Improved Immunoinformatic Methods for Rationale T Cell Epitope Discovery

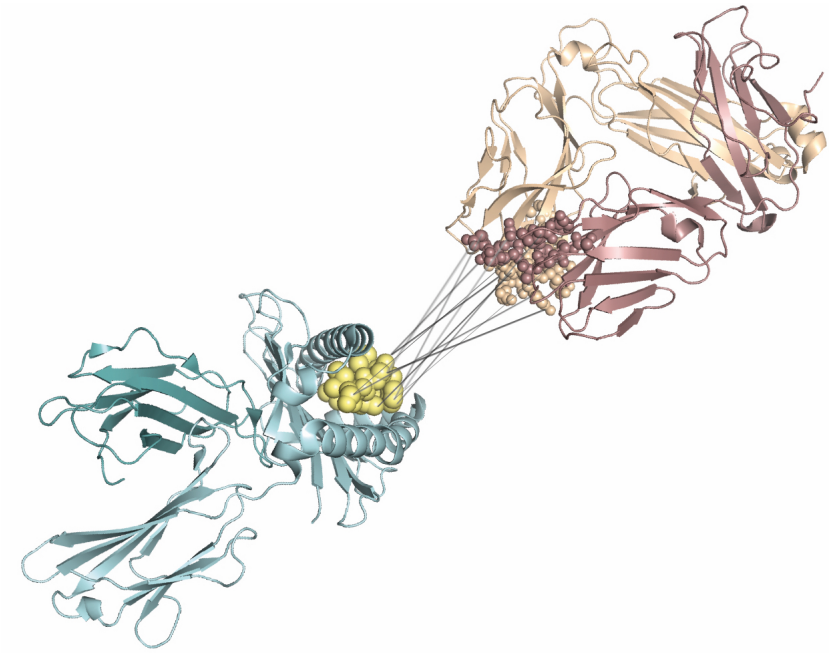
Montemurro, Alessandro

Publication date:
2022

Document Version
Publisher's PDF, also known as Version of record

DTU Health Tech
Department of Health Technology

Improved Immunoinformatic Methods for Rationale T Cell Epitope Discovery



Alessandro Montemurro

PhD Thesis
September 2022

Improved Immunoinformatic Methods for Rationale T Cell Epitope Discovery

Alessandro Montemurro

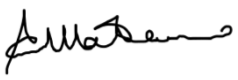
PhD Thesis

September 2022

Preface

This PhD thesis was prepared as part of the requirements to obtain a PhD degree at the Technical University of Denmark (DTU). The work presented in this thesis was carried out in the Immunoinformatics and Machine Learning group, section of Bioinformatics, at DTU Health Technology.

The here presented work was produces form October 2019 to September 2022 under the supervision of Professor Morten Nielsen and co-supervision of Associate Professor Leon Eyrich Jessen.



Alessandro Montemurro
Kongens Lyngby, September 2022

Publications Included in the Thesis

PAPER I

NetTCR-2.0 enables accurate prediction of TCR-peptide binding by using paired TCRα and β sequence data

Alessandro Montemurro, Viktoria Schuster, Helle Rus Povlsen, Amalie Kai Bentze, Vanessa Jurtz, William D. Chronister, Austin Crinklaw, Sine R. Hadrup, Ole Winther, Bjoern Peter, Leon Eyrich Jessen, Morten Nielsen

Published in: Communications Biology, Volume: 4, Issue: 1, Pages: 1-13, Year: 2021

PAPER II

NetTCR-2.1: Lessons and guidance on how to develop models for TCR specificity predictions

Alessandro Montemurro, Leon Eyrich Jessen, Morten Nielsen

Submitted to: Frontiers in Immunology, September 2022

PAPER III

Benchmark of data-driven filtering approaches for single-cell screening of T cell specificity

Helle Rus Povlsen, Alessandro Montemurro, Leon Eyrich Jessen, Morten Nielsen

Data from an on-going project

Summary

The research presented in this doctoral thesis involves the development of data-driven methods for understanding the mechanisms behind T cell recognition and predicting T cell specificity.

T cells play a crucial role in adaptive immunity as they are able to detect the presence of pathogens or malignant cell mutations. T cells engage with the other cells through the T cell receptor (TCR), and TCRs interact with the peptide-MHC complexes expressed on the cell surface. Upon detection of foreign antigens or malfunctioning self-antigens, T cells trigger a cascade of events that leads to the elimination of the malfunctioning cells. To ensure protection against the broadest variety of pathogens possible, the immune system has evolved to generate a highly diverse TCR repertoire. This diversity is achieved through a stochastic process of TCR generation. TCR repertoire diversity is what makes the immune system very powerful, but it also makes it challenging to understand the extract some common rules governing TCR-epitope recognition.

The first part of the thesis gives an overview of the theoretical aspects of the thesis's topics, followed by three research projects. The thesis is concluded with an epilogue, summarizing the main findings of the research and future perspectives.

In the first published work we proposed NetTCR-2.0, a convolutional network trained on TCR and epitope amino acid sequences. We successfully built a model able to predict binding between a TCR and a peptide presented by the MHC I molecule HLA-A*02:01. We trained the neural network using both α and β -chain CDR3 loops, showing that this method consistently outperformed the models trained on single chain inputs. Subsequently, we expanded the proposed model to include the full set of six CDR sequences as input, showing that this yields a gain in performance. Furthermore, as new data was released, NetTCR-2.1 was trained on a larger dataset covering more HLA molecules. Special attention was given to data curation during the model development. We defined a pipeline to pre-process the input data and prevent performance inflation due to data redundancy. The pipeline also

included an analysis on how to artificially generate a set of negative interactions, as these are usually not available.

The final research project reported in this dissertation presents the results from an ongoing project and proposes an application of the NetTCR method described in the previous research papers. Given the potentially large amount of data generated with single-cell RNA sequencing platforms, filtering pipelines are being developed to remove artifacts and noisy data points from the dataset. We presented two data-driven filtering approaches, ICON and ATRAP, and compared their ability to filter the data. We concluded that the two pipelines successfully filter out noisy TCR-peptide annotations, retaining only the most reliable interactions. We confirmed this by training a neural network on the raw and the filtered data, showing that the models trained on the cleaned dataset yield improved performance.

As a whole, the presented work aims to uncover the mechanisms behind TCR recognition and provides a computational framework to predict TCR-peptide interaction. Being able to predict T cell specificity will make it easier to create novel strategies for the treatment of infections, autoimmune diseases, as well as cancer.

Resumé

Forskningen som er beskrevet i denne Ph.d.-afhandling, omhandler datadrevne metoder til at forstå mekanismerne bag T celle genkendelse og til at forudsige T celle specificitet.

T celler spiller en afgørende rolle i det adaptive immunforsvar da de kan detektere tilstedeværelsen af patogene mikroorganismer eller ondartede cellemutationer. T celler interagerer med andre celler ved brug af T celle receptorer (TCR'er), og TCR'er interagerer med peptid-MHC komplekser udtrykt på celleoverfladen. Ved genkendelse af fremmede antigener eller selv-antigener med funktionsfejl, igangsætter T celler en kaskade af begivenheder der resulterer i udryddelse af celler med funktionsfejl. For at sikre beskyttelse mod så mange forskellige patogene mikroorganismer som muligt, har immunforsvaret udviklet sig til at producere et TCR-repertoire med meget høj diversitet. Denne diversitet er opnået gennem en stokastisk proces af TCR-fremstilling. TCR-repertoirets diversitet gør immunforsvaret meget kraftfuldt, men det vanskeliggør også forståelsen af reglerne bag TCR-epitop genkendelse.

Den første del af afhandlingen giver et overblik over de teoretiske aspekter af afhandlingens emner, efterfulgt af tre separate forskningsprojekter. Afhandlingen afsluttes med en epilog der opsummerer hovedresultaterne af forskningen samt fremtidige perspektiver.

I den første publikation præsenterede vi NetTCR-2.0, som er et convolutional neural network trænet på TCR og epitop aminosyresekvenser. Vi fik med succes udviklet en model der kan forudsige binding mellem en TCR og peptider præsenteret af MHC I molekylet HLA-A*02:01. Vi trænede modellen på både - og - kæde CDR3 loops, og viste at denne model konsekvent udkonkurrerede modeller trænet på data med kun en enkelt kæde. Efterfølgende udvidede vi modellen til at inkludere det fulde sæt af seks CDR-sekvenser som input, og viste at dette gav forbedret performance. Da nye data blev tilgængelige, fik vi herudover trænet NetTCR-2.1 på et udvidet datasæt der dækker flere HLA-molekyler. Særlig opmærksomhed blev rettet mod datakurater-

ing under modeludviklingen. Vi udarbejdede en pipeline som kan præprocessere input dataet og forhindre overestimering af performance pga. dataredundans. Datasætkurateringen omfattede også en analyse af hvordan man syntetisk kan generere et sæt af negative interaktioner, da disse normalt ikke er tilgængelige.

Det sidste forskningsprojekt beskrevet i denne afhandling præsenterer resultaterne fra et igangværende projekt, og omhandler en anvendelse af NetTCR metoden beskrevet i de forrige to forskningsartikler. På grund af den potentielt store mængde data, der genereres med scRNA-sekventeringsplatforme, bliver der udviklet filtreringspipelines til at fjerne artefakter og støjende datapunkter fra datasættet. Vi præsenterede to datadrevne filtreringsmetoder, ICON og ATRAP, og sammenlignede deres evne til at filtrere dataet. Vi konkluderede at de to pipelines med succes kan bortfiltrere støjfyldte TCR-peptid annoteringer, og dermed bevare kun de mest pålidelige interaktioner. Dette bekræftede vi ved at træne et neuralt netværk på henholdsvis de rå og filtrerede data, og viste at modellerne trænet på de rensede data opnåede forbedret performance.

Som helhed har den præsenterede forskning som mål at belyse mekanismerne bag TCR-genkendelse, og giver en beregningsmæssig metode til at forudsige TCR-peptid genkendelse. At kunne forudsige T celle specificitet vil gøre det nemmere at udvikle nye strategier til behandling af infektioner, autoimmune sygdomme, såvel som kræft.

Acknowledgements

Here we are, at the end of my PhD. It's been exciting, frustrating, stimulating, bumpy, and much more. But it has been an amazing experience. I would like to express my gratitude to the people that, in one way or another, helped me get here still intact (more or less).

First, I wish to thank my supervisor, Professor Morten Nielsen, whose inspiration, guidance, and feedback have been crucial during our research. Thanks for being always so direct and honest, this has helped me grow a lot, both professionally and personally. I would also like to thank my co-supervisor Associate Professor Leon Eyrich Jessen, for his insightful comments and precious advice through my PhD journey.

A big thanks goes to all the members of the Immunoinformatics and Machine Learning group, Helle, Birkir, Yat, Carol, Magnus, Jonas, Heli. Thanks for the great scientific discussions but also for the small conversations over a coffee when it was much needed. I would also like to thank the entire Bioinformatics section. Thanks for making it a fun place to work.

A special thanks goes to my *Venner i fresh linen*. Maite, Francesca, Andrea and Irene (yes, also Irene..), thanks for sharing with me every single moment of this journey, since day one in Copenhagen.

A big thanks to all my people here in Copenhagen. Thanks Mina, talking about T cells (and not only) over a beer is always more fun. Thanks Yat, for teaching me that the right approach to life is everything (LMA). Thanks Alessandra, Maurizio, Federica, for always supporting and encouraging me whenever I needed. Thanks Albert, thanks regardless of everything. Thank you all for making Copenhagen home.

But my thanks goes also to my Italian half-heart. Thanks Ilaria, Giorgia (fra un mese mi laureo!!! E ora basta, lo prometto), Erika, Cucchi, Taia. Thanks for being always on my side, even being a thousand kilometers apart.

Lastly, I would like to say thanks my family, Mamma, Papy, and Anto (and Carlo), who have always supported me and always will,

unconditionally.

Thanks to all those people who made it possible for me to be here, now.

Alessandro

Abbreviations

ANN	Artificial Neural Network
APC	Antigen-Presenting Cell
AUC	Area Under the Curve
BLOSUM	Block Substitution Matrix
CD	Cluster of Differentiation
CDR	Complementarity Determining Region
CNN	Convolutional Neural Network
CV	Cross Validation
DP	Double-positive
FN	False Negative
FNN	Feed-Forward Neural Network
FP	False Positive
FPR	False Positive Rate
HLA	Human Leukocyte Antigen
k-NN	k Nearest Neighbours
LSTM	Long Short Term Memory
MHC	Major Histocompatibility Complex
NN	Neural Network
PPV	Positive Predicted Value
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristics
SC	Single Cell
SGD	Stochastic Gradient Descent
SP	Single Positive
TCR	T Cell Receptor
TN	True Negative
TP	True Positive
TPR	True Positive Rate
VAE	Variational Autoencoder

Contents

Introduction

Our immune system is a vast and intricate set of mechanisms specialized to protect the body from the outer world. The role of the immune system cells is to circulate in the body, always screening the surrounding. Specifically, T cells use a receptor on their surface, the T cell receptor (TCR) to scan the peptide-MHC complexes expressed on the surface of the cells. When a T cell encounters a peptide fragment derived from a virus or product of a mutation, an immune response is triggered and a chain of events it activated, aiming to kill the malfunctioning cell. Thus, the TCR-peptide-MHC complex represents the hallmark of T cell-mediated immunity. As the TCRs are highly specific to a pathogen, the immune system evolved so that the TCRs are immensely variable, to ensure the broadest protection possible. This variability is what makes adaptive immunity so powerful, but at the same time, it makes it challenging to study its principles. Recently, the volume of the TCR generated is steadily increasing and bioinformatics techniques are needed truly benefit from this data.

The aim of this thesis was to advance the current understanding of peptide-MHC recognition by T cell receptors and build machine learning models to predict their interaction. The ability to predict this interaction would make it easier to track the development of infectious diseases and open the door to immunotherapies for cancer or T cell-based vaccine design.

The thesis is structured in the following way:

Chapter 1 gives an introduction to the apparatus of the T cell-mediated immunity. Key concepts, such as T cell development and

maturation, antigen presentation by the major histocompatibility complex and T cell activation are discussed in the chapter.

Chapter 2 focuses on deep learning modeling approaches. Different neural network architectures are defined, and their applications to different type of data are discussed. The chapter also gives a description of the model training process, including techniques to avoid overfitting, and performance evaluation metrics.

Chapter 3 is the bridge between the first two chapters. The first part of the chapter focuses on how to deal with data redundancy and how to properly build a data set for deep learning models. The second part of the chapter describes possible modeling approaches to predict TCR specificity and gives and overview of the currently available tools to solve this task.

Chapter 4 presents the first scientific publication about NetTCR-2.0. The main aim of the project was to build a model capable of predicting TCR-epitope interaction, showing that both α and β chains of the TCR are needed as input to the model.

Chapter 5 is based on the second scientific paper, and presents NetTCR-2.1, an update version of the model presented in Chapter 4. The aim of the project as to address common challenges involved in the construction of a TCR-peptide binding predictor, i.e. data redundancy reduction, generation of negative data or inputs selection.

Chapter 6 introduces a third project, which is on-going. The contribution of the project was to show that data-driven filtering approaches such as ICON or ATRAP successfully increase the signal-to-noise ratio in single-cell sequencing data, removing noisy TCR-peptide pairs from the dataset.

Chapter 7 concludes the thesis with an epilogue, discussing the key points from the presented work as well as the future perspective.

CHAPTER 1

T Cell Mediated Immunity

The immune system is a complex network of cells, proteins and organs that protect the body from external threats. It consists of various mechanisms dedicated to recognizing and fighting pathogens. The immune system has two main components: innate and adaptive immunity [1]. The innate immune system serves as the first line of defense against pathogens that enter the body through the skin or other external barriers. It comprises various nonspecific mechanisms, such as fever or inflammation and the immune response is mounted very quickly: for instance, if bacteria enter the body through a wound, the innate immune system will make sure to clear the infection within a few hours.

When innate immunity fails to defeat the infection, the adaptive immune system is activated. Unlike the innate immunity mechanisms, which react toward common broad categories of pathogens, adaptive immunity is highly specific to a particular pathogen [2]. The two main cell components of the adaptive immune system are the T and B lymphocytes (or simply T and B cells). These cells move in the body, always surveying the surrounding for pathogens. T and B cells detect the presence of foreign substances using a receptor on their surface, the T Cell Receptor (TCR) and B Cell Receptor (BCR), respectively. The high adaptability of this type of immunity is due to the fact that T cells and B cells undergo

a process of somatic rearrangement of the DNA, producing an immense variety of clones with different receptors that each are potentially specific to a particular pathogen [3]. Adaptive immunity also creates an immunological memory: after clearing the infection, a fraction of the effector lymphocytes will develop into memory cells and will be ready for eventual future re-infections [4, 5]. This explains why the adaptive immune system might take from several days to weeks to mount a response upon the first encounter with the pathogen; however, reinfection at later time points leads to a rapid response, thanks to the memory cells that will be quickly recruited to fight the known pathogen.

T cells are responsible for the so-called cell-mediated immunity. They are produced in the bone marrow and migrate to the thymus, where they will maturate and go to the periphery. Other than the TCR defining its specificity, mature T cells also express a co-receptor, called cluster of differentiation (CD) [2]. T cells express either CD4 or CD8 co-receptor; $CD8^+$ T cells will recognize epitopes presented by the Major Histocompatibility Complex I (MHC I), whereas $CD4^+$ T cells will be specific to MHC II-presented peptides. For an introduction to MHC, refer to Section 1.2. The rest of this chapter will focus on T cell-mediated immunity.

1.1 T Cell Development

The T cell life cycle begins in the bone marrow as a double-negative (DN) hematopoietic progenitor cell, lacking the CD4/CD8 co-receptor and T Cell Receptor (TCR). The progenitor cells migrate from the bone marrow to the thymus, where they will undergo a process of maturation and selection. The first step involves the formation of a TCR, necessary for a mature T cell to recognize peptides presented by the MHC molecules. Secondly, two mechanisms, namely positive and negative selection, will ensure binding of the TCRs to the MHC molecule and non-binding to self-peptides.

The T Cell Receptor (TCR) is a hetero-dimeric protein, typically formed by an α and β chain (and less often by γ and δ). The TCR

comes in contact with the peptide-MHC complex (pMHC) via a set of highly variable regions called complementarity determining regions (CDRs), as they determine the specificity of a TCR. To detect a vast number of peptide epitopes, the immune system must generate T cells with a high degree of TCR diversity. This diversity is achieved by a process called V(D)J recombination (Figure 1.1).

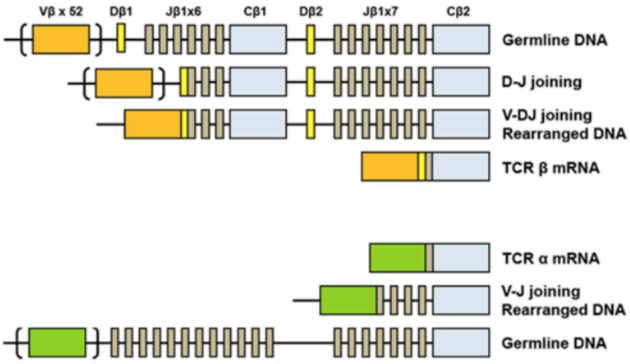


Figure 1.1: Schematic representation of the V(D)J rearrangement. Figure adapted from De Simone et al. [6]

The DNA rearrangement starts in the thymus; first, Variable (V), Joining (J), Diverse (D) and Constant (C) genes encoding the β chain are selected. The human genome contains 52 V, 2 D, 13 J and 2 C genes for the β chain (and 70 V, 61J and 1C for the α chain) [1, 7]. The first process that occurs is the D-J recombination of the selected genes, followed by the V-DJβ rearrangement. Every time this process takes place, different genes are selected, giving rise to roughly $5.8{\cdot}10^6$ possible combinations [1]. When the different gene segments are merged, the enzymes responsible for the joining randomly add or subtract some nucleotides at the junctions, leading to a theoretical diversity of $10^{15} - 10^{20}$ unique αβ TCRs [8]. Specifically, these added or deleted nucleotides at the junctions are responsible for the increased diversity in the CDR3 region of the TCR. Once the β chain is rearranged, the thymocytes are equipped with both CD4 and CD8 co-receptor, becoming double-positive (DP). At this point, the recombination of the α locus takes place, following the same rules as the β chain, but

with the only difference that no D gene is involved in the α chain rearrangement.

The double-positive lymphocytes, decorated with the newly formed TCR, undergo a process called positive selection. DP T cells are presented with self-peptides on thymic antigen-presenting cells. Based on the affinity shown towards MHC I or MHC II molecules the T cells differentiate into $CD8^+$ or $CD4^+$ single-positive (SP) T cells, respectively. Subsequently, negative selection in the thymic medulla ensures that the SP T cells do not bind too strongly to self-peptide, promoting self-tolerance. The T cells that do not successfully pass positive or negative selection are removed through an apoptosis mechanism. Only around 5% of the initial double-positive T lymphocytes will become mature, naive T cells will be part of the TCR repertoire and will migrate to the peripheral lymphoid tissues [1].

1.2 Epitopes and MHC Presentation

As described in Section 1.1, immature T cells are primed for pMHC recognition in the thymus to ensure that they will interact with MHC molecules. Furthermore, during their life cycle, the T cells scan other cells to detect infected or malfunctioning cells. The interaction between the T lymphocytes and the other cells involves the TCR on the T cell surface and the peptide-MHC complex, presented on the surface of the cells. Thus, it is clear that the MHC presentation mechanism covers a dominant role in cell-mediated immunity.

MHC is polygenic, denoting the fact that an individual has multiple different MHC genes, and polymorphic, meaning that there exist many different variations of the genes within a population. Furthermore, because the peptide fragments presented by the MHC molecules are derived from disrupted larger proteins, also peptides that are not on the surface of the proteins can be presented by the MHC on the surface of the cells. These factors make it difficult for a pathogen to escape the MHC presentation mechanism.

There are two main classes of MHC molecules, namely MHC class I and MHC class II. MHC I molecules are expressed on the surface

of all the nucleated cells. The peptides presented by MHC I are the cleavage product of intracellular, degraded proteins. They can either be derived from the human proteome, hence be self-peptides, or come from non-self proteins, produced under infection or mutation. Peptide-MHC I complexes are recognized by $CD8^+$ T cells, also called cytotoxic T cells. $CD8^+$ T cells become activated after they encounter an antigen specific to their TCR. Figure 1.2a shows the MHC I presentation pathway. Upon recognition, $CD8^+$ T cells differentiate into memory and effector cells. Effector cytotoxic T cells will bind the peptide-MHC complex and kill the infected or malfunctioning cell. As described earlier, the negative selection process during T cell development ensures that, under normal conditions, the effector T cells will not be reactive towards normally functioning cells of the body that express self-peptides.

MHC class II molecules are present on the surface of specialized antigen-presenting cells (APC), such as dendritic cells, B cells or macrophages. The antigens bound to the MHC II molecules are not cytosolic, as in MHC I, but derived from extracellular proteins. The pathogen enters the cell by phagocytosis, in the case of macrophages and dendritic cells, or endocytosis, in the case of B cells; the exogenous proteins are processed by endolysosomal enzymes and the resulting peptide fragments are ready to be loaded on an MHC II molecule. The complex will migrate to the cell surface, where it will be presented to CD4+ T cells, also called helper T cells. The MHC II presentation pathway is visualized in Figure 1.2b. Once a $CD4^+$ T cell becomes activated, it proliferates and differentiates into memory or effector T cells. Effector helper T cells trigger an immune response by releasing cytokines that attract other immune cells. Moreover, a helper T cell is usually needed to activate B cells: when the B cell finds its target antigen, it requires a $CD4^+$ T cell to start the clonal expansion.

Human MHCs are called human leukocyte antigens (HLA) and are encoded on chromosome 6. There are 3 main human molecules for MHC I: HLA-A, HLA-B, and HLA-C. Each individual has two copies of the same genes, one inherited from the mother and one copy the father; hence, up to six different HLAs are expressed. For MHC II, humans have HLA-DP, HLA-DQ, and HLA-

DR molecules. Differently from MHC I, it is possible to express more than six class II MHCs: MHC II molecules are formed by two polypeptide chains, hence combinations between different genes produce a higher number of possibilities.

