern, "HLA-DO Modulates the Diversity of the MHC-II Self-peptidome *[S]," *Molecular & Cellular Proteomics*, vol. 18, pp. 490–503, mar 2019. 83

[265] NetMHCpan, "NetMHCpan 4.1 Motif Viewer," *http://www.cbs.dtu.dk/services/NetMHCpan-4.1/logos_ps.php*. 93

[266] NetMHCpan, "NetMHCIIpan 4.0 Motif Viewer," *http://www.cbs.dtu.dk/services/NetMHCIIpan/logos.php*. 93

[267] E. Fenoy, J. M. G. Izarzugaza, V. Jurtz, S. Brunak, and M. Nielsen, "A generic deep convolutional neural network framework for prediction of receptor–ligand interactions—NetPhosPan: application to kinase phosphorylation prediction," *Bioinformatics*, vol. 35, pp. 1098–1107, apr 2019. 93

[268] PyTorch, "Conv1d," *https://pytorch.org/docs/stable/generated/torch.nn.Conv1d.html*. 94

[269] Wikipedia, "Indicator function," *https://en.wikipedia.org/wiki/Indicator_function*. 95

[270] Github, "Keras: Deep Learning for humans," *https://github.com/keras-team/keras*. 95

[271] T. Sekine, A. Perez-Potti, O. Rivera-Ballesteros, K. Strålin, J. B. Gorin, A. Olsson, S. Llewellyn-Lacey, H. Kamal, G. Bogdanovic, S. Muschiol, D. J. Wullimann, T. Kammann, J. Emgård, T. Parrot, E. Folkesson, M. Akber, L. Berglin, H. Bergsten, S. Brighenti, D. Brownlie, M. Butrym, B. Chambers, P. Chen, M. C. Jeannin, J. Grip, A. C. Gomez, L. Dillner, I. D. Lozano, M. Dzidic, M. F. Tullberg, A. Färnert, H. Glans, A. Haroun-Izquierdo, E. Henriksson, L. Hertwig, S. Kalsum, E. Kokkinou, E. Kvedaraite, M. Loreti, M. Lourda, K. Maleki, K. J. Malmberg, N. Marquardt, C. Maucourant, J. Michaelsson, J. Mjösberg, K. Moll, J. Muva, J. Mårtensson, P. Nauclér, A. Norrby-Teglund, L. P. Medina, B. Persson, L. Radler, E. Ringqvist, J. T. Sandberg, E. Sohlberg, T. Soini, M. Svensson, J. Tynell, R. Varnaite, A. V. Kries, C. Unge, O. Rooyackers, L. I. Eriksson, J. I. Henter, A. Sönnerborg, T. Allander, J. Albert, M. Nielsen, J. Klingström, S. Gredmark-Russ, N. K. Björkström, J. K. Sandberg, D. A. Price, H. G. Ljunggren, S. Aleman, and M. Buggert, "Robust T Cell Immunity in Convalescent Individuals with Asymptomatic or Mild COVID-19," *Cell*, vol. 183, pp. 158–168.e14, oct 2020. 115

[272] B. Dearlove, E. Lewitus, H. Bai, Y. Li, D. B. Reeves, M. G. Joyce, P. T. Scott, M. F. Amare, S. Vasan, N. L. Michael, K. Modjarrad, and M. Rolland, "A SARS-CoV-2 vaccine candidate would likely match all currently circulating variants," *Proceedings of the National Academy of Sciences*, vol. 117, pp. 23652–23662, sep 2020. 115

[273] G. Liu, B. Carter, T. Bricken, S. Jain, M. Viard, M. Carrington, and D. K. Gifford, "Computationally Optimized SARS-CoV-2 MHC Class I and II Vaccine Formulations Predicted to Target Human Haplotype Distributions," *Cell Systems*, vol. 11, pp. 131–144.e6, aug 2020. 115

[274] D. Montes-Grajales and J. Olivero-Verbel, "Bioinformatics Prediction of SARS-CoV-2 Epitopes as Vaccine Candidates for the Colombian Population," *Vaccines 2021, Vol. 9, Page 797*, vol. 9, p. 797, jul 2021. 115

[275] M. Shkurnikov, S. Nersisyan, T. Jankevic, A. Galatenko, I. Gordeev, V. Vechorko, and A. Tonevitsky, "Association of HLA Class I Genotypes With Severity of Coronavirus Disease-19," *Frontiers in Immunology*, vol. 0, p. 423, feb 2021. 115

[276] B. Agerer, M. Koblischke, V. Gudipati, L. F. Montaño-Gutierrez, M. Smyth, A. Popa, J.-W. Genger, L. Endler, D. M. Florian, V. Mühlgrabner, M. Graninger, S. W. Aberle, A.-M. Husa, L. E. Shaw, A. Lercher, P. Gattinger, R. Torralba-Gombau, D. Trapin, T. Penz, D. Barreca, I. Fae, S. Wenda, M. Traugott, G. Walder, W. F. Pickl, V. Thiel, F. Allerberger, H. Stockinger, E. Puchhammer-Stöckl, W. Weninger, G. Fischer, W. Hoepler, E. Pawelka, A. Zoufaly, R. Valenta, C. Bock, W. Paster, R. Geyeregger, M. Farlik, F. Halbritter, J. B. Huppa, J. H. Aberle, and A. Bergthaler, "SARS-CoV-2 mutations in MHC-I-restricted epitopes evade CD8+ T cell responses," *Science Immunology*, vol. 6, mar 2021. 115