The structure of the two classes of MHC molecules is shown in Figure 1.3a-b. MHC I molecules are formed by a polymorphic α chain and a non-polymorphic β_2-microglobulin. MHC II also consists of two polypeptide chains, α and β , that are both polymorphic (except HLA-DR, which has a monomorphic α chain). The α and β chains can pair in different combinations, producing different MHC II molecules.

In MHC I, the α_1 and α_2 domains form the binding cleft, whereas in MHC II the peptide binds within the α_1 and β_1 subunits. For both molecules, the binding cleft can accommodate peptides of length 9, or close. The major difference between MHC I and MHC II binding sites is that the MHC I binding grove is closed at the ends, while in MHC II the ends are open. Therefore, MHC I peptides are restricted to have a length of around 9; nonetheless, 8-mers can be also stretched and 10 and 11-mers can be squeezed to fit the binding cleft. On the contrary, MHC II peptides have no limits on sequence length, often being at least 15 amino acids long [10, 11]. Also here, the binding core is 9 amino acids long, but the open ends of the cleft make it possible to have flanking regions on the sides of the binding pocket. The binding clefts of the MHC molecules are shown in Figure 1.3c-d.

1.3 T Cell Activation

T cells interact with the pMHC complex expressed on the cell surface utilizing their T cell receptor (TCR). One mature T cell has more than 10^5 copies of the same TCR on its surface [1]. The TCR is a heterodimeric protein with two polypeptide chains, connected by a disulfide bond. In most cases, the TCRs have an α and a β chain. A subpopulation of around 5% T cells expresses a $\gamma\delta$ TCR [12, 13]. Since these TCRs follow different mechanisms and are not fully characterized, they will not be further considered in this thesis.

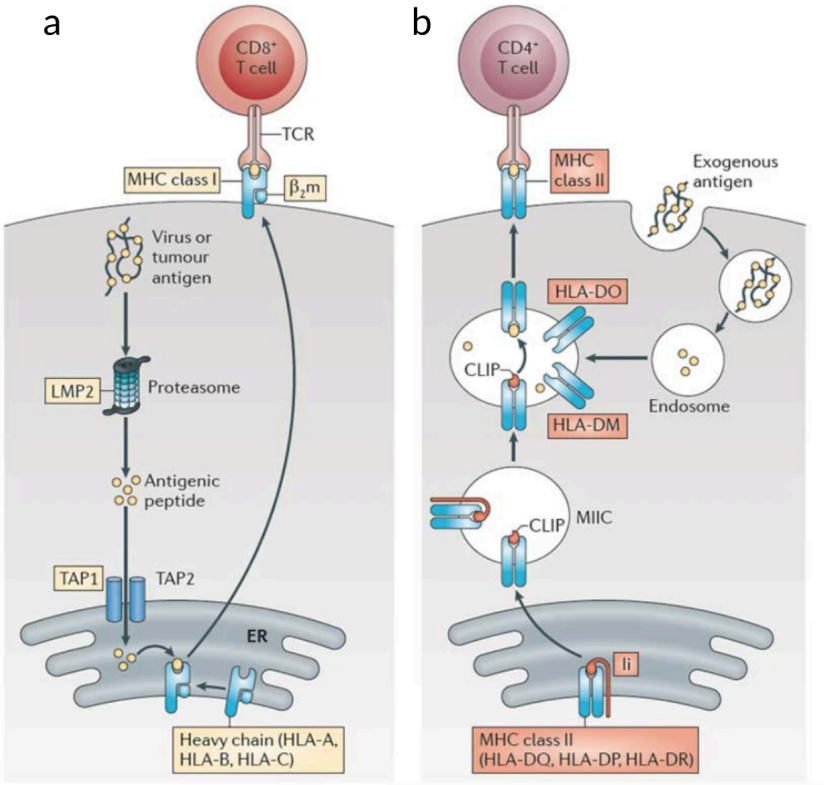


Figure 1.2: MHC class I and MHC class II presentation pathways. (a) The immunoproteasome cleaves intracellular proteins into peptide fragments. The Transporter associated with antigen processing (TAP) protein translocates the peptides over the membrane of the endoplasmic reticulum (ER). Here, the peptides are loaded into the binding groove of the MHC I molecule. Peptide-MHC I complex migrates to the cell surface and the antigen is presented to $CD8^+$ T cells. (b) Extracellular proteins are processed into peptides by endolysosomal enzymes. The peptides bind to the MHC II pocket by displacing a class II-associated invariant chain (CLIP), which comes from the class II-associated invariant chain (Ii). The antigen loading process is regulated by another MHC-like molecule, the HLA-DM in humans. Upon migration to the cell surface, MHC II will present the peptides to $CD4^+$ T cells. Figure adapted from Kobayashi et al. [9]

The structure of a TCR is shown in Figure1.4a. Each TCR chain consists of a constant and a variable region. The constant region is proximal to the cell membrane, while the variable region is re-

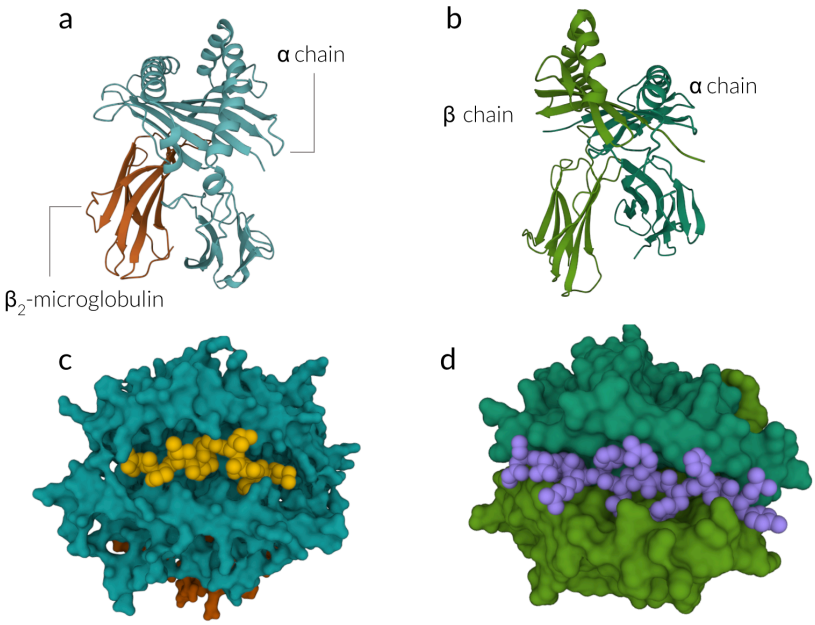


Figure 1.3: Structure of MHC molecules. (a) and (c) show MHC I molecule HLA-A2, PDB structure 6G3J. (b) and (d) show MHC II molecule HLA-DR1, PDB structure 1AQD. (a)-(c) shows the structure of the MHC molecules with the peptide binding site in the upper part of the image. (c)-(d) shows the MHC molecules' surface, viewed from above, including the bound peptides, colored in yellow and purple, respectively.

sponsible for the interaction with the pMHCs. Specifically, the variable region contains the complementarity determining regions (CDR). There are three CDR loops for each chain of the TCRs. The CDR1 and CDR2 are germline-encoded by the V gene, while the CDR3 loops are encoded by the flanking regions of the V/J gene segments; only for the β chain, the CDR3 is encoded also by a D gene segment.

The available crystal structures have revealed that the TCR positions itself diagonally above the peptide-MHC complex. This characteristic seems to be conserved across several X-ray crystallographic studies [14]. The CDR1 and 2 are closer to the MHC molecule, while the CDR3 loops are in closer contact with the peptide [15, 16]. This is also visualized in Figure 1.4b. Hence, the

CDR3 regions from both chains account for most of the specificity of the TCR towards a specific peptide. Based on the available resolved structures of the TCR-peptide-MHC complexes, and the limited diversity of TCRα , the general consensus has been, for a long time, that it is mostly the CDR3β to drive the interaction with the pMHC, while the CDR3α would participate without playing a determinant role. Lately, the research community is re-evaluating this assumption. Especially with computational methods, it has been shown that both α and β chains are needed to achieve a better performance in modeling peptide-TCR interactions; in some cases, the results suggest that the CDR3α chain is even more informative than the β CDR3 loop [17, 18].

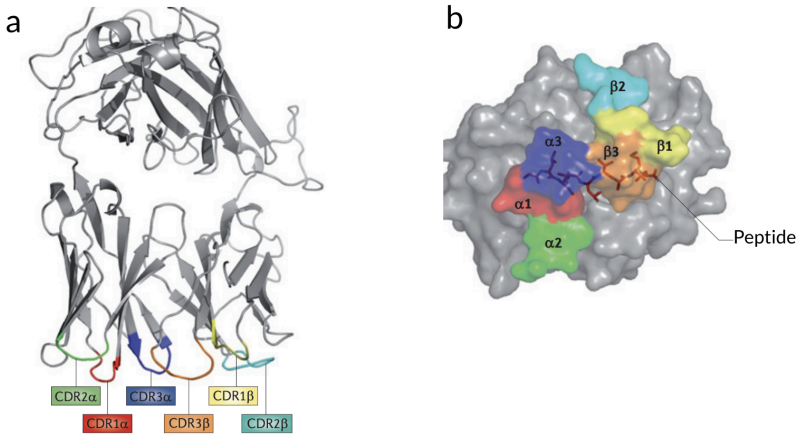


Figure 1.4: Structure of a TCR. (a) Ribbon representation of an αβTCR. The colored regions are the CDR loops. (b) Projection of the six CDR loops onto a pMHC (peptide represented by the red stick model, MHC molecule colored in gray). Specifically, the peptide is GLCTLVAML, presented by the molecule HLA-A*02:01. This projection shows that, for this pMHC, the TCR interacts with the peptide through the CDR3 loops, while CDR1 and 2 are in contact with the MHC molecule. Figure adapted from Sewell et al. [19]

Given the scarcity of crystal structures for TCR-pMHC complexes, this interaction still has to be fully characterized. One first challenge is represented by the TCR-pMHC binding mode and the plasticity of the CDR loops. Even though the diagonal orienta-

tion of the TCR over the MHC seems conserved, some studies suggest that the binding mode of a TCR changes when interacting with different peptides presented by the same MHC molecule [20]. Furthermore, the TCR-pMHC interaction is dynamic over time and the conformation of the CDR loops seems to adapt to the pMHC complex [21, 22].

Another aspect of the TCR recognition that is still unknown is to what extent TCRs are cross-reactive, i.e. be specific to more peptides. Some studies suggest that a certain level of cross-reactivity is necessary for the TCR to be able to recognize a border variety of pathogens [19].

1.4 TCR Databases

Several databases have been curated with the aim to collect TCR specificity data from different sources and aid the research community to push the field forward. Among these publicly available data sources, there are Immune Epitope Database (IEDB) [23], VDJdb [24], McPAS-TCR [25] or TBAdb [26]. These databases consist of TCR α and/or β sequences, V/J genes together with the target peptide and the relative MHC molecule. These curated datasets represent a precious resource as they allow the development of data-driven approaches to investigate TCR-pMHC interactions. However, some limitations arise due to the quantity and quality of TCR data. Most of the epitopes are characterized by one or a few TCRs, while only a few have a considerable amount of reported binding TCRs. Furthermore, the available data is biased toward few antigens. The most commonly described epitopes in the datasets are derived from common viruses such as human cytomegalovirus (CMV), influenza virus or Epstein-Barr virus (EBV). Since these infections are common in humans, these viruses are extensively studied, resulting in a large portion of available TCRs being specific to these epitopes. Table 1.1 shows the count of unique TCR sequences along with the target epitope in the IEDB, for peptides with at least 100 TCRs associated. Only 17 epitopes have a substantial amount of reported binding TCRs. Moreover, the coverage in terms of HLA molecules is also low. Among the most frequent peptides, only 6 HLA molecules are

characterized and the majority of data refers to HLA-A*02:01, which is one of the most common HLA allele in humans [27]. This bias in the data and the limited amount of sequences make the investigation of TCR-pMHC interactions challenging, as the current data is representative of only a small subset of the immense space of TCRs.

Peptide	MHC Allele	Organism	# CDR3β	# CDR3αβ
YVLDHLIVV	HLA-A*02:01	Human herpesvirus 4 (EBV)	8488	115
GLCTLVAML	HLA-A*02:01	Human herpesvirus 4 (EBV)	7032	128
NLVPMVATV	HLA-A*02:01	Human herpesvirus 5 (CMV)	4886	210
GILGFVFTL	HLA-A*02:01	Influenza A virus (CEF)	4539	438
TPRVTGGGAM	HLA-B*07:02	Human herpesvirus 5 (CMV)	2292	1
LLWNGPMAV	HLA-A*02:01	Yellow fever virus	2173	410
LPRRSGAAGA	HLA-B*07:02	Influenza A virus (CEF)	2142	-
LVVDFSQFSR	HLA-A*11:01	Hepatitis B virus	1875	-
STLPETAVVRR	HLA-A*11:01	Hepatitis B virus	925	-
ELAGIGILTV	HLA-A*02:01	Homo sapiens	558	79
KTAYSHLSTSK	HLA-A*11:01	Hepatitis B virus	476	
VTEHDTLLY	HLA-A*01:01	Human herpesvirus 5 (CMV)	274	1
EAAGIGILTV	HLA-A*02:01	Homo sapiens	214	16
RAKFKQLL	HLA-B*08:01	Human herpesvirus 4 (EBV)	187	1
ATDALMTGY	HLA-A*01:01	Hepatitis C virus	131	-
NEGVKAAW	HLA-B*44:03	Human herpesvirus 5 (CMV)	117	-
CINGVCWTV	HLA-A*02:01	Hepatitis C virus	114	28

Table 1.1: Counts of unique CDR3β and CDR3αβ for each peptide in the IEDB [23]. The table describes the epitopes with at least 100 CDR3β sequences associated. For each epitope, the MHC allele and the origin organism are reported, along with the counts of unique CDR3β and CDR3αβ in the dataset.

The TCR datasets described above collect data from previous studies and publications. Typically, TCR data is generated by multimer sorting or re-exposure assays, followed by bulk sequencing. The vast majority of these datasets contain information about CDR3β only. The reason for this is that the α and the β chain genes are not located on the same chromosome, therefore it is not possible to sequence them together. Since the CDR3β loop was thought to be the driver of the TCR-peptide interaction, given its increased diversity compared to the α chain, researchers focused their investigation on the β loop over the α.

With the advent of single-cell sequencing technologies, specifically single-cell RNA sequencing (scRNA-seq), it is now possible to have a more clear picture of the TCR-pMHC interaction. scRNA-seq

enables high-throughput screening of thousands of T cells against large libraries of pMHC complexes, linking the T cell specificity directly to the amino acid sequences of paired TCR α and β sequences.

In 2019, the commercial scRNA sequencing platform 10x Genomics released the first large single-cell database of TCR-peptide interactions [28]. This dataset contains T cell specificities from four healthy donors screened against a panel of 50 pMHCs. The assay resulted in 55,221 unique pairs of CDR3 α and β chains, together with information about their specificity. However, this data presents new challenges and single-cell platforms are generally associated with a poor signal-to-noise ratio, making specificity data more prone to artifacts and mis-annotations. Different approaches have been proposed to denoise single-cell data to truly benefit from scRNA-seq [17, 29]. A benchmark of these approaches on the 10x dataset will be the focus of Chapter 6 of this thesis.

CHAPTER 2

Deep Learning and Neural Networks

Deep Learning refers to a subset of the broader family of machine learning methods and comprises neural network-based algorithms. Neural Networks (NN), or Artificial Neural Networks (ANNs) are inspired by the human brain, simulating the way a signal is passed from one neuron to another [30, 31].

The power of ANNs is that they introduce high non-linearity in the model, allowing a more flexible and accurate mapping of the input onto the output. A central result in deep learning theory is the universal approximation theorem [32]. The main claim of the theorem is that a neural network, even a single-layer network, is able to approximate any function, provided that the number of neurons is high enough. Hence, given an adequate number of neurons and an appropriate level of non-linearity, there exists a set of weights that can approximate any function, even non-analytical ones.

Formally, an ANN is an approximation of the function f, that maps an input x to the output y: $y = f(x; \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is a set of parameters. The parameters $\boldsymbol{\theta}$ are learned from the data, to ensure the best possible approximation of the function f [30]. The building block of a neural network is called an artificial neuron. A neuron takes a data point as input and produces a real-valued output by multiplying the input with a set of weights. The non-

linearity in a neural network is achieved by applying a nonlinear function to the linear combination of inputs and weights. Artificial neurons can be combined to form layers, and multiple layers can be stacked, defining the depth of the network.

At the beginning, all the weights of the network are randomly initialized. During a process called model training, the network is shown the observational data and tries to adjust the weights in a way that the error between the original and the predicted outputs is minimized. At the end of the training, the learned vector of weights defines the approximation of f. To assess the predictive performance of a trained model, the weights are then applied to a test set, to understand how the model is able to generalize to a novel data set.

Different types of neural network architectures have been developed, to adapt the algorithms to a broader variety of input data and tasks. The simplest form of network architecture is the feed-forward neural network (FNN). In this type of network, the information moves in only one direction (forward) from the input nodes, through the hidden neurons (if any) and to the output. FNNs form the basis of every other neural network architecture. More specialized networks, e.g. Convolutional neural networks (CNN) [33] or Recurrent Neural Networks (RNN) [34], can be used to extract relevant features from the inputs; these features will be subsequently fed into one or more fully-connected layers to be combined into the output of the network. CNNs were first developed for image analysis, but then they have been successfully applied to different types of data, such as sequences or time series. Rather than looking at each single neuron, CNNs focus on sub-portions of the inputs, scanning it and trying to extract some context-dependent features. When working with sequential data, RNNs are best suited for this task. In this network architecture, the output is fed back into the network as input, forming a cycle. RNNs better model the sequentiality in the data, as each input at a specific time step t is connected to the input at the previous and next t. A popular choice of recurrent architecture is the Long Short Term Memory (LSTM) network [35]. In LSTMs, the information passes across the sequence through a variable called

cell states. In this way, LSTMs can selectively remember or forget information at each time step, allowing the flow of only the salient information in the sequence.

2.1 Feed-Forward Neural Networks

Generally, a FNN is a directed acyclic graph consisting of an input layer, one or more hidden layers and an output. The term feed-forward refers to the fact that the information flows from the input to the hidden layers, to the output in only one direction. Thus, there are no cycles where the output is fed into the network again [30]. Moreover, FNNs are fully-connected, meaning that each neuron in a specific layer is connected by an edge to all the neurons on the next layer, with an associated weight. The core component of a FNN is the artificial neuron, also called perceptron [31]. The mathematical formulation of a neuron is given by the equation

$$y = g\left(\sum_{i=1}^{n} w_i\, x_i + b\right),$$

where y is the output to predict, x is the input of size n, w_i are the weights, b is the bias term and g is a nonlinear function [36]. A FNN is a combination of multiple neurons to form a layer, and a stack of multiple layers.

Figure 2.1a shows a schematic representation of a FNN with one hidden layer. The input data access the network through the input layer, where each input neuron is connected to every hidden neuron through an edge with an associated weight. In addition to the layers' neurons, there is a special unit called bias. Adding the bias to a neuron in the next layer introduces an intercept term, allowing an adjustment of a neuron's activation [30]. The content of a hidden neuron is calculated as a weighted linear combination of the input neurons (and bias), transformed via a nonlinear function (Figure 2.1b). The function g is called activation function and it plays an important role in neural networks. In fact, it is the activation function that allows the mapping f from the input to the output to be highly nonlinear [30]. Different choices of activation functions exist, depending on the nature of the problem.

One of the most frequently used functions in classification is the sigmoid activation function, $g(z) = \frac{1}{1+e^{-z}}$. This function maps a real-valued number into a number in the interval (0, 1). When working with binary classification, it is particularly important to apply the sigmoid function on the output neuron, as the output of the network is squeezed into the interval (0, 1) and can be considered a class probability. Other common choices of activation functions are the hyperbolic tangent (tanh) or the rectified linear unit (ReLU).

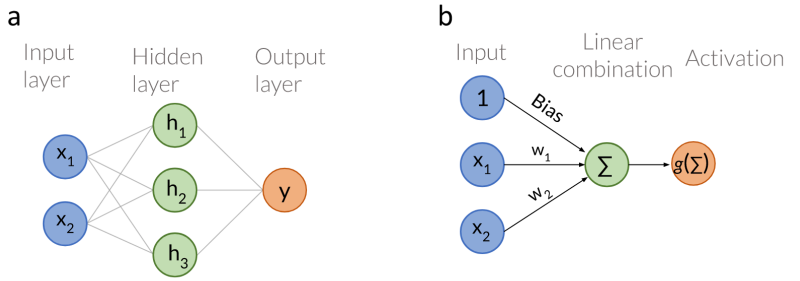


Figure 2.1: (a) Schematic example of a feed-forward neural network with two input neurons x_i, one hidden layer with three neurons h_j and one output neuron y. All the input neurons are connected to the hidden neurons, and all the hidden units are connected to the output. (b) Visualization of the mathematical operation within each neuron: the inputs x_i and the bias are summed with weights w_i; this linear combination is passed through a non-linear activation function g to give the output of the neuron.

Fully-connected networks have been successfully applied to a number of different fields of bioinformatics, including protein research [37–39]. However, some limitations arise in relation to FNNs applied to protein prediction tasks. The first limit of FNN is that the input should have a fixed length. This represents an important limitation, as proteins can go from a few amino acids to thousands [40]. In this case, the sequences should be truncated or expanded/padded to a common length, leading to loss of information. Another factor that makes FNNs not the best applicable model to sequences is that the spatial correlation and the sequentiality in the input are lost. The input neurons of a feed-forward neural network are independent from each other, and there is no such

concept as neighboring neurons. Ideally, a modeling approach should be able to look at sub-sequences at once, to capture motifs and extract some rules about the order of the amino acids in the sequences. Another class of network architectures, named convolutional neural networks, are specially designed to deal with the above-mentioned limitations. These will be discussed in the next Section.

2.2 Convolutional Neural Networks

Convolutional Neural Networks (CNN) were first described by Lecun and Bengio [33] and were developed for image and text analysis. Inspired by the visual cortex cells, convolutional neural networks introduce the concept of receptive field, i.e. a sub-portion of the visual field that solicits these cells. Following this idea, CNNs split the input into sub-regions and apply an operation, namely convolution, to the entire area at once. CNNs can be applied to inputs of any dimension: in the case of image analysis, the CNN used are 2D, as an image is represented by a 2D matrix (3D for color images). When the input is sequential, such as time series, text or protein sequences, the CNNs are 1D, as they are used to process a one-dimensional signal.

Among other applications, CNNs have been successfully applied in the field of protein research [41–43]. A protein sequence is a 1D input that can be processed by 1-dimensional CNN. As mentioned in Section 2.1, convolutional neural networks are specifically designed to address some of the limitations introduced by FNN when working with protein sequences. One of the most beneficial properties of CNNs is that they can handle inputs with different lengths: even if the input sequences have different sizes, these are scanned by multiple filters and mapped to a vector of features of equal sizes. This aspect of CNNs differentiates them from FNNs, and makes them very suitable to be applied to proteins, where the sequence length is highly variable.

One of the main differences between CNNs and FNNs is the sparse connectivity of CNNs. In FNNs, each hidden neuron is connected to all the input neurons. In CNNs, a hidden unit is connected

only to a subset of the input units; this subset is referred to as the receptive field, or convolutional filter and its width is called filter size.

Figure 2.2 depicts a 1D convolutional filter and illustrates how a CNN layer works. The filter is slid across the input, scanning the sequences. At each step, the convolution operation between the filter and the input generates an output neuron and the filter is moved by one or more positions. This process is repeated until the entire input has been covered by the filter. This defines the output of a convolutional layer.

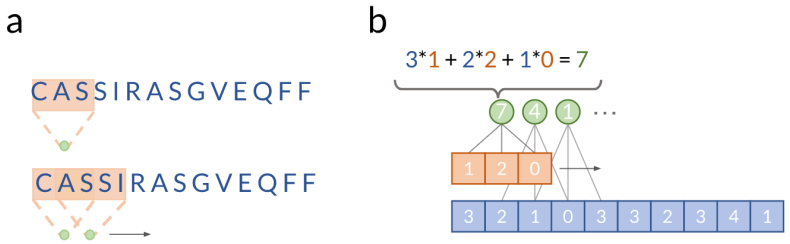


Figure 2.2: (a) A filter of size 3 is overlapped to the first three elements in the sequence and a convolution operation is applied to get a convolutional output. The filter is then slid across the sequence one position to the right and convolution is applied. This process terminates when the filter has scanned the entire sequence. (b) Numerical example of the convolutional operation: element-wise multiplication and sum of the input and the filter weights.

Formally, a convolution on k consecutive positions of an input sequence X starting at position i can be defined as

$$h_t = g(\mathrm{W}_f \cdot \mathrm{X}_{i:i+k-1} + b_f),$$

where g is an activation function, b_f is the bias term and W_f is a convolutional filter of size k [44]. The filter W_f replaces the common weights of an FNN and represents a set of learnable parameters of the network. At the beginning of the training, the filter weights are randomly initialized. During the backpropagation, the weights are updated following the same principle described earlier, i.e. in such a way that the loss is minimized.

Instead of applying many sequential convolutional layers, it is common practice to apply parallel convolutions on the input with multiple filters and different filter sizes. Having filters with multiple sizes scanning the input means that the network is analyzing the input using different resolutions. This procedure results in having many context-dependent representations of each position in the input, where each representation was influenced by a different number of neighboring neurons of the considered input. In this light, a CNN can be seen as a feature extractor. The output of each convolutional filter is a feature extracted by the network. The learned features might be abstract and hard to interpret, but they represent a set of optimal attributes, as the filter weights were learned during training while minimizing the error function.

As described up to here, a convolutional layer consists of two steps: first, many convolutions are run in parallel on the input; then, a non-linear activation function is applied to the output. Typically, these two steps are followed by a third one, that is, a pooling operation is subsequently applied. Pooling condenses the outputs on the convolutions and reduces its output dimension. Global max pooling refers to the strategy of taking the maximum activation of a given filter over the entire input. Hence, global max pooling summarizes the filters' learned features by reducing each filter to a single neuron. This vector of features can be then fed into another CNN layer or in a fully-connected layer, where these features are combined and an output is produced. As mentioned before, CNNs are not limited to sequences with the same length. It is indeed thanks to the global pooling layer that the sequence length dimension of the input is removed: an equal number of filters scans the inputs (eventually with different length) to produce a convolutional output; the outputs of the filters undergo global pooling and are reduced to a scalar value. Because each input is processed by the same number of filters, the global pooling output will have the same dimension for each input, independently of the sequence length.

2.3 Model Training and Backpropagation

At the beginning of the training process, all the weights of the network are randomly initialized. The aim of the learning algorithm is to iteratively update these parameters so that the error (or loss) between the prediction and the actual target is minimized. Depending on the prediction task, different loss functions can be used to quantify the error. When the target variable is continuous, i.e. in the case of a regression, the mean squared error loss is one of the most frequently used, $E = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$, where y is the target and $\hat{y}$ is the prediction. In classification, it is common practice to use the cross-entropy loss function, $E = -\sum_{i=1}^{n}(y_i \log(\hat{y}_i))$.

Each training iteration, also named an epoch, consists of two phases. In the first phase, called forward pass, the input data x is fed into the network, passes through the hidden layers of the network and an output $\hat{y}$ is obtained. The prediction $\hat{y}$ is compared to the actual target y and an error between the prediction and the target is computed, using a suitable loss function. The next step is called backward pass, or backpropagation. Here, the loss function is minimized with respect to the model weights until the convergence to the minimum is reached. This optimization is achieved using the gradient descent algorithm. The intuition behind this method is that the loss function is minimized by iteratively taking steps in the opposite direction of the gradient of the function in a point, as the direction given by the gradient is the steepest. Formally, the updating rule with the gradient descent at a time step t is given by the following equation:

$$w^{t+1} = w^t - \eta \frac{\partial E}{\partial w^t}.$$

The parameter η is called learning rate and represents the size of the step to take in the steepest direction. The learning rate is a hyper-parameter of the model and its tuning is crucial to ensure the convergence of the algorithm. If the chosen η is too small, the convergence will be slow and the algorithm might be stuck in a local minimum of the function; on the contrary, too big values of η will lead to divergence, as the algorithm will keep jumping over the local minimum, without reaching it. Another way of reducing the chances of being stuck in local minima is using stochastic

(or online) gradient descent (SGD). In regular gradient descent, the weights are updated only once, after the entire dataset has been shown to the network; because the gradient is accumulated over many data points, it could become very large and lead to divergence of the algorithm. In stochastic gradient descent, the weights are updated more frequently, after the network has seen a single (randomly sampled) data point, instead of the entire data set. However, a drawback of SDG is that it might produce noisy jumps, as the updates are influenced by every single sample. A solution to this problem is using mini-batch gradient descent. A random subset of the data is shown to the network and a prediction for each sample is obtained. During the backward pass, the gradients for each data point are calculated and the error is optimized using an average of these gradients. The size of the data batches is arbitrary. To avoid non-convergence, it is common practice to use small batches with size 32 or smaller [45, 46].

Different optimization schemes have been proposed to optimize the SGD algorithm. One of the most commonly used is the Adam optimization scheme [47]. Rather than having a single learning rate for all the parameters of the network, the Adam method adapts a learning rate for each weight by using estimates of the first and second momentums of the gradient. This results in faster convergence of the algorithm.

2.4 Cross-Validation

Neural network architectures allow flexibility in terms of number of layers and number of neurons, to ensure the best possible approximation of the function f; this also means that the number of parameters becomes very large. A model with a large number of parameters can be trained almost to perfection, performing very well on the training data. However, it can happen that, when the model is evaluated on new, never-seen data, the performance drops as the model has a high fitness to the training data but will not fit the test set. The problem of having a perfect model on the training data, but that is not able to generalize on an independent dataset is known as overfitting. Overfitting arises in numerous cases and its causes are different. When designing a neural network-based

model, it is of paramount importance to avoid overfitting, since the ultimate aim of the model is to learn from labeled data and have reliable predictions on a novel, unlabelled data set.

Cross-validation (CV) is a technique used during model training, to avoid overfitting. The idea behind cross-validation is to use the entire data set split into training and test sets. In k-fold CV, the data set is split into k partitions. In a rotational manner, k-1 partitions will be used as training and one as test set. This generates one trained model. In the next step, the test partition is changed (also the training splits, accordingly), and a new model is trained. This procedure results in k trained models, one for each train-test combination. Figure 2.3a illustrates how the partitions are chosen in cross-validation. Once the k models have been trained, the predictions for an independent data set are given by an ensemble of the k models, i.e. an average of the predictions of the single models. The power of cross-validation resides in the fact that the model increases in robustness since an ensemble of models is used to make predictions [48, 49].

One of the most common causes of overfitting is over-training of the model. The longer the network is trained, the more the model is fitted to the training data. This will most likely lead to a situation where the training loss approaches 0 while the test loss increases. To balance this effect, early stopping can be used. The idea is to stop the training when the performance on the evaluation set starts dropping. The dataset is split into three parts: a training set, used to fit the model, a validation set, used to monitor the loss, and a test set used to evaluate the performance. Early stopping is applied by monitoring the training and validation loss. At the end of each epoch, early stopping makes sure that both the training and validation loss decrease. When the validation loss starts increasing or reaches a plateau phase, then the training is stopped and the best model is selected as the one achieving the minimum validation loss. This selected model will then be used to make predictions over the test set. Early stopping can be integrated into the k-fold cross-validation scheme, by adding an extra layer that loops over the partitions once more. Hence, according to the nested CV scheme, the model is trained on the training set and

the validation set, independent from the training, is used to select the early stopping epoch or to perform any other hyperparameter optimization to select the best performing. The test partition serves as an independent set to assess the model performance, as no training or optimization is performed on this set. Figure 2.3b shows the nested cross-validation on one test set. Also here, the data set is split into 5 partitions. For each iteration, one partition is selected as test set; the four remaining splits are used for training and validation, in all possible combinations. Hence, for each test partition, there are four train-validation combinations, producing 4 different trained models and the prediction over the selected test set is calculated by averaging the predictions of the four models. After looping over all the test sets, a total of 20 models are trained. An ensemble of all the 20 models is used to get predictions over an independent dataset.

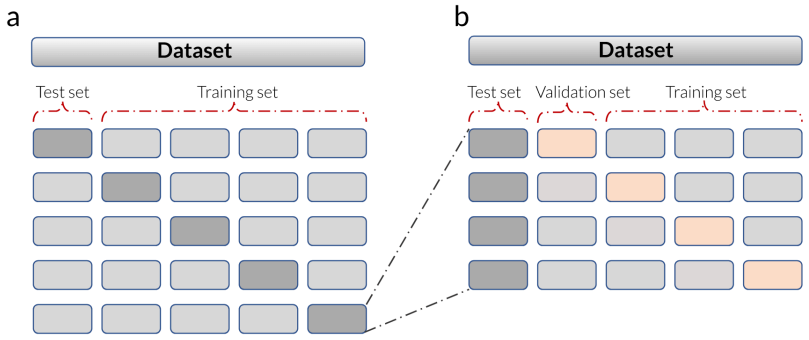


Figure 2.3: (a) Representation of 5-fold cross-validation scheme. The entire dataset is split into 5 partitions. Iteratively, one partition is used as a test set and the remaining 4 as training set. The overall performance is given by the test performances. (b) Nested 5-fold cross-validation on a single test fold. For each train/test split, the training set is further split into training and validation set; the validation data is used to perform model selection, including hyperparameter tuning and early stopping.

We have seen that cross-validation is widely used to reduce overfitting and it is based on the concept of splitting the data into independent training and test sets. The key point is to partition the data set making sure that the partitions are independent. An example of non-independent partitions is when some data points

are present both in the training and in the test set. When making predictions over the test set, the model will simply replicate the training predictions, leading to inflated test performance. Thus, it is very important to take care of data redundancy and partitioning to have a fair performance evaluation. In the context of biological sequences, it is not trivial to define independence between data points, as requiring only that the same sequences are not shared between partitions might not be enough. This aspect of redundancy reduction and data partitioning for protein sequences will be discussed in detail in Section 3.2.

2.5 Performance Evaluation

Several performance measures exist to evaluate the predictive power of an algorithm and to compare different models. Usually, the neural network outputs a real-valued prediction for each data point. When dealing with a classification task, e.g. binary classification, a threshold has to be set to classify the data points in predicted positives and predicted negatives. This is visualized in Figure 2.4. The positive points that are correctly classified as positives constitute the true positives (TP); similarly, the true negatives (TN) are the correctly classified negative points. When a labeled positive point is classified as negative, this is regarded as a false negative (FN); on the contrary, when a negative point is predicted to be positive, this is considered as false positive (FP).

An intuitive way of assessing the ability of a model that classifies data points into positives and negatives is using accuracy, defined as the fraction of correctly predicted examples. Accuracy has the advantage of being easily interpretable. However, a big disadvantage is that this metric does not take into account if the dataset is balanced, in terms of the number of positives and negatives. This means that high accuracy can be achieved simply because the data distribution is highly skewed towards one class and the model predicts all the data to belong to that class. Furthermore, it might not be trivial to decide on a specific threshold to build the confusion matrix.

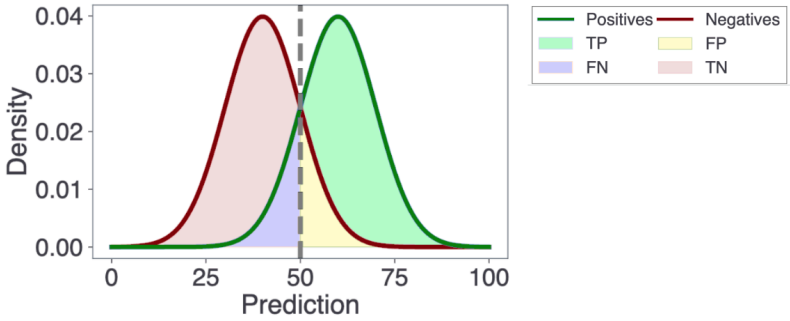


Figure 2.4: Visualization of the TP, TN, FP and FN after setting a threshold on the prediction values (the dashed line on 0.5 in the plot). All the predictions higher than the threshold are classified as positives and all the lower scores are considered negative predictions.

A non-parametric measure used in machine learning is the receiver operating characteristics (ROC) curve [50]. Rather than setting a fixed threshold, the confusion matrix for many values of the threshold is calculated. For a threshold $\bar{t}$, we can define the true positive rate, TPR=TP/(TP+FN), and false positive rate FPR=TP/(TP+FN). The ROC curve is defined as the TPR as a function of the FPR; thus, the ROC curve gives an indication on the model performance as the threshold $\bar{t}$ changes. The performance of a model can then be estimated using the area under the ROC curve (ROC-AUC, or simply AUC). The AUC value represents the probability of a positive data point scoring higher than a negative one [51]. Higher the AUC value the better the model is in separating positives and negatives. A perfect classifier will have an AUC of 1 while the AUC of a random classifier will be 0.5. Figure2.5 shows the process of generating a ROC curve, as $\bar{t}$ is changed. Another useful metric derived from the AUC is the AUC0.1, defined as the normalized area under the ROC curve, integrating the FPR up to 0.1. Thus, AUC0.1 focuses on the data points to which the model assigned a very high prediction score.

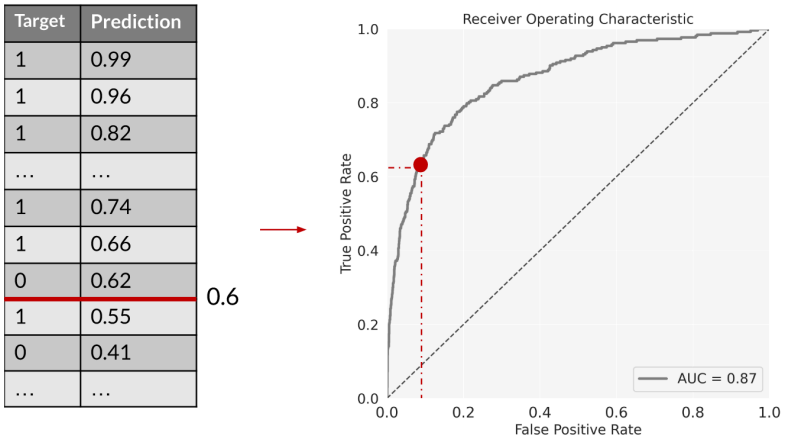


Figure 2.5: Generation of a ROC curve from the predictions, by varying the classification threshold. Once a threshold is set (shown by the red line in the table), the pair (FPR, TPR) for the specific threshold corresponds to the black point in the left plot. The entire ROC curve is generated by selecting multiple threshold values and computing the FPRs and TPRs.

CHAPTER 3

T Cell Immunoinformatics

The main focus of immunoinformatics is the development of data-driven algorithms to extract information and patterns from complex immunological data. One of the problems where immunoinformatics has made a massive contribution is the MHC I and MHC II binding prediction [52, 53]. These tasks are to a large extent solved and the developed models are able to generate nearly-perfect predictions. However, the peptide-MHC binding constitutes only one side of the T cell-mediated immune response. Especially in the last years, the attention has shifted towards understanding how the T cell interacts with the pMHC complex. Being able to understand how this interaction takes place, and potentially predict it, would pave the way to the development of novel T cell-based immunotherapies and rational design of vaccines [54–56].

The focus of this chapter is to provide a description of how immunoinformatic methods can be applied to T cells. First, the most commonly used TCR similarity measures will be defined, and their application to data cleaning and partitioning will be discussed. The last part of the chapter gives an overview of the state-of-the-art published models to predict TCR-pMHC interactions and what are the challenges involved.

3.1 TCR Similarity Measures

Defining a measure of similarity between TCR sequences plays a key role in different applications, from data partitioning to TCR specificity prediction. These applications will be discussed in the following Sections.

Different sequence similarity measures exist that can be applied to TCRs. Naively, the similarity between two sequences could be measured with the Hamming distance [57], i.e. the number of substitutions required to transform one string into another. The use of this metric is very limited as it requires the two sequences to have the same length, white TCR sequences are very diverse in terms of sequence length. An extension of the Hamming distance is the Levenshtein measure [58]. According to this metric, the distance between two strings is given by the total number of moves (insertion, deletion, substitution) needed to transform one sequence into the other, normalized by the length of the longest sequence.

Hamming and Levenshtein distance, however, are not biologically informed. They apply a uniform edit penalty to the sequences, no matter what is the position and what substitution is made. This scheme is sub-optimal when applied to protein sequences, as we know that some substitutions are more likely to happen in nature than others [59]. A suitable distance should take this aspect into account. One solution to this is using a substitution matrix, such as the BLOSUM matrix [60]. For each of the 20 amino acids, the BLOSUM matrix gives the observed probability for the amino acid of being substituted with any of the other 19. The BLOSUM scheme weights the alignment between two sequences based on the substitution probabilities. The BLOSUM score can be considered a similarity measure since it reflects how likely the protein folding will be conserved after the substitution, and hence its function. However, BLOSUM scoring might not be the optimal choice to assess TCR similarity. The substitution scores are based on the evolutionary likelihood of conservation or mutation, but CDRs (and in particular CDR3 loops) are stochastically generated through the V(D)J rearrangement and do not share any evolutionary relationship.

Another similarity measure capable of handling sequences with different lengths and that doesn't use sequence alignment is the kernel similarity measure [61]. The algorithm takes two sequences s_1 and s_2 as input. For each of the two sequences, a list of all the possible k-mers is generated, with k ranging from 1 to the length of the shortest sequences. Iteratively, for each value of k, all the k-mers of s_1 are aligned to the k-mers in s_2 and the BLOSUM62 score between these is computed; for each k-mer, all the BLOSUM scores are then multiplied. Lastly, all the k-mers products are summed and normalized, so that the k-mer scoring ranges from 0 to 1, where 1 means a perfect match between the two sequences. Kernel similarity is not influenced by varying sequence lengths, as all the possible k-mers of equal length are compared. Further, this metric places more emphasis on the central part of the sequences, as the central residues fall into more k-mers and are involved in the similarity calculation more often compared to the terminal parts. This aligns with the biological understanding of CDRs-peptide interaction; especially for the CDR3 loops, the termini are conserved while the central part of the loops is more variable and is in closer contact with the peptide [14].

3.2 Similarity Reduction and Data Partitioning

The aim of any machine learning model is to learn from the training data and to be able to generalize over unseen data, giving a reliable prediction over a test data set. It is of crucial importance that the training and the test data are disjoint and independent. If a subset of the data is shared between the two sets, the model will make good predictions over these data points simply because it remembered these from the training set. Hence, the overall test performance will be inflated.

When dealing with biological sequences, it is not trivial to define the meaning of independence between sequences. For example, let us consider the problem of predicting TCR specificity. One could define a similarity metric between two TCR sequences as the sequence identity. When partitioning the data into training and test, a simple approach to ensure having independent partitions would be to place sequences with 100% sequence identity in

the same partition, to avoid data leakage between training and test set. However, this criterion might not be enough. In fact, if two TCRs have one single mutation (hence not being 100% identical), they most likely recognize the same epitope because they have a high sequence similarity. If we partition based solely on sequence identity, these two TCRs might fall into different partitions and the performance of the classifier will be overestimated since the model didn't learn the signal but only remembered the specificity of the TCR seen in the training set. This example shows that a suitable partitioning scheme should be implemented with protein sequences. The scenario described above is usual in the TCR sequence space, where many sequences with the same specificity differ from each other for one or two mutations. A good data pipeline should take into account this aspect, and avoid data leakage between train and test splits to prevent performance overestimation.

A widely used method for reducing the redundancy in sequence data is the Hobohm 1 algorithm [62]. This method sorts the data into a redundant and non-redundant list. The algorithm starts by sorting the sequences according to a criterion of interest, for instance, TCR length in descending order. The TCR on top of the list is placed in the non-redundant list; iteratively, all the candidate TCRs will be compared to the non-redundant ones, using a suitable similarity measure: if the new TCR has a similarity higher than a certain threshold to any of the TCRs in the non-redundant list, then it is discarded and placed in the redundant stack. On the contrary, if a new TCR doesn't share high similarity with any of the non-redundant TCRs, this is considered unique and placed in the non-redundant list. The algorithm terminates when all the TCRs have been scored against the non-redundant TCRs. The output of the Hobohm 1 algorithm is a list of TCRs where any possible pair of sequences have a similarity that is equal to the similarity threshold, at most. Because some of the TCRs are redundant, the set of unique TCRs is reduced in size. The non-unique TCRs are either discarded from the data set or reintroduced, making sure that they will be placed in the same partition as their most similar TCR in the unique list. Depending on the problem to be analyzed and the nature of the data, different

similarity measures can be adopted here; some examples of such measures were presented in Section 3.1. Regarding the choice of the threshold, this is also arbitrary and depends on how strict the redundancy criterion should be, i.e. how different and separated we require the resulting partitions to be.

Once the highly similar TCRs are removed from the set and the redundancy is reduced, it is possible to partition the dataset and get k clusters to be used for k-fold cross-validation. The Hobohm 1 algorithm ensures that, regardless of the way the data is partitioned at this point, there will not exist any pair of redundant TCRs across partitions. Different approaches can be used to split the data. The simplest one is to randomly assign the TCRs to the k clusters. Another more sophisticated approach uses similarity-based graphs to cluster the data. A random TCR is selected from the redundancy-reduced set. This TCR is then connected to all the TCRs that have a similarity higher than a new threshold, referred to as the partitioning threshold. After, all the sequences that were connected to the initial TCRs are, in their turn, connected to the similar TCRs. This process continues until no other TCRs can be merged into clusters. This construction of the similarity graph continues until all the TCRs have been touched by a graph, or only singlets with no similarity to other TCRs are left. This procedure results in many components that can be now merged into k partitions. One of the problems that might arise with this approach is that most of the data could fall into the same, big cluster, leading to partitions with very unbalanced sizes.

3.3 Protein Sequence Encoding

The input of the neural networks should be numerical. A critical step in designing a deep learning algorithm is to choose an appropriate way of representing protein sequences into numerical values. This step is referred to as sequence encoding. A first simple approach is to use the so-called one-hot encoding. In this scheme, each amino acid is represented as a vector of length m, where m is the length of the amino acid alphabet. For a given amino acid, the vector contains 1 in a position that is unique for the specific letter and 0 elsewhere. One-hot encoding produces a sparse rep-

resentation of the sequences, where the vector representing each amino acid is orthogonal to all the other amino acids. While this approach might work well when there is no clear dependence between the inputs, it is too simplistic in the context of protein sequences because amino acids at different positions are not independent. One drawback of representing the amino acid with a set of orthogonal vectors is that the pairwise distances between the vectors are the same. However, we know that some amino acids share some similarities in terms of physico-chemical properties and that exchanging similar amino acids will result in no change in the structure or function of the protein [63].

A more specialized amino acid encoding uses substitution matrices, such as BLOSUM50 [60]. These substitution matrices contain the observed probabilities of an amino acid remaining unchanged or being exchanged with the other 19. Hence, also according to this scheme, each letter of the alphabet is represented as a vector with 20 columns. This vector representation contains more information compared to one-hot encoding, as similar residues in the protein sequence will have a similar representation [64].

A more recent approach in amino acid encoding consists in letting a neural network learn the representation. In this case, the amino acids representation is a set of parameters that have to be learned during the training process, together with all the other network weights. This approach is heavily data-driven and it is task-specific. In fact, the encoding is updated at each iteration of the backpropagation algorithm, with the aim of minimizing the error. Learning the encoding from the data is typically used in recurrent neural networks, such as LSTM, or in transformers architectures [65].