[277] C. J. Reynolds, C. Pade, J. M. Gibbons, D. K. Butler, A. D. Otter, K. Menacho, M. Fontana, A. Smit, J. E. Sackville-West, T. Cutino-Moguel, M. K. Maini, B. Chain, M. Noursadeghi, U. C. I. C. Network‡, T. Brooks, A. Semper, C. Manisty, T. A. Treibel, J. C. Moon, U. C. Investigators‡, A. M. Valdes, Á. McKnight, D. M. Altmann, and R. Boyton, "Prior SARS-CoV-2 infection rescues B and T cell responses to variants after first vaccine dose," *Science*, vol. 372, pp. 1418–1423, jun 2021. 115

[278] K. M. Campbell, G. Steiner, D. K. Wells, A. Ribas, and A. Kalbasi, "Prediction of SARS-CoV-2 epitopes across 9360 HLA class I alleles," *bioRxiv*, 2020. 115

[279] K. M. Campbell, G. Steiner, D. K. Wells, A. Ribas, and A. Kalbasi, "neoCOVID Explorer," *https://rstudio-connect.parkerici.org/content/13/*. 115

[280] A. A. Quadeer, S. F. Ahmed, and M. R. McKay, "Landscape of epitopes targeted by T cells in 852 individuals recovered from COVID-19: Meta-analysis, immunoprevalence, and web platform," *Cell Reports Medicine*, vol. 2, p. 100312, jun 2021. 115

[281] A. A. Quadeer, S. F. Ahmed, and M. R. McKay, "SARS-CoV-2 T cell epitopes," *https://www.mckayspcb.com/SARS2TcellEpitopes/*. 115

[282] S. A, K. M, R. M, H. MN, Ø. T, B. MR, T. S, G. M, H. MB, N. M, C. JP, R. T. A, and B. S, "A Systematic, Unbiased Mapping of CD8 + and CD4 + T Cell Epitopes in Yellow Fever Vaccinees," *Frontiers in immunology*, vol. 11, aug 2020. 115

[283] T. J. O'Donnell, A. Rubinsteyn, and U. Laserson, "MHCflurry 2.0: Improved Pan-Allele Prediction of MHC Class I-Presented Peptides by Incorporating Antigen Processing," *Cell Systems*, vol. 11, pp. 42–48.e7, jul 2020. 116

[284] P. Phloyphisut, N. Pornputtapong, S. Sriswasdi, and E. Chuangsuwanich, "MHCSeqNet: a deep neural network model for universal MHC binding prediction," *BMC Bioinformatics 2019 20:1*, vol. 20, pp. 1–10, may 2019. 116

[285] X. Yang, L. Zhao, F. Wei, and J. Li, "DeepNetBim: deep learning model for predicting HLA-epitope interactions based on network analysis by harnessing binding and immunogenicity information," *BMC Bioinformatics 2021 22:1*, vol. 22, pp. 1–16, may 2021. 116

[286] B. Pei and Y.-H. Hsu, "IConMHC: a deep learning convolutional neural network model to predict peptide and MHC-I binding affinity," *Immunogenetics 2020 72:5*, vol. 72, pp. 295–304, jun 2020. 116

[287] G. Venkatesh, A. Grover, G. Srinivasaraghavan, and S. Rao, "MHCAttnNet: predicting MHC-peptide bindings for MHC alleles classes I and II using an attention-based deep neural model," *Bioinformatics*, vol. 36, pp. i399–i406, jul 2020. 116

[288] T. Zhao, L. Cheng, T. Zang, and Y. Hu, "Peptide-Major Histocompatibility Complex Class I Binding Prediction Based on Deep Learning With Novel Feature," *Frontiers in Genetics*, vol. 0, p. 1191, nov 2019. 116

[289] J. Wu, W. Wang, J. Zhang, B. Zhou, W. Zhao, Z. Su, X. Gu, J. Wu, Z. Zhou, and S. Chen, "DeepHLApan: A Deep Learning Approach for Neoantigen Prediction Considering Both HLA-Peptide Binding and Immunogenicity," *Frontiers in Immunology*, vol. 0, p. 2559, nov 2019. 116

[290] J. Jin, Z. Liu, A. Nasiri, Y. Cui, S.-Y. Louis, A. Zhang, Y. Zhao, and J. Hu, "Deep learning pan-specific model for interpretable MHC-I peptide binding prediction with improved attention mechanism," *Proteins: Structure, Function, and Bioinformatics*, vol. 89, pp. 866–883, jul 2021. 116

[291] J.-W. Sidhom, D. Pardoll, and A. Baras, "AI-MHC: an allele-integrated deep learning framework for improving Class I & Class II HLA-binding predictions," *bioRxiv*, p. 318881, may 2018. 116

[292] Y. Han and D. Kim, "Deep convolutional neural networks for pan-specific peptide-MHC class I binding prediction," *BMC Bioinformatics 2017 18:1*, vol. 18, pp. 1–9, dec 2017. 116

[293] J. Cheng, K. Bendjama, K. Rittner, and B. Malone, "BERTMHC: improved MHC–peptide class II interaction prediction with transformer and multiple instance learning," *Bioinformatics*, jun 2021. 116