3.4 Predicting TCR Specificity

Peptide-MHC recognition by T cell receptors represents the cornerstone of T cell mediated immunity. Developing computational models to predict TCR interaction with the pMHC is highly desirable. It would deepen our current understanding of how T cells interact and would pave the way to personalized immune treat-

ments and targeted vaccine development. However, this task is highly challenging and the available prediction models work under specific circumstances and are not fully able to generalize to never-seen epitopes.

This Section describes a fairly simple but efficient way of predicting TCR specificity using TCR similarity. The second part gives an overview of the current state-of-the-art prediction models.

3.4.1 Similarity-based Modeling

A first attempt to predicting TCR specificity consists of using sequence similarity to assess what is the likelihood of a TCR binding to a specific epitope. The main hypothesis behind these models is that, even though the TCRs are very diverse, TCRs binding to the same pMHC complex have a higher sequence similarity compared to TCRs with a different specificity [66]. Among the unsupervised, similarity-based algorithms that attempt to predict TCRs specificity, there are TCRdist [67], GLIPH [68, 69], TCRMatch [66], and TCRbase [70].

TCRdist deploys the BLOSUM scoring scheme to compute distances between TCRs. The TCRdist measure is defined as a weighted distance between αβ-pairs of CDR2, CDR2.5 and CDR3 sequences. The distance between the loops is calculated as an alignment score but weighted by the BLOSUM62 matrix. The TCRdist measure is used to compute an all-against-all similarity matrix that is used for clustering the TCRs using the k-Nearest Neighbors (kNN) algorithm.

GLIPH is an unsupervised clustering algorithm that attempts to cluster TCRs with the same specificity using conserved motifs and similarities between CDR3 loops. The distance between two sequences is calculated as the motif frequency of k-mer frequencies (k=2,..,5), compared to a background distribution of expected frequencies.

In the TCRMatch publication, the authors benchmarked a set of seven different similarity measures to predict TCR specificity. In the proposed models, TCR similarity is calculated using the kernel similarity measures on the CDR3β sequences. The algorithm

works as follows: define a query as a set of CDR3β for which a prediction is needed. Each of the query TCRs is scored against a database, formed by the peptide-TCR pairs present in the IEDB. The prediction for a given peptide-TCRs is then given by the similarity of the evaluation TCR to its nearest neighbor in the IEDB. Figure 3.1 shows a schematic representation of how this method works.

We have developed TCRbase [70], an extension of the TCRMatch algorithm. The main extension present in TCRbase is that input space is not restricted only to the CDR3β loop, but the users can choose to use also the paired CDR3αβ sequences, or also the entire set of the six CDRs. In the case multiple sequences are used, for instance in the case of CDR3αβ sequences, the kernel similarity score is given by a weighted average of the kernel similarities of the single loops. If all six CDRs are input to TCRbase, the CDR3 loops should be weighted four times higher than the CDR1 and 2, in the linear combination of similarities [71]. Lastly, differently from TCRMatch, the user can choose a custom database in TCRbase, not being limited to only IEDB. The algorithm described in Figure 3.1 also applies to TCRbase, as TCRMatch and TCRbase conceptually work in a very similar way.

3.4.2 Available Tools

With the advent of novel machine learning models and increasingly powerful computational infrastructures, a vast variety of models were recently developed to predict TCR specificity. The published works try to tackle the problem of predicting TCR-peptide interaction from different perspectives; the supervised modeling approaches range from classical machine learning algorithms, such as random forests, to neural network-based models, such as CNN or LSTM, to the recently developed transformers architectures.

Among the different deep learning frameworks, convolutional neural networks seem to be the most widely applied architectures across many publications. TCRAI [17] is a CNN-based model that runs convolutions on CDR3αβ sequences and combines the hidden representation with the V and J genes information to predict binding between a TCR and an epitope. In the DeepTCR publication

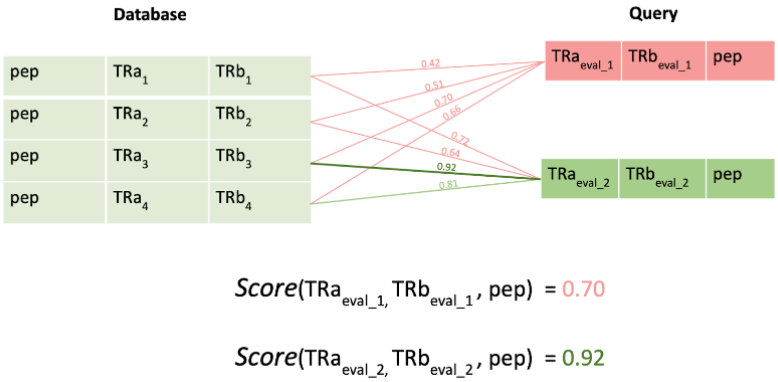


Figure 3.1: Schematic representation of a similarity-based modeling approach, such as TCRMatch or TCRbase. The database, or training set, contains positive peptide-specific TCRs, while the query, or evaluation set, contains the TCRs relative to that specific peptide, both positive and negative, the model has to make predictions on. Each TCR in the query is scored against all the TCRs in the database; the prediction for an evaluation TCR is given by the similarity to its nearest neighbor in the training set.

[72] the authors also make use of convolutions but applied them to a variational autoencoder (VAE) [73] : in VAEs, the inputs are mapped into a latent space and reconstructed back, while minimizing the reconstruction loss. The latent space representation is a high dimensional Gaussian distribution of the input data and represents a compressed version of the input, but with no loss of information. This latent representation can be used to perform downstream tasks. In DeepTCR, the CDR3 loops and the epitope are mapped to a hidden space by convolutional layers and they are deconvoluted to reconstruct the input. The resulting compressed representation is used to perform the TCR-epitope binding prediction task. In TITAN [74], a one-dimensional CNN is combined with a context attention map to predict the binding probability: during the training of the model, a set of attention weights is learned, assigning an importance factor to each residue in the sequences. Lastly, also ImRex [75] uses convolutional layers but on a novel representation of the CDR3 loops and epitope, which is built using physicochemical properties of the amino acid sequences. In

the context of CNNs, we have developed two methods based on 1D CNN, NetTCR-2.0 [18], and the updated NetTCR-2.1 [70]. We investigated the impact of using only CDR3β chains or paired αβ data; we also examined if the model would benefit from having the entire set of all six CDRs as input.

ERGO [76] and ERGO-II [77] also use deep learning architectures, but focus on language models and natural language processing (NLP). In their first publication, the authors proposed a method to predict TCR-epitope interaction using only CDR3β information as input to an LSTM network or a VAE. Subsequently, they expanded their model to accept also CDR3α chain, HLA allele, T-cell Type and V/J genes.

Recently, deep learning research has focused on deploying transformer models [65] in many different fields, given their ability to work very well across different domains. One of the transformer architectures that has attracted attention, especially in protein research, is BERT [78], designed to pre-train deep bidirectional representations from an unlabeled text by jointly conditioning on both the left and right contexts. In TCR-BERT [79], a BERT-like architecture is used to train a self-supervised algorithm on a large corpus of unlabeled CDR3 α and β sequences. This generates a position-specific, context-dependent representation of the amino acids; the learned embedding is then applied to the labeled TCRs to perform different downstream tasks, including CDRαβ-peptide binding prediction.

More classical machine learning approaches to model TCR interactions include TCRex [80], TCRGP [81], and SETE [82]. These techniques approach the problems differently. TCRex is a random forest-based method to identify epitope-binding TCRs. TCRGP is a method based on Gaussian processes to predict if TCRs recognize specified epitopes. Lastly in SETE, the authors use a k-mer feature representation of adjacent amino acids in combination with principal component analysis and decision trees.

3.4.3 Current Challenges

A big effort is being put in by the research community to investigate TCR-pMHC interactions and develop models to predict

TCR specificity. However, the task is challenging and remains largely unsolved. One of the main challenges that the field is facing is related to the data. The data available at the moment is scarce and represents only a small subspace of the very vast and diverse space of TCRs. Furthermore, the coverage on different epitopes and HLA molecules is low: the available data covers few peptides and only the most common HLAs. In order to build a general predictor, the model should be exposed to a broad range of peptides and HLAs, to be able to detect differences across the different molecules. Moreover, most of the available data only contains information about CDR3β chains, as bulk sequencing as described earlier does not allow the generation of paired αβ TCR data. Single-cell sequencing, instead, is promising since it allows sequencing of both al and β chains of the TCRs, in a high-throughput way. However, single-cell technologies for TCR-peptide sequencing are still being developed and are generally characterized by a low signal-to-noise ratio.

Another problem lies in the fact that most public data sets only contain positive examples of binding pMHC-TCR pairs. A model capable of separating interacting and non-interacting pairs should be exposed to both positive and negative interactions. However, it is not trivial to define negative data starting from positive pMHC-TCR pairs. One proposed solution is to generate negative data by pairing a TCR with a peptide that is different from its target cognate. This, however, might introduce false negatives in the data, making it challenging for the model to learn to separate positives and negatives. Another approach would be to use TCRs from healthy control data and pair them with the non-self peptides, assuming that the donors had never been exposed to the corresponding virus or pathogen. This however often might also be sub-optimal since healthy control TCRs are likely positive to dominant peptides derived, for instance, from influenza virus, cytomegalovirus (CMV) or Epstein-Barr virus (EBV). Hence, pairing healthy control TCRs with these peptides might again introduce falsely labeled negative TCRs in the dataset.

Lastly, there is no consensus on how to model the peptide-TCR interaction. For a long time, CDR3β was considered the only part

of the TCR that played a role in the TCR-peptide interaction. The reason for this is that the CDR3β in most crystal structures is found to lie closer to the peptide compared to CDR3α ; furthermore, the CDR3β is the more diverse chain, as also the D gene is involved in the DNA rearrangement of the β chain. However, it is becoming more and more clear that the CDR3β is only one of the components involved in the peptide-TCR interaction. Most of the publications regarding computational models to predict TCR binding are showing that also the α chain contributes, and that including this chain in the modeling results in better performance. On the same line, it is still up for discussion if the integration of all the CDR loops, or the full TCRαβ sequences, would have an impact.

The work presented in this thesis aims to analyze each aspect involved in building a pMHC-TCR interaction classifier. Paper I presents a CNN-based approach to model peptide-TCR interaction, showing the contribution of both α and β CDR3 loops. Special attention was given to curating a training data set, in terms of data redundancy and data partitioning. Paper II is the natural extension of the previous work; it aims to enlarge the set of peptides and HLA molecules and it shows the contribution of the CDR1 and CDR2 loops. Lastly, Paper III presents a data-driven approach to filter the data set that aims to remove inaccurate peptide-TCR pairs, increasing the signal-to-noise ratio.

CHAPTER 4

Prediction of TCR-peptide binding by using paired TCRα and β sequences

This chapter presents the work on NetTCR-2.0, a deep learning model used to predict TCR specificity. NetTCR-2.0 uses convolutional neural networks to predict whether a given TCR binds a specific peptide. NetTCR-2.0 was trained on publicly available data coming from IEDB and 10x Genomics datasets, and validated on a novel dataset, generated in-house.

Historically, the research about predicting TCR specificity focused on using CDR3β data, as most of the available data lacked information about paired αβ chains. With the advent of single-cell sequencing, datasets containing paired chains were made available. The main contribution of the project was to show that both α and β chains of the TCR are needed to achieve better performance. The models trained on αβ paired data consistently outperformed the ones trained on single chain data, and in some cases, the α chain was found to be even more informative than the β .

Even though NetTCR-2.0 is a pan-specific model, meaning that it can make predictions on any epitope, the model can make reliable prediction only for three HLA-A*02:01 presented peptides. We estimated that approximately 150 positive TCRs for a given peptide are needed to train a model that is able to successfully generalize on unseen TCRs.

The NetTCR framework is flexible and can be easily expanded to integrate more inputs, such as the MHC molecule, V/J genes or the full sequence of the TCR.

communications biology

ARTICLE

https://doi.org/10.1038/s42003-021-02610-3 OPEN

NetTCR-2.0 enables accurate prediction of TCR-peptide binding by using paired TCRα and β sequence data

Alessandro Montemurro[1], Viktoria Schuster[1], Helle Rus Povlsen[1], Amalie Kai Bentzen[2], Vanessa Jurtz[1], William D. Chronister[3], Austin Crinklaw[3], Sine R. Hadrup[2], Ole Winther[4,5,6], Bjoern Peters[3,7], Leon Eyrich Jessen[1] & Morten Nielsen[1,8]✉

Prediction of T-cell receptor (TCR) interactions with MHC-peptide complexes remains highly challenging. This challenge is primarily due to three dominant factors: data accuracy, data scarceness, and problem complexity. Here, we showcase that "shallow" convolutional neural network (CNN) architectures are adequate to deal with the problem complexity imposed by the length variations of TCRs. We demonstrate that current public bulk CDR3β-pMHC binding data overall is of low quality and that the development of accurate prediction models is contingent on paired α/β TCR sequence data corresponding to at least 150 distinct pairs for each investigated pMHC. In comparison, models trained on CDR3α or CDR3β data alone demonstrated a variable and pMHC specific relative performance drop. Together these findings support that T-cell specificity is predictable given the availability of accurate and sufficient paired TCR sequence data. NetTCR-2.0 is publicly available at https://services.healthtech.dtu.dk/service.php?NetTCR-2.0.

[1] Department of Health Technology, Section for Bioinformatics, Technical University of Denmark, DTU, 2800 Kgs, Lyngby, Denmark. [2] Department of Health Technology, Section for Experimental and Translational Immunology, Technical University of Denmark, DTU, 2800 Kgs, Lyngby, Denmark. [3] Center for Infectious Disease and Vaccine Research, La Jolla Institute for Immunology, La Jolla, CA 92037, USA. [4] Department of Biology, Bioinformatics Centre, University of Copenhagen, 2200 Copenhagen, Denmark. [5] Section for Cognitive Systems, Department of Applied Mathematics and Computer Science, Technical University of Denmark, Kgs., Lyngby 2800, Denmark. [6] Centre for Genomic Medicine, Rigshospitalet, Copenhagen University Hospital, København Ø 2100, Denmark. [7] Department of Medicine, Division of Infectious Diseases and Global Public Health, University of California, San Diego, La Jolla, CA 92037, USA. [8] Instituto de Investigaciones Biotecnológicas, Universidad Nacional de San Martín, Buenos Aires, Argentina. ✉email: morni@dtu.dk

T cells survey the health status of cells by scrutinizing their surface for the presence of foreign peptides presented in complex with major histocompatibility complex (MHC) molecules. This recognition by the T cell is facilitated by the T-cell Receptor (TCR). This crucial interaction between TCRs and peptide-MHC (pMHC) molecules thus forms a molecular switch defining a bottleneck for immune activation. Understanding the rules governing this interaction hence represents a paramount step in both personalized immune treatment and development of targeted vaccines.

The TCR is a heterodimeric protein, consisting of an α- and β-chain. The subpart of the TCR interacting with the pMHC complex is defined by six loops, three for each α- and β-chain. These loops determine the specificity of the TCR and are denoted complementarity determining regions (CDRs) 1–2–3. The current consensus is that the CDR3 loops primarily interact with the peptide, while the CDR1 and CDR2 loops interact with the MHC[1–3]. The peptide specificity is thus predominantly defined by the CDR3 loops. The diversity of the CDR3s is defined by the genomic recombination of the variable, diversity, and joining (VDJ) TCR-genes. However, while the α-chain is the result of a V- and J recombination, the β-chain contains the V-, D- and J genes creating a broader diversity. The result of this is that most data-generating studies have focused on the β-chain alone.

The majority of the publicly available TCR-pMHC-specificity data resides in the Immune Epitope Database (IEDB)[4], VDJdb[5], and McPAS-TCR[6], all of which primarily contain CDR3β-data. Several recent works have demonstrated the important shortcoming of this limited view on the TCR and demonstrated how the information on the specificity of the TCR toward its cognate pMHC target is carried by CDR3 of both α- and β-chains[7,8]. To investigate the pMHC specificity on paired α-/β-chains, single-cell (SC) technology is required. SC is considerably more costly, and thus much less paired-specificity data are publicly available. This is a critical shortcoming of current databases and highlights the urgent need for further development of cost-efficient SC technologies capable of accurate high-throughput paired-data generation[9].

While cost-efficient and accurate state-of-the-art high-throughput technologies for experimentally and computationally assessing the binding of a peptide to an MHC are available[10–12], for reasons explained above, the TCR component of the triad remains highly cost-intensive and low throughput and sparsely explored. This represents a major challenge in moving the field forward.

A number of studies have been published related to the prediction of TCR-pMHC interactions[7,13–21]. They present a wide range of data and modeling techniques. Most are constructed based on data from the IEDB, VDJdb, and/or McPAS-TCR and, in addition to the epitope information, make use of either CDR3β sequences alone[13–15], a mixture of CDR3α and CDR3β sequences[16], or smaller data sets entailing all 6 CDR3 sequences and potentially additional cellular information[17,18]. Methodologically, the different studies range from simple CDR3β alignment-based methods[19,22], over CDR similarity-weighted distances such as TCRdist[7], k-mer feature spaces in combination with PCA and decision trees (SETE[13]), random forests[20,21] such as TCRex[23], CNN-based (ImRex)[16], and Gaussian process classification methods (TCRGP[17]), to more complex approaches integrating natural language processing (NLP) methods (ERGO[14]). The overall conclusion from these earlier works is that while the prediction of TCR specificity is feasible, the volume and accuracy of current data limit the performance of the developed models. Moreover, these earlier works only to a limited extent address the high degree of redundancy present in TCR-interaction data sets, making it difficult to assess the generalizability of the developed models.

We have earlier proposed a simple 1D CNN-based model, NetTCR-1.0[15], integrating peptide and CDR3β sequence information into a model for the prediction of TCR peptide specificity. Using a similar modeling framework, we here present an in-depth analysis of publicly available TCR-pMHC interaction data, with an emphasis on investigating the impact of data limitations and quality on model performance. Furthermore, the performance of the developed model is compared with simpler sequence-based models as well as more complex deep learning approaches and the impact of training on paired versus single-chain TCR-sequence data is investigated.

Results

We set out to develop and benchmark models for the prediction of TCR-pMHC binding with a particular focus on investigating the quality of different data types, and the effect of using paired CDR3α/β versus CDR3β information only.

We started with data obtained from the IEDB, consisting of 9204 unique CDR3β sequences, each labeled to bind a single pMHC complex, and 387,598 negative data points derived from 10X single-cell sequencing (for details see "Materials and methods"). This data set is referred to as the β-chain data. Another, but smaller set of positive data points, was derived from combining IEDB and VDJdb data providing both CDR3α- and CDR3β-chain, leading to a paired chain set of 2744 unique TCR-peptide data points. The available data were highly heterogeneous in terms of studied peptides and HLA alleles with a majority (62%) of the IEDB data being restricted to HLA-A*02:01. Likewise, the vast majority of the HLA-A*02:01 restricted peptides were of length 9. Given this, for the further part of this work, we limited the analysis to HLA-A*02:01 and 9-mer peptides. Supplementary Fig. 1 presents TCR counts in the positive data sets for the three most abundant peptides NLVPMVATV (NLV) from human herpesvirus 5 (cytomegalovirus), GILGFVFTL (GIL) from influenza A virus, and GLCTLVAML (GLC) from human herpesvirus 4 (Epstein–Barr virus) in the two data sets. These three represent 99% and 92% of the β-chain and paired-chain data, respectively.

Model performance: CDR3β data. In a first attempt to evaluate the possibility of predicting TCR-peptide interactions, prediction models were constructed from the TCRβ data set. A critical part of the model development and evaluation relates to the procedure implemented for data preparation in the context of data redundancy and partitioning. Models were therefore trained and evaluated using cross-validation on different CDR3β data sets, characterized by different degrees of interpartitional redundancies. The performance was further evaluated on an external data set. For details on the data set preparation and interpartitional redundancies, refer to "Materials and methods". Here, two models were investigated, a sequence-similarity and a 1D CNN-based (NetTCR) model. The sequence-similarity-based model (baseline) serves here as a benchmark to investigate the added benefit of modeling the data using the more complex CNN framework. Performance of deeper and different neural network architectures was investigated subsequently. Cross-validation performance results as a function of the partitioning thresholds are shown in Fig. 1a. Here, the baseline model demonstrated the expected strong association between internal data redundancy and model performance, with a substantial and highly significant ($p < 0.0001$, bootstrap test with 10,000 replications) drop in performance as the partitioning threshold is decreased (from an AUC value of 0.67 at 99% to 0.63 at 90%)—resulting in a lower

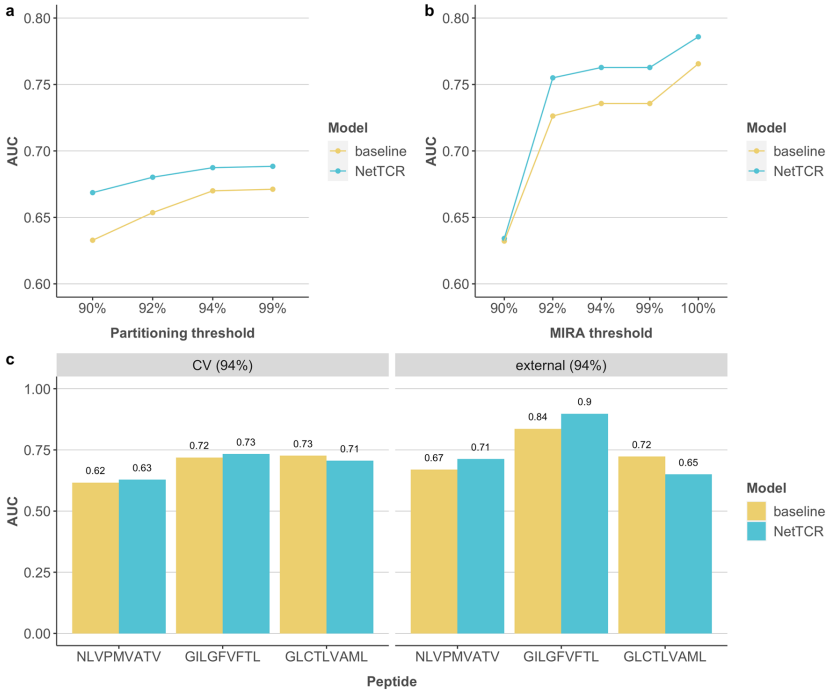


Fig. 1 Performance of models trained on CDR3β data alone. a Overall AUCs evaluated via cross-validation of different training data-partitioning thresholds for the baseline model and NetTCR. Partitioning thresholds are indicated in percent on the x-axis. **b** Overall AUCs evaluated on the MIRA sets at different thresholds (shown on the x-axis) using the model trained on the 94% similarity-partitioned data. The MIRA threshold represents the degree of separation between the training set and the MIRA set. **c** Peptide-specific AUCs for 94% partitioned cross-validation (CV) and external evaluation with a similarity threshold of 94%, colored by model.

similarity between the training and test data sets. This dependency on the partitioning threshold is diminished for the NetTCR neural network method. The performance of the NetTCR method was low even at the highest partitioning threshold with a maximum AUC of 0.69.

We next evaluated the performance of the models trained on the 94% partitioned data on the independent MIRA data set (Fig. 1b) using an ensemble of the 20 models obtained from cross-validation. Five different MIRA datasets were obtained by imposing a separation from the training set of 90, 92, 94, 99, and 100% similarity. That is, MIRA 94% TCRs do not share more than 94% Levenshtein similarity to any of the TCRs in the training set. Overall, this benchmark revealed a higher performance of all models compared to that observed in the cross-validation with a performance value of up to 0.79 in AUC. This performance is higher than the best-performance values observed during cross-validation and suggests that the MIRA data share an overall higher quality compared with the IEDB data used for training (for further discussion of this see later). Also here, the NetTCR method outperformed the baseline model, and we likewise observed a continued drop in performance of the models as the similarity between the evaluation and training data sets was diminished. This drop was particularly large for the 90% similarity threshold where all models achieved a comparable performance of AUC 0.635. Similar results were obtained for the models trained using other partitioning thresholds (see Supplementary Fig. 2).

Figure 1c displays the peptide-specific AUCs in cross-validation and the external evaluation (defined using a 94% similarity threshold) of the models trained on the 94% partitioned training data set for the three dominant peptide sequences in the training data set. These peptide-specific AUCs strongly suggest that the model performance does not correlate with the amount of training data. That is, the performance of the NLV peptide characterized by the largest amount of training data displayed the lowest performance value in both the cross-validation and MIRA evaluation. Additionally, the neural network method did not in this evaluation perform overall better than the baseline model.

In conclusion, the observed relatively low predictive performance—even at high interpartitional redundancies—and the lacking correlation between data set size and predictive performance, suggest that TCR-peptide interactions can only to a very limited extent be characterized using current CDR3β-peptide data.

To further elaborate on this conclusion, and to ensure that it was not a result of the data set and/or modeling framework investigated here, we extended the benchmark to include the recently published ERGO method[14]. ERGO predicts peptide-TCR binding using long-short term memory (LSTM) networks or autoencoders (AE). Both network architectures were trained on

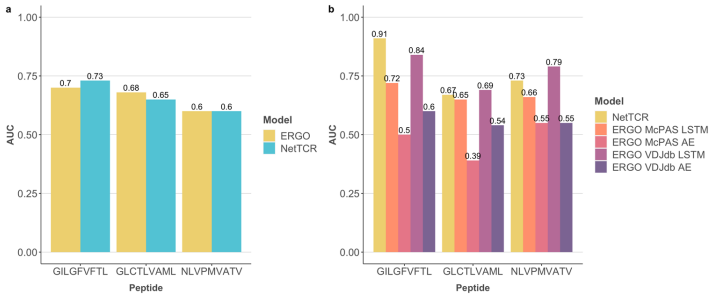


Fig. 2 Comparison between NetTCR and ERGO. a Test AUCs per peptide for NetTCR and ERGO trained on four out of five partitions of the IEDB + 10X data set and evaluated on the left-out partition. **b** Peptide-specific AUCs for NetTCR and all the four variants of ERGO evaluated on the MIRA data.

data sets derived from VDJdb and/or McPAS. Training NetTCR and the LSTM-based ERGO on four out of the five partitions of the IEDB + 10X data set and evaluating both models on the left-out partition, we observed that NetTCR and ERGO shared comparable performance in terms of peptide-specific AUC (see Fig. 2a) and both models have an overall AUC of 0.66. We further tested the performance of NetTCR trained on the complete IEDB + 10X data set and all the variants of ERGO on the MIRA data. In this case, NetTCR achieved an overall AUC of 0.77 and outperformed the best ERGO model (LSTM trained on VDJdb), which achieved an AUC of 0.74 (see Fig. 2b). These results show that NetTCR has a comparable performance to that of ERGO, hence demonstrating that the relatively low performance for TCR-peptide interactions observed here for NetTCR and the baseline is not imposed by the limited complexity of these models, compared with ERGO. Further, the results suggest that simple shallow models like the CNNs used here, rather than more sophisticated architectures, are sufficient to achieve optimal performance for the prediction of TCR-peptide specificity (at least given the current data).

Model performance: paired CDR data. Given the low performance of the CDR3β models, we next moved toward data sets consisting of both CDR3α and CDR3β. Figure 3a shows the overall and peptide-specific cross-validation AUC performance value of the baseline and NetTCR models trained on different TCR chain components for data sets created at 90% and 95% partitioning threshold. Here, data sets including both α- and β-chains, were partitioned by the average similarity of CDR3α and CDR3β. These partitions were maintained when training and evaluating models on α- or β-chains alone. The results from a chain-specific partitioning approach are included in Supplementary Fig. 3. These results in Fig. 3a demonstrate a comparable performance for models based on the CDR3α or CDR3β information and superior performance when including both the α- and β-CDR3 information for both the NetTCR and baseline models. With an overall AUC performance of 0.89, NetTCR significantly ($p < 0.0001$, bootstrap test with 10,000 replications) outperformed the baseline model. Further, the performance of the NetTCR model was found to be maintained when trained on the 90% compared with the 95% partitioned data. This was in contrast to the baseline model that suffered a significant drop in performance ($p = 0.006$, bootstrap test with 10,000 replications) when lowering the partition threshold. These observations are confirmed in Figs. 3b and 3c by the peptide-specific AUCs derived from the 90% and 95% partitioned data, respectively. Also here, and for both partitioning thresholds, the NetTCR model, including both the α- and β-chain information, outperformed all other models, and both single-chain models achieved a lower but comparable performance. Investigating in more detail the effect of the size of the training data on the predictive performance of the two models, Fig. 3d displays the peptide-specific cross-validation AUC for the set of peptides included in the training data. Overall, this figure shows a decrease in AUC as the number of positive data points present in the training data drops, with an average AUC of NetTCR for peptides characterized by 200 or more TCRs of 0.88, and an average of peptides characterized by 20 or fewer TCRs of 0.38. One clear exception from this was the FLYALALLL peptide with only 37 binding TCRs and an AUC of 0.94. This potential outlier can however be explained by comparing the sequence similarities between positive and negative data points. Estimating a difference in similarity per positive TCR as the maximum similarity to all other positives for the given peptide in other partitions minus the maximum similarity to all negatives for the same peptide in other partitions, the expectation is that a higher dissimilarity between positives and negatives for a given peptide would ease the discrimination task, resulting in a higher peptide-specific performance value. This was confirmed by the result shown in Supplementary Fig. 4, where the AUC displays a clear tendency to increase as a function of the similarity difference (a Spearman correlation between AUC and median difference in similarity of 0.63). This result thus supports that FLYALALLL is an outlier and its high performance is imposed by the high difference in similarity score between its positive and negative TCRs.

Overall, these results suggest that consistent and high-performing models for TCR-pMHC interaction predicting can be developed from paired TCR data and that the low quality of current models is imposed by the low quality of bulk-sequenced CDR3β data. To further quantify this, we went back to the model trained on the bulk CDR3β data and evaluated using cross-validation the performance of a subset of 500 positive CDR3β shared with the paired TCR data sets, and an equal-size data set of positive CDR3β not sharing an overlap with the paired TCR data set. Both sets of positive TCRs were evaluated in the context of the complex negative dataset. The results of this experiment confirmed the high quality of the shared CDR3 data with an AUC of 0.80, and the likewise lower performance (AUC = 0.68) of the CDRs not shared with the single-cell data. Further, we evaluated the model trained on the 95% partitioned CDR3β data from the paired TCR data set on the CDR3β MIRA data (excluding identical overlap to the training data). This resulted in an overall AUC of 0.81. This performance is lower than the cross-validated performance but slightly higher than the performance of 0.79 demonstrated in Fig. 3b for the CDR3β-alone model. These results demonstrate that the MIRA data have a quality

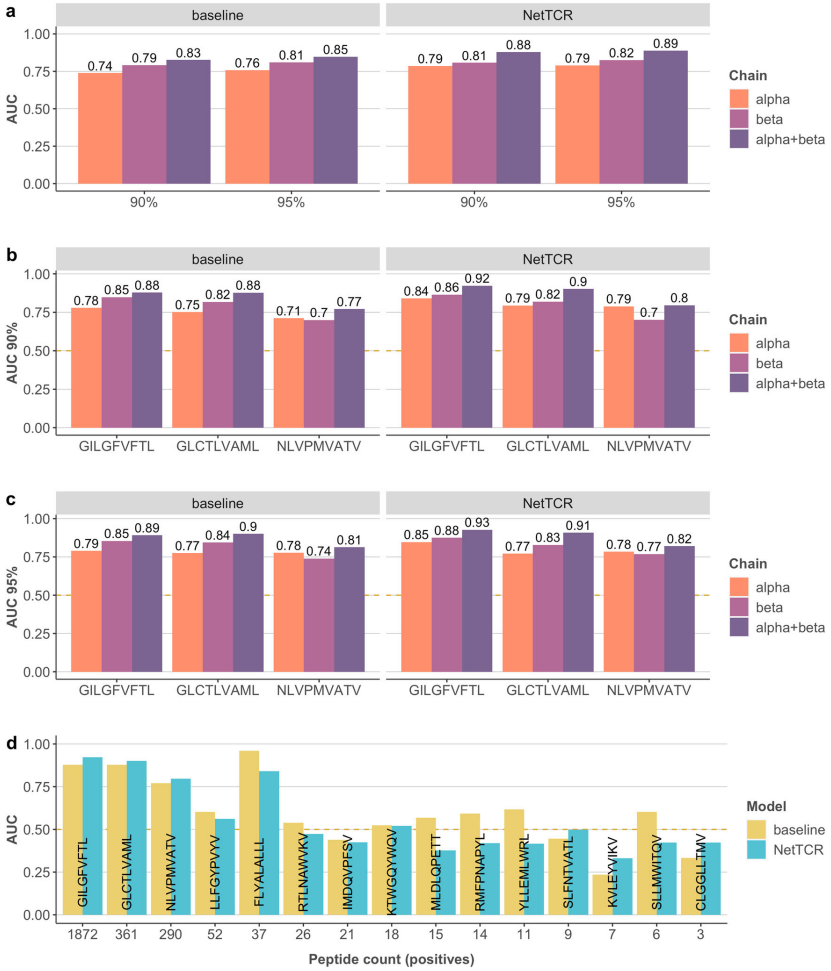


Fig. 3 Performance of models trained on paired-chain data. a Overall AUCs evaluated via cross-validation. **b**, **c** Peptide-specific AUCs from the 90% and 95% partitioned data for the three most frequent peptides. **d** Peptide-specific AUCs colored by model and plotted against the number of positive data points.

comparable to that of CDR3β from the paired TCR data, and thus, in line with the observation earlier, suggest a higher accuracy of these data compared with the overall accuracy of the bulk CDR3β alone data.

To further validate the high performance of the NetTCR-2.0 model, a performance comparison against TCRdist is included in Fig. 4 (for details on the implementation of the TCRdist method, refer to "Materials and methods"). This analysis aligns with the results from Fig. 3 demonstrating a consistent and highly significant ($p < 0.001$ for the α- and α + β-chain models, $p = 0.03$ for β-chain, bootstrap test with 1000 repetitions) superior performance of NetTCR-2.0 over TCRdist, and likewise showing that also for TCRdist is the signal in the CDR3β sequence lower compared with CDR3α when it comes to predicting the specificity toward the NLV peptide.

Next, we investigated the power of the developed model to identify the correct peptide target of a given TCR. Here, binding to the three peptides GIL, NLV, and GLC was predicted (using cross-validation) for each TCR positive to any of these three peptides. To deal with peptide-specific scoring biases, the raw prediction values were transformed into the percentile rank values as described in "Materials and methods" and the predicted target for each TCR was identified from the peptide with the lowest rank value. This analysis was performed for the three models trained on the CDR3α and CDR3β, CDR3α alone and CDR3β alone, and the performance for each peptide was reported

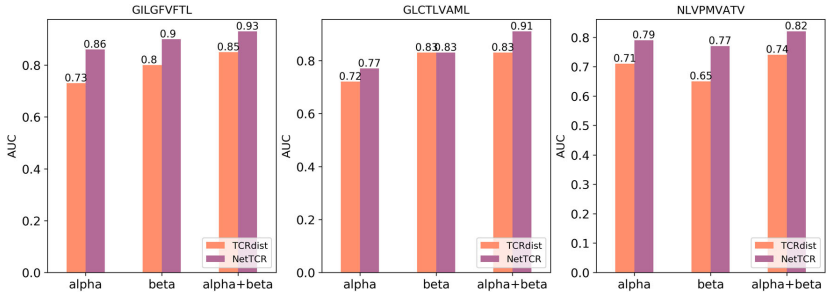


Fig. 4 Comparison between NetTCR and TCRdist. Performance is evaluated via cross-validation on the 95% partitioned data for the three most frequent peptides.

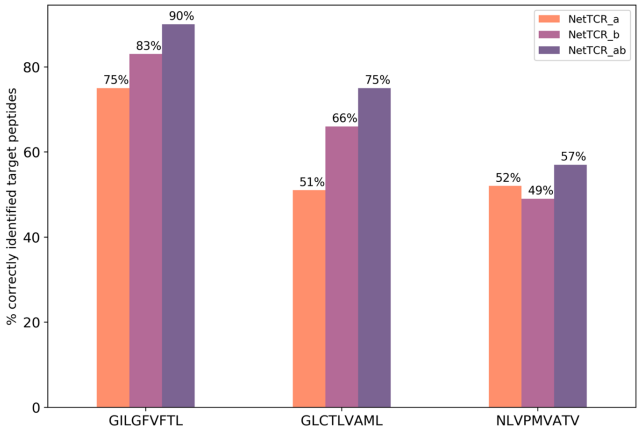


Fig. 5 Peptide-ranking analysis. Each TCR positive to GIL, GLC, or NLV peptide was paired to the other two peptides and a binding prediction was obtained. The percentages show, for each peptide and for each model, the proportion of TCRs for which the predicted lowest-ranking peptide matched with the "true" target peptide.

as the proportion of correctly identified targets (see Fig. 5). Here, all models performed better than random with the proportion of correct targets >33%. Further, the model trained on both CDR3α and CDR3β significantly outperformed both other models for all three peptides (p-value < 0.05 in all the cases, bootstrap test with 1000 repetitions); meanwhile, the choice of the best single-chain model was peptide dependent, with NetTCR_α outperforming NetTCR_β for the NLV peptide, in line with the result of Fig. 3. To further quantify to what extent the peptide sequence contributes to the model performance, models were trained on a data set where the TCR sequences were paired with a wrong peptide. Repeating the peptide-ranking analysis with these models demonstrated a highly reduced performance, exemplified with, for instance, the TCR_αβ for all TCRs predicting the optimal target as the GIL peptide (see Supplementary Fig. 5).

We propose that the improved predictive power of NetTCR over the sequence-based baseline model is driven by the representation of the TCRs in the max-pooled CNN layer of NetTCR. To elucidate this, the 160-dimensional representation max-pooled output (80 for each of the CDR3α and CDRb TCR sequences, respectively) from the NetTCR CNN layer of the CDR3α and CDR3β input was extracted for all TCRs specific to the GIL peptide. Likewise, a raw input representation of the TCR was constructed using a simple encoding scheme where each amino acid was represented by five features (normalized Van der Waals volume, hydrophobicity, number of hydrogen bond donors, number of hydrogen bond acceptors, and net charge). Next, the t-distributed stochastic neighbor embedding (t-SNE[24]) algorithm was used to visualize the relationship between these vectors in a 2-dimensional space (see Fig. 6). In contrast to the raw sequence representation (Fig. 6b), Fig. 6a shows the separation of the positive from the negative GIL TCRs with a clear positive TCR-enriched region in the upper-left part.

To further illustrate how the max-pooled feature space allows for separation of the positive from the negative GIL TCRs, Fig. 7 shows a hierarchically clustered heatmap of a random set of 50 positive and 50 negative GIL TCRs. This figure clearly illustrates the increased power for separation of the positive from the negative TCR when information from both CDR3α and CDR3β is included. Further comparing the results obtained using the paired-chain max-pooled representation (Fig. 7a) to the raw input space (Supplementary Fig. 6), confirmed the improved clustering potential of the max-pooled sequence representation. To further quantify the increased ability of classification in the CNN space, the positive and negative

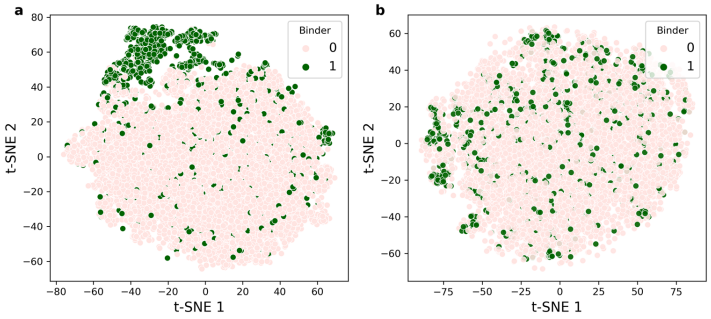


Fig. 6 t-SNE plot for the TCRs of the GIL peptide. a The output from the max-pooled CNN layer of NetTCR trained on the 90% partitioned data set was extracted for each TCR specific to the GIL peptide using cross-validation, resulting in a set of vectors, each of dimension 160. T-SNE was used to visualize this data set in two dimensions. **b** In the input space, the TCRs were encoded using a 5-feature physicochemical encoding and then flattened into a vector. The perplexity hyperparameter of the t-SNE algorithm was chosen to be 40 and the number of iterations was set to 1000. In the plot, positive TCRs are shown in green, and negative TCRs in pink.

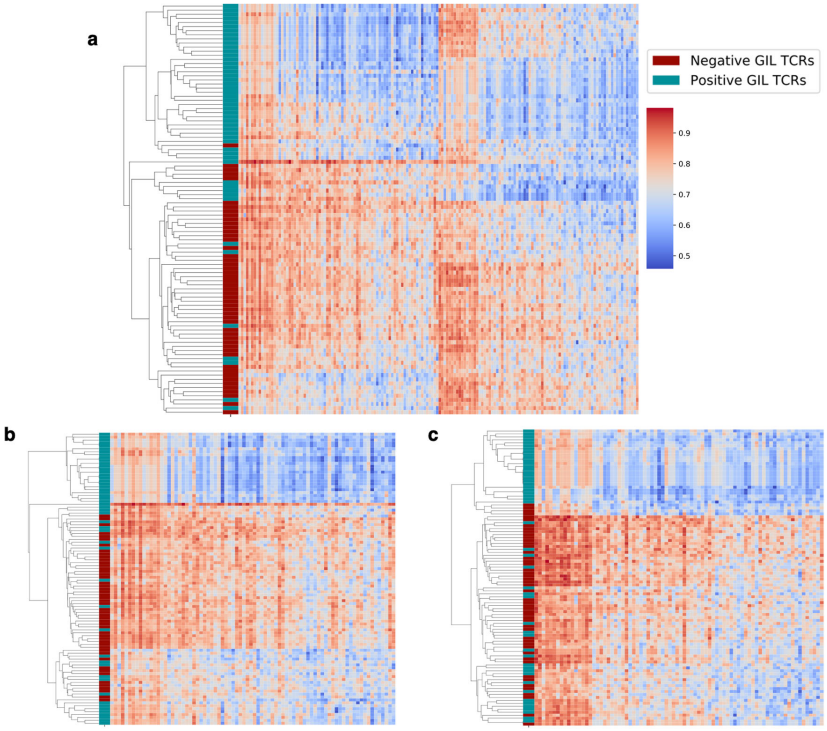


Fig. 7 Hierarchical-clustered heatmaps of 50 positive GIL TCRs and 50 negatives. The clustering was performed using both α- and β-sequences (**a**) or using single chains (α chain in **b**, β chain in **c**). Each row in the heatmap represents a TCR sequence in the max-pooled feature-space representation; the color bar on the side of each plot delineates whether the TCR is positive or negative. Cosine distance was used as a metric for clustering.

TCRs were clustered into two groups using the K-medoids algorithm. The two clusters were labeled as positive and negative by the majority vote of the TCRs falling in the cluster, and the clustering accuracy was evaluated using the Matthews correlation coefficient (MCC). The clustering was performed using both the max-pooled and the raw input representation of the TCRs, resulting in MCC values of $0.64 + /-0.09$ and $0.21 + /-0.14$ (standard-deviation values obtained using 1000 resamplings of TCR), confirming that the separation between positives and negatives is significantly more pronounced in the CNN space.

The NetTCR server. The presented NetTCR method is available as a web server at https://services.healthtech.dtu.dk/service.php?NetTCR-2.0. The server offers the possibility of predicting binding of the input TCRs with one or more peptides; predictions are made using the models trained on the 95% partitioned training data. Supplementary Fig. 7a, b serves as a guide to select thresholds for interpretation of prediction scores that the server outputs, and displays sensitivity–specificity curves of the method for the three individual peptides and the pooled data set with prediction values obtained as percentile rank scores using cross-validation. These figures demonstrate the very high specificity of the method with sensitivity values greater than 50% (and in most cases greater than 75%) and false-positive rates less than 2% in all cases using a percentile rank score threshold of 2%.

Real-life validation. As a real-life validation of the NetTCR-2.0 method, a performance comparison of the different models was conducted on a novel independent paired TCR data set generated specifically for this study. In short, the data were defined from T cells from four HLA-A*02:01-positive donors with pre-established responses to GILGFVFTL, NLVPMVATV, and GLCTLVAML sorted into a positive subset, containing TCRs responsive to one or more of the three peptides and a negative subset, containing TCRs negative to the three peptides. Here, the performance was estimated by predicting for each TCR binding to the three peptides and assigning a score corresponding to the lowest-predicted rank value. Next, performance values were calculated in terms of AUC, AUC0.1 (defined as the area under the ROC curve in the interval [0, 0.1]), and positive predictive value (PPV), calculated as the proportion of positive hits within the top 89 (the total number of positive TCR) predicted TCR. Here, the performance measures were used to quantify how this prediction score could be used to separate the positive and negative TCRs (see Fig. 8). Also in this benchmark, NetTCR_αβ significantly outperform all other methods ($p < 0.05$, bootstrap test with 10000 repetitions), with a performance gain of more than 10% in terms of PPV. Here the method demonstrate a very high specificity, identifying 79% of the positive TCR at a false-positive rate of 2% using a percentile rank threshold of 2% (Supplementary Fig. 7c).

Discussion

Identification of cognate targets of TCRs is a critical bottleneck of the development of T-cell therapeutics. Here, we have presented a study aiming to resolve this bottleneck, developing models capable of predicting TCR-pMHC interactions based on the amino acid sequences of the peptide and CDR3 region of the TCR chains. Several model architectures were investigated spanning from simple sequence-similarity models to more complex convolutional neural networks (CNN). The models were trained using cross-validation and validated using independent evaluation data carefully constructed using strict data-redundancy reduction rules. The overall best-performing model was found to be a 1D CNN. This model is a variant of the model proposed earlier by us for pan-specific prediction of kinase-specific phosphorylation[25]. This model significantly outperformed simpler sequence-based models implemented using the TCRMatch[22] and TCRdist[7] frameworks.

Two important issues related to the understanding of the TCR-binding characterization and prediction were addressed during the model development, namely the quality of the current data, and the impact of including paired CDRα and CDRβ information. First, models were developed using data available from the IEDB (similar results were obtained using CDR3β data from VDJdb) with CDR3β information available only. This data set was substantially larger compared with data with paired TCR-sequence information, and one would expect that models trained on such larger data sets should achieve overall higher performance values compared with models trained on the more reduced paired TCRα and TCRβ data sets. This was however not the case. Models constructed from data with CDR3β information from paired TCR data demonstrated significantly higher performance to similar models trained on the data with CDR3β information only. This result strongly suggests that the quality of the data with only CDR3β information is lower than that of the data with paired CDRs. Further, and in line with earlier work[7,8], the conclusions from the current study clearly supported the notion that both TCR chains contribute to the TCR specificity (and importantly, that their relative importance is pMHC specific), and that only by including this combined information can one achieve accurate TCR-specificity prediction.

In contrast to the models trained on the data with only CDR3β information, the model trained on the data with paired TCR information demonstrated a clear and statistically significant correlation of the peptide-specific performance to the number of different positive TCR available for a given peptide and suggested that ~150 unique TCRs are required to achieve an AUC > 0.75 for a given peptide. Currently, this criterion is only met for a very small set of MHC-peptide combinations placing great limitations on the applicability of the developed model, since it can only, given the current data, provide reliable predictions for three peptides. This limitation underlines the urgent need for the development and refinement of technologies for high-throughput paired sequencing of TCRs with known pMHC targets. The developed framework is trivially extendable and retrainable, as more data become available.

Investigating the TCR-specific performance of the model revealed a likewise high predictive power, with ~75% of predicted peptide targets (from the pool of three) being correct. Taken with some reservations, given the small peptide space covered, this high performance suggests that the model has the potential to resolve not only which TCRs are specific to a given peptide, but also which peptide is specific for a given TCR, pointing to important biomedical applications within T-cell therapy[26,27].

The power of the CNN model compared with the simpler sequence-based approaches lies in its ability to translate the variable length of the TCR sequences into an abstract feature space suitable for specificity classification. To illustrate this, a similarity analysis between TCRs specific to the GIL peptide was conducted in the CNN feature space compared with the original sequence space. This analysis confirmed the improved ability to perform classification in the CNN feature space and suggests that this representation potentially could be used as an alternative to the conventional autoencoding approaches for feature extraction and compression of biological data[28,29].

The current model only includes information from the two CDR3 regions of the TCR. Earlier work has demonstrated that also CDR1 and CDR2 carry information of potential importance for prediction of TCR specificity[7,8]. The modeling framework proposed here can readily be extended to include such information (as well as information related to HLA and V- and

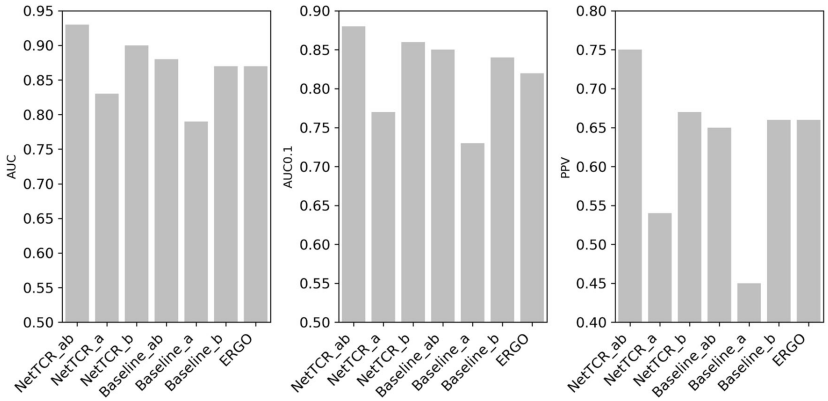


Fig. 8 Benchmark performance on in-house TCR data set. Methods included are NetTCR and baseline trained on paired CDR3α-CDR3β data (ab), CDR3α (a), CDR3β (b), and the LSTM-based ERGO trained on the VDJdb. Performance measures are (left) AUC, center (AUC 0.1), and right (PPV).

J- germline usage), and future work will tell if integrating this information can lead to an improved predictive power of the model proposed here. Also, the neural network architecture proposed here is relatively simple, consisting of one single max-pooled CNN layer. In this work, we did not perform an exhaustive performance comparison to other more complex models; however, our comparison to the ERGO model on the CDR3β-only data demonstrated comparable performance between the two modeling architectures, strongly suggesting that, at least for the current data and data volumes, a simple network architecture, like the one we have proposed here, is sufficient.

A critical issue for the development of machine learning models is the availability of accurate negative data. Often, not simply more but rather more accurate data are needed. Earlier works have proposed to resolve this issue by either mispairing the positive data, or by including data from healthy controls as negatives[14,15]. Both approaches share potential pitfalls in that the proposed negatives either share a compositional bias (imposed by the fact that they are positive to one or more of the other peptides in the data) or that the TCRs are falsely labeled as negative (imposed by the fact that TCRs in healthy controls are likely positive to the dominant peptides in the positive data set). Here, we have therefore taken a different approach, benefitting from the study published by 10X Genomics, and complemented the mis-paired artificial negative data with TCRs explicitly found not to be positive to any of the peptides in the training data. While this proved a highly useful approach, the 10X Genomics MHC-feature barcode platform is still in development, and the negative data defined here are hence likely not fully accurate. Given this, we suggest that substantial further work is needed to assess how to best define a proper TCR-negative data set.

The high performance of the developed NetTCR-2.0 model was validated on an in-house data set of paired TCR data with qualitative-interaction measurements to a set of 3 HLA-A*02:01 peptides. Here, a predictive positive value of ~75% was observed, greatly surpassing the performance of both the baseline and ERGO models. This result confirmed that the development of accurate prediction models for TCR specificity is contingent on the availability of paired (and accurate) α- and β-sequence data and suggests that a predictive power can be achieved to a degree where the tool can have actual biomedical applications.

Finally, in this work, we have used a rather simple definition of TCR similarity based on the relative Levenshtein distance when defining data redundancy. This distance has obvious shortcomings when comparing the similarity between pairs of TCR of very different lengths—i.e., a similarity score of 0.9 corresponds to both one mutation/edit when comparing two TCRs of length 10 and to 4 mutations/edits if the TCRs are of length 36. Given the relatively limited length variation of the CDR3 sequences included in the current work (90% of the paired CDR3α and CDR3β sequences from the paired data set have a length in the range of 9–13 amino acids), this shortcoming does not have large impacts for the current work. However, it will be essential to consider alternative and less length-biased approaches, such as, for instance, the kernel similarity method underlying TCRmatch[22], if the work is extended to cover full-length TCRs and/or include the complete set of CDR sequences.

In conclusion, we have successfully trained a model to predict interactions between TCRs and their cognate, HLA-A*02:01-restricted peptide target. Our results indicate that accurate prediction is feasible only by training on data of paired TCRα- and β-chains. Due to the small number of training peptides, the model can at present only be applied to the limited set of peptides included in the training data. However, as more data become available, we expect the predictive power of the model to increase and allow for accurate predictions also for uncharacterized peptides, as has been observed earlier for the pan-specific prediction models of peptide-MHC interactions[30]. Finally, the presented model framework is highly flexible and allows for the straightforward integration of the MHC molecule or TCRα chain in the future when data become available, to train a truly global prediction method.

Materials and methods

Training data

CDRβ data. The initial set of CDR3β sequences binding to epitopes presented by HLA-A*02:01 with corresponding epitopes was collected from the Immune Epitope Database (IEDB) on January 29th, 2020. The original IEDB data set consisted of 25,300 data points with 21,855 unique CDR3β sequences and 675 unique peptides, covering both class-I and -II binders. Cross-reactive TCRs were excluded. Quality assessment and uniform CDR3β-sequence frame were ensured by applying a k-mer-based scoring method using a profile hidden Markov model (pHMM) to the data (see Supplementary Note 1 details). Following quality assurance, the IEDB data set specific for HLA-A*02:01 and peptides of length 9 consisted of 10,987 unique CDR3β sequences and 168 peptides.

Nonbinding peptide-CDR3β pairs were derived from 10X Genomics Chromium Single Cell Immune Profiling of four donors. All T cells in this assay had been exposed to all tested pMHC multimers[31]. Each entry of the data set includes a unique molecular identifier (UMI) and counts of a given TCR to all peptides in the assay. From this data set, an initial negative data set was constructed from the HLA-A*02:01-restricted peptides filtered to only include TCR-peptide pairs with UMI counts $<= 10$. This data set comprised 1,325,949 distinct peptide-CDR3β pairs with 69,847 unique CDR3β sequences and 19 different peptides of which seven were shared with the IEDB peptides.

Positive and negative training data points were reduced to peptide-TCR pairs with CDR3β lengths within the range of 8–18 amino acids, and peptides of length equal to nine amino acids shared between the two data sets (7 peptides). The final data set representing seven epitopes characterized with both positive and negative TCR data consists of a positive set of 9204 unique CDR3β-peptide pairs and a negative data pool of 387,598 data points.

Paired CDR data. Positive data points were taken from IEDB and VDJdb. The databases were downloaded on August 26th, 2020 and August 5th, 2020, respectively. Restricting to data with both CDR3α and CDR3β chains available, a length range of 8–18 and reported to bind peptides of length 9, 3859 unique binding pairs were identified from IEDB and 2843 from VDJdb. These provided 4598 unique CDR3α-/β-peptide interactions with 276 different peptides specific to allele HLA-A*02:01.

Negatives were derived from 10X. Using the same restrictions as for the positives (CDR3 length between 8 and 18 AAs, peptide length 9, and peptides specific for HLA-A*02:01), 627,323 unique data points with 0 UMI counts to all the tested peptides were identified. These contained 33,017 unique TCRs tested against a set of 19 different peptides. In total, 17 of these overlapped with the peptides in the positive data set.

External evaluation data

MIRA. Positive data points for external evaluation were derived from the MIRA set[32]. It entailed 376 CDR3β-peptide pairs associated with HLA-A*02:01. Negative samples were taken from an excluded subset of the 10X negative set (see above).

Validation data. Healthy donor material was collected under approval by the local Scientific Ethics Committee and written informed consent was obtained according to the Declaration of Helsinki. Peripheral blood mononuclear cells (PBMCs) from healthy donors were isolated from whole blood by density centrifugation on Lymphoprep (Axis-Shield PoC) and cryopreserved at −150 °C in FCS (FCS; Gibco) +10% DMSO.

The three peptides, GILGFVFTL, NLVPMVATV, and GLCTLVAML, were purchased from Pepscan (Pepscan Presto) and dissolved to 10 mM in DMSO. UV-sensitive ligands were synthesized as previously described[33]. In brief, recombinant HLA-A*02:01 heavy chains and human β_2 microglobulin light chain were produced in Escherichia coli. HLA heavy and light chains were refolded with UV-sensitive ligands. Specific peptide-MHC complexes were generated by UV-mediated peptide exchange[33] and MHC tetramers were assembled on PE-conjugated streptavidin (BioLegend, Nordic Biosite, Denmark) as previously described[34].

Cryopreserved PBMCs from four HLA-A*02:01-positive donors were thawed and washed in RPMI + 10% FCS. The presence of T cells binding to GILGFVFTL, NLVPMVATV, and GLCTLVAML was preestablished using DNA barcode-labeled MHC multimers as described in Bentzen et al.[9]. In total, $3 \times 10^6 - 6 \times 10^6$ cells from each donor were washed in cytometry buffer (PBS + 2% FCS) and incubated, 15 min, 37 °C, with a pool containing all three MHC multimers in a total volume of 80 µL (final concentration of each distinct pMHC, 23 nM). Next, a 5x antibody mix composed of CD8-BV480 (clone RPA-T8, BD 566121) (final dilution 1/50), dump-channel antibodies: CD4-FITC (BD 345768) (final dilution 1/80), CD14-FITC (BD 345784) (final dilution 1/32), CD19-FITC (BD 345776) (final dilution 1/16), CD40-FITC (Serotech MCA1590F) (final dilution 1/40), CD16-FITC (BD 335035) (final dilution 1/64), and a dead-cell marker (LIVE/DEAD Fixable Near-IR; Invitrogen L10119) (final dilution 1/1000) was added and incubated for 30 min at 4 °C. Cells were washed twice in cytometry buffer before proceeding directly to sorting.

Cells were sorted on a FACSMelody Cell Sorter (Becton Dickinson) into tubes containing 150 µl of PBS + 0.5% BSA (tubes were presaturated with PBS + 2% BSA). Using BD FACSChorus Software, we gated on single, live CD8-positive and "dump" (CD4, 14, 16, 19, and 40) negative lymphocytes. Within this population, we sorted all multimer-(PE) positive cells from all donors into one tube and a proportion of multimer negative/CD8 positive from all donors into another tube. The sorted cells were centrifuged for 10 min at 390 g and the buffer was removed. An overview of samples and gating strategy is included in Supplementary Table 1 and Supplementary Fig. 8.

VDJ sequences from the CD8 T cells were obtained through the 10x Genomics pipeline using Chromium Next GEM Single Cell 5′ Reagent Kits v2 (Dual Index) according to the manufacturer's instructions (10x Genomics, USA). Up to 17,000 cells of the multimer-positive or the multimer-negative CD8 T cells were loaded onto each of their separate lane, to yield a maximum of 10,000 cells with an intermediate/high doublet rate. TCRs were sequenced on a MiSeq as recommended by Illumina.

The single-cell data were processed via the 10x Genomics software Cell Ranger v5.0.1, using cellranger mkfastq and cellranger vdj, to extract V(D)J gene annotations and CDR3 sequences for each T cell. The GRCh38/Ensembl reference genome v4.0.0 for mapping V(D)J genes was downloaded from 10x Genomics. The pool of all multimer-positive cells and the pool of multimer-negative cells yielded 1091 and 12,801 mapped and annotated T cells. Of these sets, 520 and 3074 cells, respectively, met the criteria of having both an α- and β-chain with unambiguous annotations, meaning that each T cell should only have one α-chain and one β-chain annotation. Reducing the sets to contain only unique pairs of CDR3 α/β and removing the TCRs already present in the training set, resulted in 89 multimer-positive pairs and 1694 multimer-negative pairs.

Data preparation. Figure 9 gives a schematic overview of how the data-redundancy and data partitioning procedure was implemented in the current work. The sections below describe the details of each of the outlined steps.

Similarity scoring. A critical component of data redundancy is related to the metric chosen to define the similarity between two points. Here, the Levenshtein similarity was used as a measure of the similarity between CDR3 sequences. The Levenshtein similarity is based on the Levenshtein distance. The Levenshtein distance is a similarity measure between words. Given two strings, the distance describes the number of modifications needed to transform one word into another. The possible changes are insertion, deletion, and replacement. Each of these three operations adds one to the distance. The Levenshtein similarity score is given by the relation

$$Sim_{Lev} = \frac{max(|u|, |v|) - Distance_{Lev}(u, v)}{max(|u|, |v|)}, \quad (1)$$

where u and v represent two CDR3 sequences, and $|\cdot|$ defines their length.

Redundancy reduction. Peptide-specific redundancies regarding CDR3 sequences were removed using the Hobohm 1 algorithm[35]. The positive and negative data specific for each peptide were each first sorted by CDR3 length in descending order. Next, the sorted negative data were appended to the sorted positive data. Sequences were then iteratively sorted into non-redundant and redundant stacks based on a given similarity threshold, hereafter referred to as redundancy threshold. The algorithm starts by assigning the first sequence to the nonredundant list. It then iterates through the peptide-specific CDR3 sequences and assesses whether a sequence's similarity to the list of nonredundant sequences is above the redundancy threshold or not. Similarities above the threshold lead to the examined sequence being assigned to the redundant list.

Data partitioning. Partitioning was performed using single-linkage clustering of the redundancy-reduced positive training data. First, the Levenshtein similarity scores between all CDR3 sequences are binarized based on a given threshold, referred to as the data-partitioning threshold. In the case of paired-chain data, TCR similarity is defined as the average α and β Levenshtein similarity. Next, single-linkage clustering was performed on this binary matrix, and the connected components of this graph were sorted by size into a list and iteratively assigned partitions 1–5. The selected similarity threshold thus presents an upper limit of similarities between different partitions.

Next, negative CDR3 data were added to each partition. For each peptide in each partition, 5 times the number of positive CDR3 were added from the negative data. Negatives were gradually added under the condition that their similarity to all TCRs in the other partitions was lower than the given partitioning threshold. In addition, negative examples were generated by mismatching the positive data, i.e., combining a TCR sequence with a peptide different from its cognate target. Each positive TCR was paired with 5 peptides, randomly sampled from the list of unique peptides in the dataset. These added negatives were used during the training but were not included when evaluating the model performance.

Separating external Evaluation Data from the Training Data. The evaluation data sets for the CDR3β model were separated from the training data by a given Levenshtein similarity threshold, meaning that the data points with similarities to the training data above this threshold were removed. Negatives reserved for external evaluation were reduced to CDR3β sequences with similarities below the given threshold to the training data. Subsequently, five times the number of positives per peptide were randomly selected from the remaining negatives.

Paired chain data preparation pipeline. Positive and negative data from IEDB, VDJdb, and 10X were prepared and cleaned as described in the training data section. Positives and negatives were then reduced to data points containing their shared set of peptides. This is represented by 18 different peptides and resulted in 2886 unique positive interactions and 594,306 unique negative data points. Positive data were subsequently partitioned into 5 partitions with a similarity threshold based on their average chain similarities. Negatives were then added to the partitioned positives as 5 times the number of positives per peptide and partition, upholding the similarity restraint of the partitioning. Further were mismatched negatives added as described above.

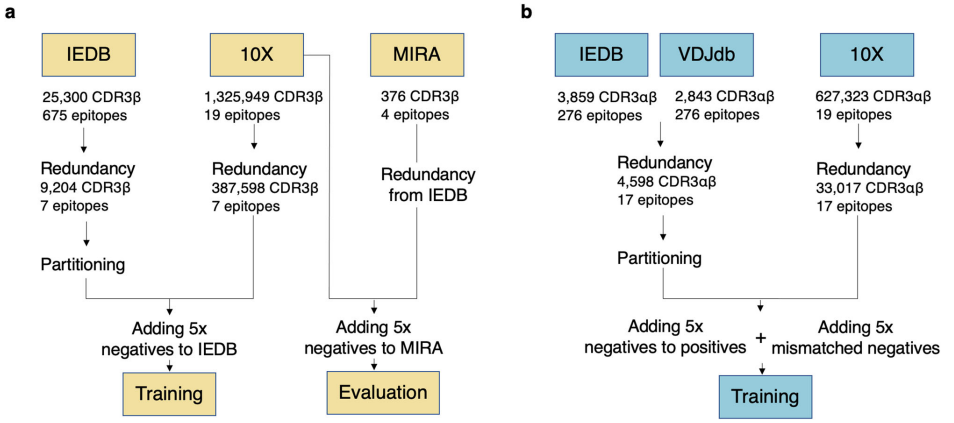


Fig. 9 Data-partitioning pipeline schematics. a Data-preparation pipeline for the β-chain data; **b** pipeline for the paired-chain data. The positive and negative data sets were each redundancy-reduced with the Hobohm 1 algorithm, according to a Levenshtein similarity threshold. The redundancy-reduced set of positives was partitioned into five groups using a single-linkage clustering algorithm. Negative data were subsequently added to each partition: for each peptide, 5 times the number of positives was randomly selected from the pool of nonredundant negative data. In **a**, to ensure that the MIRA external evaluation data did not share similarity with the training set, positive points from the MIRA set with a Levenshtein similarity above a certain threshold were removed. Each step of the pipeline is described in detail in the text.

Baseline model. A baseline model was designed to establish the predictive power of simple similarity-based methods. The similarity-scoring approach used in the baseline model was the kernel-scoring method introduced by Shen et al.[36] with default parameters, as described earlier in the MAIT Match[19] and TCRMatch[22] methods. In the model, the prediction score for a given TCR is calculated as the highest score obtained when scoring the CDR3β against a database of positive CDR3βs. In 5-fold cross-validation, each of the 5 partitions, in turn, represents a test set, and the positive elements in the remaining 4 partitions define the database. For external evaluation, all positive elements in the training data set define the database. For analysis of paired α and β TCR sequences, the similarity score was calculated as the highest average of the individual α and β CDR3-sequence scores for each TCR.

TCRdist model. The TCRdist model was implemented identically to the baseline model only using the distance metric proposed in the TCRdist publication[7]. That is, the prediction score for a given TCR is calculated as 1—the closest distance obtained when scoring the TCR against a database of positive TCRs for the given peptide (defined in a cross-validated manner).

Neural networks

The NetTCR model. A 1-dimensional CNN model, similar to the one proposed by Jurtz et al.[15], was implemented to predict whether or not a given TCR can bind to a specific peptide. The neural network takes the peptide, the CDR3α, and/or CDR3β regions of the TCR amino acid sequences as inputs. The CDR sequences were zero-padded to a maximum length of 30. The amino acids were encoded using the BLOSUM50 matrix[37]. That is, each amino acid is represented as the score for substituting the amino acid with all the 20 amino acids. Hence, the BLOSUM encoding scheme maps a sequence of length l into an array of dimension l x 20. The peptide and the CDR3 sequences are processed separately by a 1D convolutional layer with channels corresponding to the given sequence encoding. On each sequence (peptide, CDR3(s)), 16 convolutional filters with kernel size {1, 3, 5, 7, 9} process the input (80 filters per sequence). The kernel weights were initialized with the Glorot normal initializer[38]. For each kernel size, the convolutional output was max-pooled and the resulting feature vectors concatenated in a single vector with 240 entries (80 for each input sequence) representing the convoluted peptide and CDR3 sequences. This vector was then fed into a dense layer of 32 hidden neurons; the output consists of one single neuron, giving the probability of a peptide-TCR pair to bind. The activation function used through the network was the sigmoid function. A schematic representation of the CNN model is given in Fig. 10.

Model training. Models were trained using nested 5-fold cross-validation (CV) for 300 epochs with early stopping and patience of 50 epochs. The weights were updated using the Adam optimizer with a learning rate of 0.001. The batch size was 128 and the loss function was binary cross-entropy.

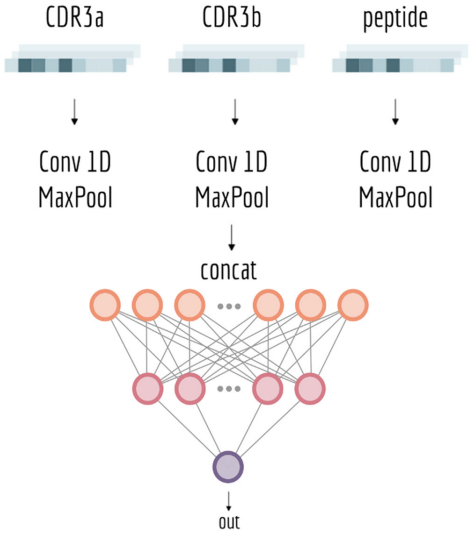


Fig. 10 Setup of NetTCR model. The CDR3 and peptide sequences are encoded using the BLOSUM50 matrix. The encoded sequences are passed independently through a 1D convolutional layer and a max-pooling layer. The convolutional filter size is set to {1, 3, 5, 7, 9}, and for each filter size, 16 filters are used. The extracted features are then concatenated and fed into a dense layer with 32 hidden units. The output of the network consists of a single neuron, giving the binding probability.

Performance evaluation. In cross-validation, the performance was evaluated from the concatenated test sets either globally over the entire data set, or in a per-peptide manner. Likewise was the performance on the independent evaluation reported either globally over the entire data set, or in a per-peptide manner. To normalize the prediction scores across peptides, the raw prediction values were transformed into the percentile rank values. Percentile rank scores were estimated from a set of 10,000 natural TCRs, extracted from the 10X data set with no overlap with the training set. The percentile rank score of a given peptide-TCR pair was then calculated by comparing the prediction score with the distribution of prediction scores for the particular peptide.

Reporting summary. Further information on research design is available in the Nature Research Reporting Summary linked to this article.

Data availability

All data and data partitions used for NetTCR-2.0 training and evaluation are available at https://github.com/mnielLab/NetTCR-2.0.

Code availability

The NetTCR-2.0 code is available at https://github.com/mnielLab/NetTCR-2.0. The NetTCR-2.0 prediction model is available as a web-server tool at https://services.healthtech.dtu.dk/service.php?NetTCR-2.0.

Received: 8 April 2021; Accepted: 27 August 2021;
Published online: 10 September 2021

References

1. La Gruta, N. L., Gras, S., Daley, S. R., Thomas, P. G. & Rossjohn, J. Understanding the drivers of MHC restriction of T cell receptors. *Nat. Rev. Immunol.* **18**, 467–478 (2018).
2. Feng, D., Bond, C. J., Ely, L. K., Maynard, J. & Garcia, K. C. Structural evidence for a germline-encoded T cell receptor-major histocompatibility complex interaction "codon". *Nat. Immunol.* **8**, 975–983 (2007).
3. Rossjohn, J. et al. T cell antigen receptor recognition of antigen-presenting molecules. *Annu. Rev. Immunol.* **33**, 169–200 (2015).
4. Vita, R. et al. The immune epitope database (IEDB): 2018 update. *Nucleic Acids Res.* **47**, D339–D343 (2019).
5. Bagaev, D. V. et al. VDJdb in 2019: database extension, new analysis infrastructure and a T-cell receptor motif compendium. *Nucleic Acids Res.* **48**, D1057–D1062 (2020).
6. Tickotsky, N., Sagiv, T., Prilusky, J., Shifrut, E. & Friedman, N. McPAS-TCR: a manually curated catalogue of pathology-associated T cell receptor sequences. *Bioinformatics* **33**, 2924–2929 (2017).
7. Dash, P. et al. Quantifiable predictive features define epitope-specific T cell receptor repertoires. *Nature* **547**, 89–93 (2017).
8. Lanzarotti, E., Marcatili, P. & Nielsen, M. T-cell receptor cognate target prediction based on paired α and β chain sequence and structural CDR loop similarities. *Front. Immunol.* **10**, 2080 (2019).
9. Bentzen, A. K. et al. Large-scale detection of antigen-specific T cells using peptide-MHC-I multimers labeled with DNA barcodes. *Nat. Biotechnol.* **34**, 1037–1045 (2016).
10. Purcell, A. W., Ramarathinam, S. H. & Ternette, N. Mass spectrometry-based identification of MHC-bound peptides for immunopeptidomics. *Nat. Protoc.* **14**, 1687–1707 (2019).
11. Peters, B., Nielsen, M. & Sette, A. T cell epitope predictions. *Annu. Rev. Immunol.* **38**, 123–145 (2020).
12. Nielsen, M., Andreatta, M., Peters, B. & Buus, S. Immunoinformatics: predicting peptide–MHC binding. *Annu. Rev. Biomed. Data Sci.* **3**, 191–215 (2020).
13. Tong, Y. et al. SETE: Sequence-based ensemble learning approach for TCR epitope binding prediction. *Comput. Biol. Chem.* **87**, 107281 (2020).
14. Springer, I., Besser, H., Tickotsky-Moskovitz, N., Dvorkin, S. & Louzoun, Y. Prediction of specific TCR-peptide binding from large dictionaries of TCR-peptide pairs. *Front. Immunol.* **11**, 1803 (2020).
15. Jurtz, V. I. et al. NetTCR: sequence-based prediction of TCR binding to peptide-MHC complexes using convolutional neural networks. *BioRxiv* https://doi.org/10.1101/433706 (2018).
16. Moris, P. et al. Treating biomolecular interaction as an image classification problem – a case study on T-cell receptor-epitope recognition prediction. *BioRxiv* https://doi.org/10.1101/2019.12.18.880146 (2019).
17. Jokinen, E., Heinonen, M., Huuhtanen, J., Mustjoki, S. & Lähdesmäki, H. TCRGP: Determining epitope specificity of T cell receptors. *BioRxiv* https://doi.org/10.1101/542332 (2019).
18. Fischer, D. S., Wu, Y., Schubert, B. & Theis, F. J. Predicting antigen specificity of single T cells based on TCR CDR3 regions. *Mol. Syst. Biol.* **16**, e9416 (2020).
19. Wong, E. B. et al. TRAV1-2 + CD8 + T-cells including oligoclonal expansions of MAIT cells are enriched in the airways in human tuberculosis. *Commun. Biol.* **2**, 203 (2019).
20. Gielis, S. et al. Detection of enriched T cell epitope specificity in full T cell receptor sequence repertoires. *Front. Immunol.* **10**, 2820 (2019).
21. De Neuter, N. et al. On the feasibility of mining CD8 + T cell receptor patterns underlying immunogenic peptide recognition. *Immunogenetics* **70**, 159–168 (2018).
22. Chronister, W. D. et al. TCRMatch: Predicting T-cell receptor specificity based on sequence similarity to previously characterized receptors. *BioRxiv* https://doi.org/10.1101/2020.12.11.418426 (2020).
23. Gielis, S. et al. TCRex: a webtool for the prediction of T-cell receptor sequence epitope specificity. *BioRxiv* https://doi.org/10.1101/373472 (2018).
24. Van der Maaten, L. & Hinton, G. Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**, 2579–2605 (2008).
25. Fenoy, E., Izarzugaza, J. M. G., Jurtz, V., Brunak, S. & Nielsen, M. A generic deep convolutional neural network framework for prediction of receptor-ligand interactions-NetPhosPan: application to kinase phosphorylation prediction. *Bioinformatics* **35**, 1098–1107 (2019).
26. Yee, C. Adoptive T cell therapy: addressing challenges in cancer immunotherapy. *J. Transl. Med.* **3**, 17 (2005).
27. Jones, H. F., Molvi, Z., Klatt, M. G., Dao, T. & Scheinberg, D. A. Empirical and rational design of T cell receptor-based immunotherapies. *Front. Immunol.* **11**, 585385 (2020).
28. Tang, B., Pan, Z., Yin, K. & Khateeb, A. Recent advances of deep learning in bioinformatics and computational biology. *Front. Genet.* **10**, 214 (2019).
29. Karim, M. R. et al. Deep learning-based clustering approaches for bioinformatics. *Brief. Bioinforma.* **22**, 393–415 (2021).
30. Hoof, I. et al. NetMHCpan, a method for MHC class I binding prediction beyond humans. *Immunogenetics* **61**, 1–13 (2009).
31. 10X Genomics. A New Way of Exploring Immunity - Linking Highly Multiplexed Antigen Recognition to Immune Repertoire and Phenotype | Technology Networks A New Way of Exploring Immunity - Linking Highly Multiplexed Antigen Recognition to Immune Repertoire and Phenotype (2019). https://www.technologynetworks.com/immunology/application-notes/a-new-way-of-exploring-immunity-linking-highly-multiplexed-antigen-recognition-to-immune-repertoire-332554.
32. Klinger, M. et al. Multiplex identification of antigen-specific t cell receptors using a combination of immune assays and immune receptor sequencing. *PLoS ONE* **10**, e0141561 (2015).
33. Rodenko, B. et al. Generation of peptide-MHC class I complexes through UV-mediated ligand exchange. *Nat. Protoc.* **1**, 1120–1132 (2006).
34. Hadrup, S. R. et al. Parallel detection of antigen-specific T-cell responses by multidimensional encoding of MHC multimers. *Nat. Methods* **6**, 520–526 (2009).
35. Hobohm, U., Scharf, M., Schneider, R. O. & Sander, C. Selection of representative protein data sets. *Protein Sci.* **1**, 409–417 (1992).
36. Shen, W.-J., Wong, H.-S., Xiao, Q.-W., Guo, X. & Smale, S. Towards a mathematical foundation of immunology and amino acid chains. *arXiv* arXiv:1205.6031 (2012).
37. Henikoff, S. & Henikoff, J. G. Amino acid substitution matrices from protein blocks. *Proc. Natl Acad. Sci. USA* **89**, 10915–10919 (1992).
38. Glorot, X. & Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. *PMLR* **9**, 249–256 (2010).

Acknowledgements

We would like to thank DTU Multi-Assay Core (DMAC) for sequencing the set of novel paired-chain TCRs. This research was funded in part through the Independent research fund Denmark (DFF-7014-00055 to M.N), the Federal funds from the National Institute of Allergy and Infectious Diseases, National Institutes of Health, Department of Health and Human Services (under Contract No. HHSN272201200010CERC to M.N), StG NextDART (677268 to S.R.H.), and the Lundbeck Foundation Experiment (R324-2019-1671 to A.K.B.).

Author contributions

M.N., L.E.J., A.M., V.S. and V.J. designed the study. A.M. and V.S. conducted the majority of the experiments. V.S. and L.E.J. contributed to the data cleaning and partitioning pipeline. A.K.B., H.R.P. and S.R.H. performed the T.C.R. sequencing and analysis. W.D.C., A.C. and B.J. contributed to the data collection and method-performance comparisons. The paper was written by A.M., V.S., L.E.J. and M.N. with contributions from all authors. All authors have read and approved the final version of the paper.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s42003-021-02610-3.

Correspondence and requests for materials should be addressed to Morten Nielsen.

Peer review information *Communications Biology* thanks the anonymous reviewers for their contribution to the peer review of this work. Primary Handling Editors: Eirini Marouli and Luke R. Grinham. Peer reviewer reports are available.

Reprints and permission information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

CHAPTER 5

Tips and Tricks to Build a TCR Specificity Prediction Model

This chapter presents the work on NetTCR-2.1, the natural extension of the models proposed in Chapter 4, NetTCR-2.0.

Rather than benchmarking other publicly available methods, the work aimed to set some standard rules that would guide researchers in developing new TCR-peptide interaction prediction models.

With the work on NetTCR-2.0, we demonstrated that the inclusion of both α and β CDR3 loops led to improved performance compared to single-chain models. We set this as a starting point for NetTCR-2.1 and expanded the model to also include CDR1 and CDR2 sequences. Further, we investigated if a pan-specific or a peptide-specific approach would better model the interaction, given the currently available data.

Lastly, we focused on defining guidelines for building an optimal training dataset. Two main aspects were taken into consideration, namely data redundancy, and negative data generation. First, we showed that it is of paramount importance to properly handle sequence redundancy in the data, to avoid performance overestimation due to data leakage between training and test set. Secondly, we analyzed different approaches for generating artificial negative

data, as most of the publicly available datasets only report positive binding events.

Together with NetTCR-2.1, we proposed TCRbase, a similarity-based model to predict TCR-peptide binding. We showed that this model achieves satisfying performance, despite being simple and less complex than a neural network-based model.

NetTCR-2.1: Lessons and guidance on how to develop models for TCR specificity predictions

Alessandro Montemurro[1], Leon Eyrich Jessen[1,*], and Morten Nielsen[1,2,*,$]

Affiliations

1. Department of Health Technology, Section for Bioinformatics, Technical University of Denmark, DTU, 2800 Kgs. Lyngby, Denmark
2. Instituto de Investigaciones Biotecnológicas, Universidad Nacional de San Martín, Buenos Aires, Argentina

* These authors share last authorship

$ Corresponding author: morni@dtu.dk

Abstract

T cell receptors (TCR) define the specificity of T cells and are responsible for their interaction with peptide antigen targets presented in complex with major histocompatibility complex (MHC) molecules. Understanding the rules underlying this interaction hence forms the foundation for our understanding of basic adaptive immunology. Over the last decade, efforts have been dedicated to developing assays for high throughput identification of peptide-specific TCRs. Based on such data, several computational methods have been proposed for predicting the TCR-pMHC interaction. The general conclusion from these studies is that the prediction of TCR interactions with MHC-peptide complexes remains highly challenging. Several reasons form the basis for this including scarcity and quality of data, and ill-defined modeling objectives imposed by the high redundancy of the available data.

In this work, we propose a framework for dealing with this redundancy, allowing us to address essential questions related to the modeling of TCR specificity including the use of peptide- versus pan-specific models, how to best define negative data, and the performance impact of integrating of CDR1 and 2 loops. Further, we illustrate how and why it is strongly recommended to include simple similarity-based modeling approaches when validating an improved predictive power of machine learning models, and that such validation should include a performance evaluation as a function of “distance” to the training data, to quantify the potential for generalization of the proposed model. The conclusion of the work is that, given current data, TCR specificity is best modeled using peptide-specific approaches, integrating information from all 6 CDR loops, and with negative data constructed from a combination of true and mislabeled negatives. Comparing such machine learning models to similarity-based approaches demonstrated an increased performance gain of the former as the “distance” to the training data was increased; thus demonstrating an improved generalization ability of the machine learning-based approaches.

We believe these results demonstrate that the outlined modeling framework and proposed evaluation strategy form a solid basis for investigating the modeling of TCR specificities and that adhering to such a framework will allow for faster progress within the field.

The final devolved model, NetTCR-2.1, is available at https://services.healthtech.dtu.dk/service.php?NetTCR-2.1.

Introduction

T cells form the cornerstone of the adaptive immune system orchestrating and executing attacks on pathogens and pathogen-infected/malfunctioning cells (1,2). T cell interacts with pathogen or self-aberrant derived peptides (p) presented on the cell surface by MHC (Major Histocompatibility Complex) molecules. This interaction is mediated via the trans-membrane T cell receptor (TCR). Not all MHC-presented peptides are able to form an interaction with TCR, and vice versa individual TCRs form a highly specific interaction only with a limited repertoire of pMHC complexes. Understanding the rules underlying this interaction thus holds promise for furthering our understanding of T cell immunogenicity, T cell tolerization, and T cell cross-reactivity.

The TCR is a heterodimeric protein, most often formed by an α- and β-chain. The interaction of TCRs with the cognate pMHC target is primarily defined by 6 loops, 3 on each chain denoted CDR1-3 (complementarity determining regions 1-3). Of these loops, CDR3 interacts primarily with the peptide, and CDR1 and CDR2 primarily with the α loops of the MHC complex (1,2). The diversity of TCRs is focused mainly on the CDR3s, a region defined by the genomic recombination of the variable, diversity (for CDR3β only), and joining (VDJ) TCR genes.

Large efforts have been dedicated over the years to develop assays for high throughput identification of peptide-specific TCRs. Most of these techniques and assays have focused on sequencing the CDR3β segment, applying cell sorting followed by bulk repertoire sequencing (3,4). While such approaches are highly cost-effective, they suffer from a relatively high proportion of wrongly identified TCR (present due to carryover in the sorting step). However and more importantly, they suffer from limited information capture and they only describe the CDR3β part of the TCR interaction. We and others have demonstrated the important shortcoming of this limited view on the TCR-pMHC interaction and demonstrated how the information on the specificity of the TCR toward its cognate pMHC target is carried by CDR3 of both α- and β-chains (5,6). A solution to this is to apply single=cell sequences enabling the identification of paired α- and β-chains.

A large plethora of methods has been published within the field of prediction of TCR-pMHC interactions. Given this limited amount of paired TCR α- and β data available, the majority of

these have focused on CDR3β information only (7,8,9). Recently however, models have merged benefitting from the growing volume of paired TCR data allowing for boosting performance by integrating information from both chains (10,11,6).

Data on TCR specificity is available in several public databases including VDJdb (12), IEDB (13), McPAS-TCR (14), and TBAdb (15). These databases are highly biased towards data on positive TCR-pMHC interactions. Furthermore, TCR data sets are often highly redundant and composed of many highly similar sequences. Both of these properties pose a challenge when it comes to developing and performance evaluating machine learning (ML) models. In terms of negative data, different approaches have been suggested including mispaired negatives and/or data from healthy controls (7,16). Most works within TCR specificity have paid very limited attention to data redundancy and sequence similarity, meaning that often the issue has been accessed by only removing identical data points (17,18). This is clearly an oversimplification, and we have earlier proposed an approach based on the Levenshtein similarities, Hobohm-based redundancy reduction, and single-linkage clustering, and have demonstrated how such careful redundancy considerations can aid the development of models with improved power for generalization (11).

Another critical aspect of TCR-pMHC interaction prediction is the choice between peptide- and pan-specific models. Peptide-specific models are, as the name indicates, models trained specifically for individual peptides, whereas pan-specific models are models encompassing all peptides in the given training data into a single model. Ideally one would seek to develop pan-specific models since these in principle would allow for ab-initio predictions for novel peptides not included in the training data by extrapolation from information and patterns learned across the different peptides. However such extrapolations might only be possible when the coverage of the peptide space in the training data reaches a certain limit. Anecdotally, this is in line with what was observed for the modeling of HLA-peptide binding. Here, HLA-specific models were found to outperform the early pan-specific models and only when the HLA coverage was increased did the pan-specific models perform the best (19). For TCR specificity, modeling the coverage of the peptide space is highly limited, and it hence remains an open question as to whether or not pan-specific models can demonstrate boosted performance.

TCR specificity is as described above defined by the combined signal contained within all 6 CDR loops. Most prediction models have however focused only on the CDR3 loops (and many

as stated above only on CDR3β). We have earlier demonstrated how a simple similarity-based model could benefit from the incorporation of information from CDR1 and CDR2 (5), but the overall importance of expanding the CDR information in the context of ML models remains to be settled.

Finally, the development of ML methods within TCR specificity prediction is challenged by the lack of a well-defined baseline model for assessment of ML model performance increase and justify the application of more complex model architectures. Given the very short length of CDRs, usually consisting of 5-25 residues, and the stochastic nature of the generation of in particular CDR3, commonly used evolutionary-based alignment methods cannot be applied here.

Here, we set out to investigate these fundamental questions for the optimal development of TCR specificity prediction models. It is essential to underline that we are not seeking to benchmark different published methods, but that we are solely seeking to address and answer questions related to best practices for developing and evaluating TCR-pMHC models. This with the purpose of aiding the field as a whole, by establishing a foundation and best practice for future work allowing researchers to avoid repeatedly addressing these fundamental issues, and rather focus on developing novel ideas enabling faster progress.

Materials and Methods

Data Preparation

The initial datasets were collected from IEDB, VDJdb, McPAS and 10X Genomics Single Cell Immune Profiling of four donors (20). The original dataset consisted of 21,121 unique paired TCRs relative to 499 peptides and 14 different HLA molecules. Non-binding peptide-TCR pairs were obtained from the 10X dataset. In the 10X assay, T cells were exposed to a panel of 50 peptide-MHC multimers. A negative TCR is defined as a TCR that does not bind any of the tested peptides and that has a Unique Molecular Identifier (UMI) count of 0.

Only data points with both CDR3 α- and β-chains and V/J gene annotations were kept. Further, any cross-reactive TCRs were removed, and the data was restricted to TCRs with CDR3α/β lengths in a range from 6 to 20 amino acids. Finally, only peptides with at least 100 positive TCRs were considered (11). After these initial cleaning steps, the dataset contained 4,111 positive peptide-TCR instances, spanning 10 different peptides and 4 HLA molecules. The negative pool of TCRs counted 40,949 TCRs negative to 6 out of the 10 peptides present in the positive set. The positive TCRs specific to the four non-overlapping peptides were discarded.

The set of positive TCRs was redundancy-reduced with the Hobohm 1 algorithm (21) applied to the CDR3 α- and β-sequences. The TCRs were first sorted in descending order according to the sum of the CDR3α CDR3β sequence lengths. Briefly, the Hobohm 1 algorithm starts by placing the first TCR into the non-redundant list. Iteratively, all the TCRs are similarity scored against the list of non-redundant TCRs: if the similarity to all the non-redundant TCRs is less than a specified threshold, then the new TCR is assigned to the non-redundant list, otherwise it is discarded. The similarity between sequences was calculated using the kernel similarity measure as defined in (22) and was calculated as the average of the CDR3α- and β-similarity scores. For the positive set, a threshold of 0.95 was chosen to ensure that only highly similar entries were removed. A similar approach was used to reduce the set of negatives, but with a similarity threshold of 0.9. After running the Hobohm 1 algorithm, 3,400 positive and 36,366 negative TCRs were left in the two data sets.

Once the redundancy in the positive set was reduced with the Hobohm 1 algorithm, the data points were randomly split into 6 partitions, 5 for cross-validation and one for external

evaluation. For each partition, for each positive peptide-TCR combination, 5 TCRs were sampled from the pool of negative TCRs and added to the partition of the peptide-TCR. These negatives are referred to as true negatives or 10X negatives. Each partition was further augmented with swapped negatives. Here, each positive TCR was paired with 5 peptides (different from the target peptide) and labeled as swapped negative.

The last step in the data curation was to reconstruct the full TCR sequences and annotate gene usage in the CDR loops. First, the full TCR sequences were constructed from V/J genes + CDR3: the CDR3 sequence was merged on the C-terminus of the V gene by looking for a cysteine (C) in the last six residues of the V gene sequence and on the N-terminus by matching a phenylalanine (F) or a tryptophan (W) followed by a glycine (G) within the first 11 amino acids of the J gene sequence. Lastly, Lyra (23) was used to annotate the CDR and 2 loops. A total of 473 positive TCR sequences were removed in this step, due to a failure in the TCR reconstruction or CDR annotation.

The final dataset consists of 2,541 unique positive peptide-TCR pairs, 12,848 negatives from 10X and 12,705 swapped negatives. A summary of the peptides included in the training set is shown in Table 1.

Peptide Sequence	**Organism**	**HLA**	**# positive TCRs**
GILGFVFTL	Influenza A virus	HLA-A*02:01	969
RAKFKQLL	Epstein Barr virus	HLA-B*08:01	659
ELAGIGILTV	Melanoma	HLA-A*02:01	316
IVTDFSVIK	Epstein Barr virus	HLA-A*11:01	275
GLCTLVAML	Epstein Barr virus	HLA-A*02:01	173
NLVPMVATV	Human CMV	HLA-A*02:01	149

Table 1: Description of the peptides included in the training set.

Baseline Model

A baseline model was used to benchmark the performance of the NetTCR model. The baseline used here was inspired by (9) and is solely based on TCR similarities. As for TCRmatch, the kernel similarity (22) measure was used. Briefly, this measure assigns a similarity score between two sequences by comparing all the *k*-mers, with *k* ranging from 1 to the length of the shortest sequence. For a fixed value of *k,* the BLOSUM62 score of all the *k*-mers from the first sequence against the *k*-mers from the second sequence is computed. The similarity score is then given by the self-similarity normalized sum of all the BLOSUM scores, for all the values of *k.*

For each peptide, a database of positive TCRs to the peptide from the training set was constructed and a query with positive and negative (both 10X and swapped negatives) TCRs from the evaluation set. Each TCR in the query is scored against the database using the kernel similarity score. The prediction for a given TCR in the test set is then given by the similarity score to the nearest neighbor in the training set. For the CDR3 model, the similarity score is calculated as the average of the similarities of α- and β-chains. When adding CDR1 and 2 to the model, the overall similarity is calculated as a weighted average of the similarities of each of the 6 CDR loops (3 for the α- and 3 for the β-chain) using weights [1,1,4] and [1,1,4] as suggested earlier (5). It should be noted that the baseline model is inherently peptide-specific as databases and queries are constructed for each peptide separately. TCRbase-1.0, a web server version of the baseline model, is available at https://services.healthtech.dtu.dk/service.php?TCRbase.

NetTCR Model

NetTCR is a sequence-based 1D-convolutional neural network, similar to the one proposed by (11). The inputs to the network are the amino acid sequences of the six CDR loops; for the pan-specific model, also the peptide sequence is used as input to the network. The inputs are zero-padded to the left, to ensure the same lengths across input: 10 for CDR 1 and 2, 20 for CDR3, and 13 for the peptides. The sequences are encoded using the BLOSUM50 (24) encoding scheme, mapping each amino acid into a vector with 20 entries. The encoded sequences are processed independently by different convolutional blocks. Each block applies 1D convolutions with 16 filters and kernel sizes {1, 2, 5, 7, 9} (80 filters for each sequence in total). The outputs of the convolutional layers are max-pooled across the sequence length dimension and concatenated. The final part of the network consists of a hidden layer with 32 neurons and an

output layer with a single neuron, giving the binding score of the input peptide and TCR. The sigmoid activation function was used in all the layers of the network.

Model training

All models were trained using nested 5-fold cross-validation for 200 epochs with early stopping, monitoring the validation loss. Adam optimizer was used, with a learning rate of 0.001. The code was developed in Python 3.7; the neural networks were designed using Pytorch 1.11 and the models were trained on an NVIDIA® GeForce GTX TITAN X GPU.

Performance Evaluation

The predictive power of the models was measured using the area under the receiver operating characteristic curve (AUC) and AUC 0.1, defined as the normalized area under the ROC curve with a maximum false positive rate of 0.1. The performance was assessed also with Positive Predictive Value (PPV), defined as the proportion of positive labeled TCRs within the top *n* predictions, where *n* is the number of positive data points in the set.

Each proposed model was trained using nested 5-fold cross-validation resulting in 20 individual networks. The performance was assessed on the left-out evaluation set. Here, the ensemble of the 20 trained models was used and the evaluation predictions were calculated by the average of the predictions from each of the 20 models.

The performance of the models was evaluated in a per-peptide manner (i.e from the subset of TCRs with target values towards a given peptide). For each model, an overall performance was also given by the average AUCs across peptides. We reported the average AUCs both as a mean value of the AUCs from each peptide and as a weighted average of the peptide AUCs, weighted by the number of positive TCRs for that specific peptide in the evaluation set. Each model's performance was reported by analyzing two tasks: i) positives versus 10X negatives prediction; ii) positives versus swapped negatives prediction.

To overcome the problem of having peptide-specific prediction biases, we performed calibration by transforming the raw prediction scores into percentile rank scores. The rank scores were estimated using a set of 13,847 COVID-specific TCRs (25), not sharing any overlap with the

training set. Percentile rank scores for a query TCR was next estimated as the proportion of COVID TCRs that scored higher than the considered TCR, in terms of raw prediction score.

To assess whether the differences in performance were significant, a bootstrap test was performed on the AUC values. Given two prediction vectors from two different models to compare, these were sampled *n* times with replacement, with the same size as the original vectors. Given the null hypothesis that the two models performed equally, a p-value was calculated as the number of times the AUC of the first model, calculated on the resampled vector, was smaller than the one from the second model, normalized by *n*.

Results

Here, we set out to investigate three essential questions related to the modeling of TCR specificity namely i) the use of peptide- versus pan-specific models, ii) how to best define negative data, and iii) the impact of model-integration of CDR1 and 2 loops. The three questions were addressed by developing and comparing the performance of simple ML models inspired by the earlier NetTCR architecture trained and tested using cross-validation of data extracted from the public domain.

Baseline Model

As a baseline model to compare the performance of the more complex ML models, we designed a simple similarity-based model for predicting TCR specificity, TCRbase-1.0, under the assumption that the TCRs that bind the same epitope share a high degree of sequence similarity. Here, for each peptide, a prediction for both positive and negative TCRs from the evaluation set was obtained by comparing these TCRs to all the positive TCRs for that specific peptide in the training set. The similarity score of two TCRs was given by the weighted sum of the similarities of the single CDR loops (see methods). We experimented with different sets of weights for the CDRs, as shown in Figure 1 and Supplementary Figure 1. These results suggest that including CDR1 and CD2 results in an improved predictive power of the baseline model (p-value<0.001 for all the peptides except IVT and NLV, based on a bootstrap test on the AUC values, with 1000 repetitions). Given the overall improved prediction of the model with CDR3s weighted four times higher than CDR1 and 2, we set these weights to be the default configuration of the baseline model.

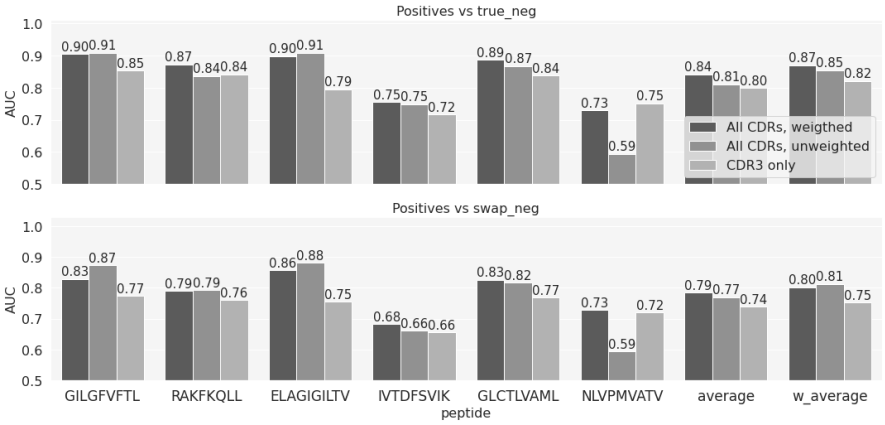


Figure 1: Baseline model performance for weighted and unweighted CDRs. Performance is reported as the AUC for each individual peptide, as well as the average and weighted (by number of positive TCRs) average AUC over the 6 peptides. The performances of three version of the baseline are shown: weighted, where the similarity is given by the weighted sum of the similarities of the three CDRs using the weights [1, 1, 4]; unweighted where all the CDRs are given equal weights; CDR3 only baseline, where CDR1 and 2 are given a weight of 0.

Peptide- vs pan-specific model

Next, we wanted to investigate whether peptide or pan-specific models would yield better performance. Ideally, one would like to train pan-specific models pooling all peptide-TCRs in the training data. Thereby, potentially allowing the model to leverage and transfer information between different TCR-pMHC combinations. Such data leverage is however only expected to be beneficial in situations where binding mode information is shared between peptides.

To compare the predictive power of peptide versus pan-specific models, two sets of models were trained using cross-validation and next evaluated using the left-out evaluation data set (for details see methods). Peptide-specific models were trained for each of the 6 peptides in the training data. The pan-specific model was trained on all data combined. All models were trained using an identical architecture, including the CDR3α and β sequence information from the TCRs, and the peptide sequence as inputs (the peptide information was fully conserved for the

peptide-specific models). The result of this experiment is shown in Figure 2 and demonstrated both for the individual peptides and the combined average performance values that for the data included in this study, the peptide-specific models in the majority of cases achieved superior performance. Particularly for the positives vs swapped negatives prediction task, all the differences are significant, except for the GIL peptide (p-value<=0.01, bootstrap with 1000 repetitions). Supplementary Figure 2 provides AUC01 and PPV values for the same experiment. Given this, the subsequent work focused only on peptide-specific models.

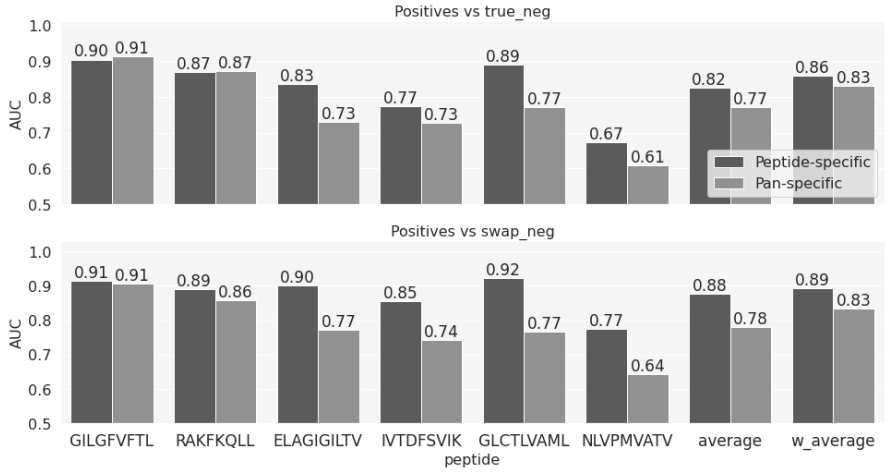


Figure 2: The predictive performance for each peptide measured in terms of AUC of the NetTCR architecture based models trained on α- and β-chains and stratified on negative usage and peptide- versus pan-specific approach. Average and w_aveage denotes the average and weighted (by the number of positive TCRs) average AUC over the 6 peptides.

On the different sources of negatives

Nextly, we aimed to investigate the impact of the different sources of negative data points on model performance: 10X negatives and swapped negatives. Briefly, the former set of negatives was derived from the 10X dataset and it is formed byTCRs that were found to not bind any of the 50 tested pMHC multimers. The swapped negatives are artificially generated by pairing TCR sequences with peptides aside from the one to which they were originally annotated to bind.

To investigate the performance impact of the different types of negative data, three models were trained. The first model was trained on the full data, i.e., positives, 10X and swapped negatives. Two more models were trained including either the 10X or swapped negatives. All models were trained using 5 fold cross-validation and evaluated on the 6th independent data set. The results of this experiment are shown in Figure 3 and Suppl. Figure 3, and demonstrated that the models trained on the complete set of negative data overall performed superior compared to the other models. That is, the model trained on the mixed type of negatives outperformed the model trained only on swapped negatives when asked to differentiate between positive and 10X negatives (upper panel). Likewise, it outperformed the model trained on 10X negatives when asked to differentiate between positive and swapped negative (lower panel). Further, the model trained on mixed negatives only suffered a limited decrease in performance when evaluated on the type of negative used to train the two other models. Given these results, we focused on the model trained using mixed negative data moving forward.

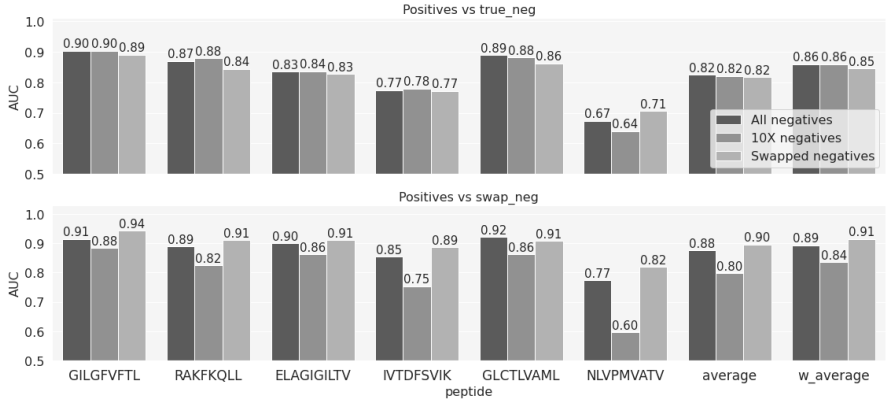


Figure 3: The predictive performance of the three models trained using negatives either from the 10X dataset, the swapped or both combined. The performance is evaluated in terms of AUC on two evaluation sets, each sharing positive observations, but with negatives defined by either true negatives from the 10X dataset or swapped negatives. Average and w_aveage denotes the average and weighted (by number of positive TCRs) average AUC over the 6 peptides

Adding CDR1 and CDR2

We next expanded the NetTCR architecture to also include CDR1 and -2 sequences as input, hereby representing the TCR as 6 sequences, the three CDRs from the α chain and the three from the β. Figure 4 shows the AUCs on the evaluation set of the model with all the CDR and the model with only CDR3s. Figure 4 demonstrates an overall improved performance when adding the CDR1 and 2. This gain is larger when looking at the AUC calculated on the positive vs swapped negative prediction task (Figure 4, lower panel) compared to positives versus true negatives (Figure 4, upper panel). Except for the GLC peptides, the model trained on all the CDRs significantly outperforms the one trained on CDR3 only (p-value<0.001, based on a bootstrap test with 1000 resampling with replacement) across all the peptides, when looking at the positives versus swapped negatives prediction. AUC01 and PPV comparisons are shown in Suppl. Figure 4.

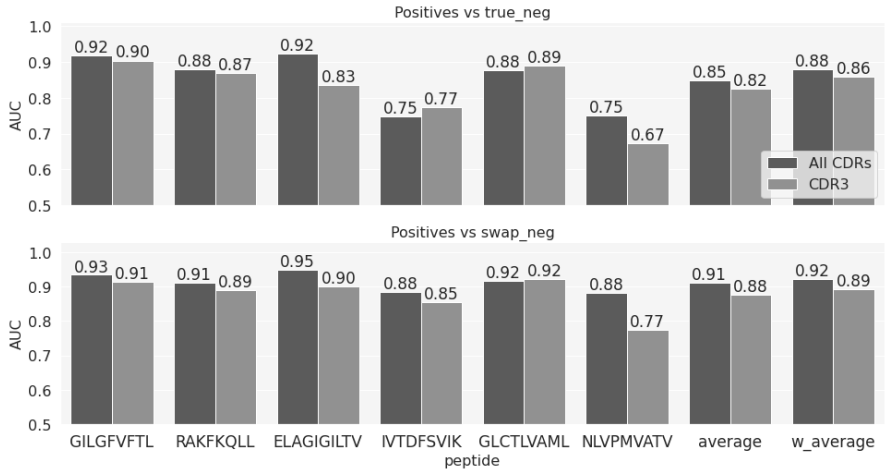


Figure 4: Performance comparison in terms of AUC for the NetTCR model using all CDR loops versus using only CDR3 loops from both α- and β chains.

Lastly, we compared NetTCR to the baseline model (TCRbase) with weighted CDRs contributions, as shown in Figure 5 and Supplementary Figure 5. The two models achieved comparable performance with a minor advantage of NetTCR when tested on the task of predicting the positive vs 10X negatives (Fig. 5, upper panel). However, NetTCR significantly outperformed the baseline (p-value<0.001, bootstrap test with 1000 repetitions) for all

evaluations when separating between positives and swapped negatives (Fig. 5, bottom panel).

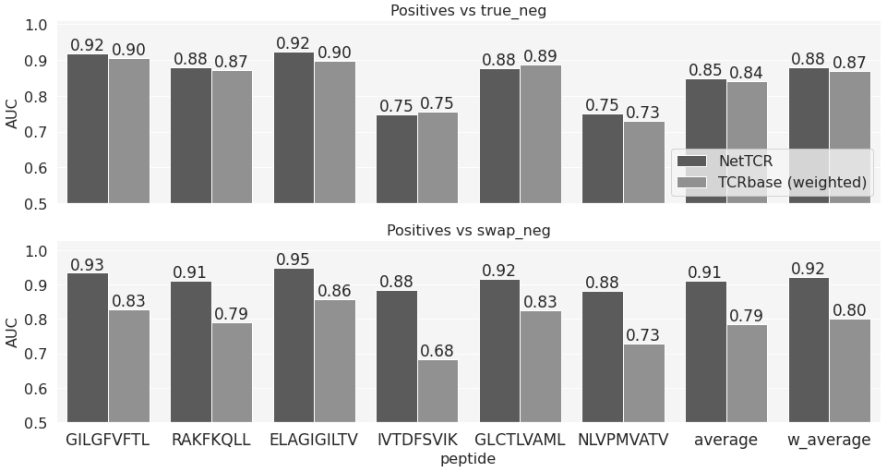


Figure 5: Predictive performance measured in terms of AUC for the peptide-specific NetTCR CDR123 model and the baseline.

Predicting peptide targets

So far, the performance evaluations performed have focused on evaluating to what degree models can differentiate between TCRs being positive or negative towards a given peptide. Equally interesting is whether a model is capable of identifying the true target peptide from a pool of possible peptides. To evaluate this, we compiled a data set where all the positive TCRs were paired to all the six peptides in the training set. Next, we used the peptide-specific models to get predictions for these peptide-TCR combinations and the scores were sorted in descending order. Ideally, the TCR paired to its target peptide should get a rank of 1, meaning that the prediction score for this true positive combination was the highest among all possible combinations resulting in 0 false positive predictions. The results of this experiment are shown in Figure 6. Here, the rank distribution for the positive TCRs for each peptide is shown. Most of the TCRs are observed to get a rank of 1, meaning that they were assigned to the correct peptide and thus received the highest score by the model corresponding to the correct target peptide. In all cases, the rank distributions are improved compared to the uniform distribution of a random model. However, the proportion of top-ranked predictions varied between the different peptides with values above 80% for the three most covered peptides and a drop to around 55%

for the three least covered. The number of top 1 positive TCRs for each peptide are GIL 114/136, RAK 77/96, ELA45.53, IVT 22/38, GLC 16/27, NLV 10/19.

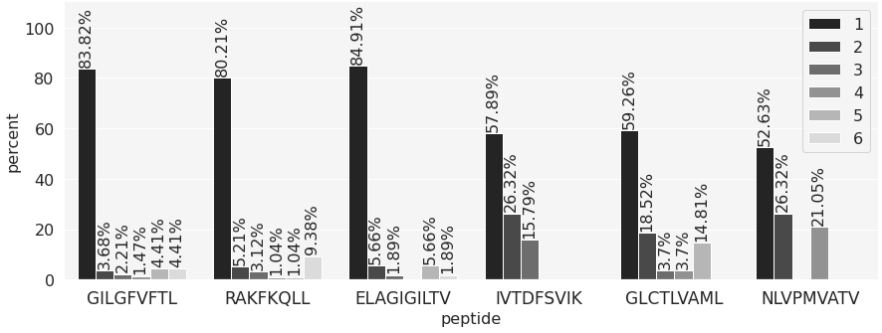


Figure 6: Peptide ranking analysis. Each positive TCR in the evaluation set was paired with all 6 peptides and predictions were obtained using the peptide-specific models. For each TCR, the six prediction scores were sorted in descending order and a rank was obtained. A rank of 1 means that the model correctly predicted the true TCR-peptide pair, assigning the highest score. The bars in the plot show the proportion of TCRs for each rank value.

To further investigate the source of these performance variations, Figure 7 shows box-plots of the prediction scores for different subsets of TCRs. Here the "top_TP" and "second_TN" refer to scores of the top and second scoring peptide for a given TCR, in the situation where the true peptide is ranked top-one. The other two distributions refer to the case where the model was not able to top-rank the correct peptide for the TCR. Here "top_false FP" displays the distribution of the prediction scores for the wrongly predicted top-one peptides, and "FN" is the score distribution for the correct peptide. Comparing the first two box-plots thus informs about the gap in the scores between top one and two in the situation of a correct prediction, and the last two plots about both the overall score distribution for TCRs with wrong predictions and the score of the best peptide in these situations. Several important conclusions can be drawn from these plots. First and foremost are the score distributions for "top_TP" and "second_TN" in all cases very well separated, suggesting that in these cases, the model has high certainty in predicting the correct peptide target. Secondly, variations in score distribution for the "top_TP" between the different peptides - the median score values decrease as one moves from the highest covered (GIL) towards the least covered (NLV) peptides, suggest that a score calibration would

potentially benefit the peptide ranking evaluation. Lastly, the scores for the FN TCR are in all cases very low and distinctively different from the "top_TP" score distributions. This strongly suggests that these FN TCRs at least in part are TCRs, which have been incorrectly annotated. We can pursue this further by investigating the source of the TCRs in the two classes “top_TP” and “FN”. Doing this, we find that one publication (26) in particular is enriched in “FN” TCR. This publication contributes ~19% of the TCRs in the FN category while only contributing ~10% to the overall positive data set and ~5% to the top_TP category. The underlying source of this FN enrichment is unclear.

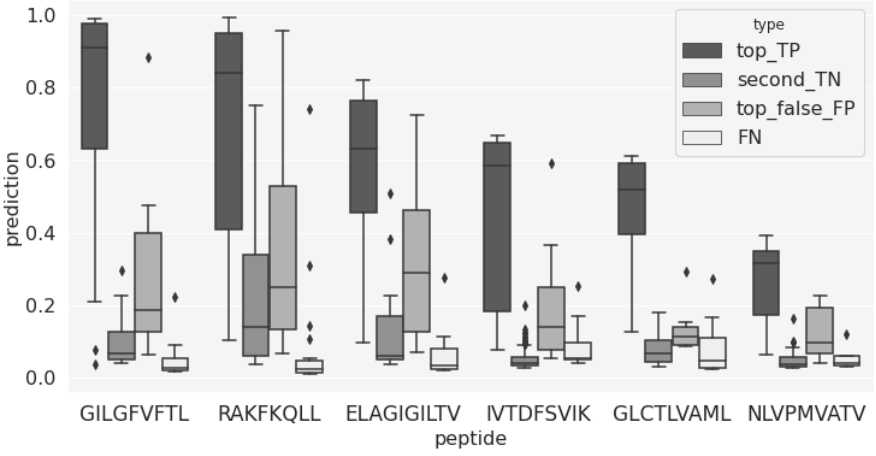


Figure 7: Box-plots of the prediction scores from the peptide ranking analysis. "top_TP" refers to the predictions for the positive peptide-TCRs that obtained the highest prediction score with the model trained on that specific peptide; "second_TN" shows the predictions for the second highest scoring TCR. "top_false_FP" and "FN" refer to a scenario where a TCRs gets the highest prediction score when paired to a peptide that is different from its target. "top_false_FN" shows the score distribution of these wrong combinations of peptide and TCR; "FN" represents the prediction score of the correct peptide-TCR pairs that did not score top 1.

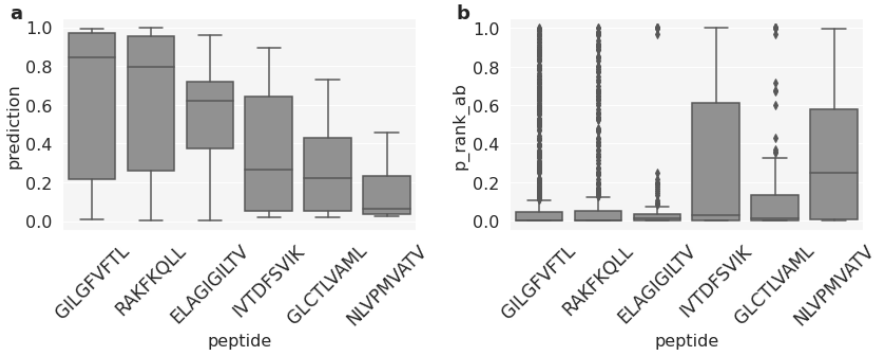


Figure 8: Motivation for using percentile ranks. Box-plots of the prediction scores (a) and percentile rank values (b) for the set of positive TCRs in the test CV sets.

As illustrated in Figure 7, the prediction scores for the top1 TCRs have very different median values, depending on the peptide. In general, this happens for all the positive TCRs, as shown in Figure 8a. This represents a limitation when comparing predictions from different models, thereby indicating that a score calibration is needed. To address this, we applied a percentile rank transformation to the raw prediction scores to avoid these peptide-specific scoring biases, as described in Materials and Methods. Here, a set of 13,847 COVID-specific TCRs (25) were used to estimate the background distributions of the peptide-specific models. The percentile rank score for a peptide-TCR pair in the evaluation set was then estimated as the proportion of the background COVID TCRs with a higher prediction score than the pair in consideration. Figure 8b shows the percentile rank scores for the positive TCRs. Except for the NLV peptide, the median values of the percentile rank scores are now comparable across peptides. This suggests that using the percentile rank scores is more appropriate than using the raw prediction scores, making the different models more directly comparable.

Performance as a function of distance to training data

Next, we wanted to investigate how the similarity between the training and evaluation set drove the performance of both NetTCR and the baseline models. In these experiments, we exclude positive TCRs with a percentile rank score above 0.3 (to exclude potential noise imposed by the FN TCRs described above). For each TCR, we defined its similarity to the training set as the kernel similarity score to its nearest neighbor TCR, either positive or negative, in the training set.

Next, we excluded TCRs with a similarity to train above a given threshold and calculated the AUC value based on the predictions of the remaining data points. Figure 9 shows the results of this experiment, using different 10 similarity threshold values between 0.89 and 0.98 (results shown for the three most frequent peptides). These results show that when the TCRs in the evaluation set are allowed to share a similarity to the training set up to 0.98, the baseline and NetTCR models perform similarly. However as the maximum similarity between the train and evaluation set is reduced, the gap in performance between the two models increases (in particular for the GIL and RAK peptides), with a substantial drop in baseline AUC for the baseline model, while NetTCR to a high degree maintains performance.

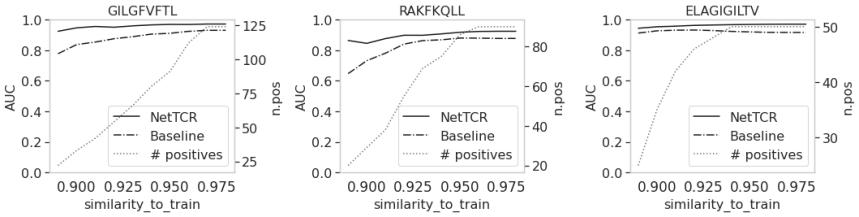


Figure 9: AUC values as a function of the similarity between training and evaluation set. Percentile rank transformation was applied to the TCRs and only positive TCRs with a rank score less than 0.3 were kept in this analysis. For each TCR in the evaluation set, we calculated the similarity to the training set using the kernel similarity score. We then removed the TCRs with a similarity above a threshold and calculated the AUC. The curves in the plots show the AUC varies when different similarity thresholds were used to filter the evaluation set; 10 similarity values between 0.89 and 0.98 were chosen. The dashed line shows the number of positive TCRs left in the evaluation set at each step of filtering by similarity to the training set.

The NetTCR-2.1 method

The presented model is available as a web-server implementation at https://services.healthtech.dtu.dk/service.php?NetTCR-2.1. The server allows users to make TCR-binding predictions to one or more peptides, using the peptide-specific models. It is possible to use either the models trained on CDR3α- and β-sequences or trained using all the six CDR loops.

The output of NetTCR-2.1 is a list of CDR-peptide pairs along with the binding prediction. For each prediction, the method outputs also the percentile rank score, estimated from a

background set of 13,847 COVID-specific TCRs. The percentile rank is a normalized score across the different peptide-specific models, ranging from 0 to 1, where 0 is the best possible percentile rank. The rank score should serve as a guideline to select a peptide invariant threshold on the binding probability prediction. For each peptide, the threshold could be defined as the 75th percentile of the background prediction score distributions (boxplots shown in Figure 8b).

Discussion

Here, we present NetTCR-2.1, which is an extension of our earlier NetTCR-2.0 method for prediction of pMHC-TCR interactions. The main augmentation is an extended peptide coverage and the ability to include all CDRs in the binding prediction.

In our work, we investigated several important aspects of model development when aiming at predicting TCR specificity and have presented our results of this, aiming at supplying the TCR-specificity prediction field with a set of suggested best practices. These, include recommendations on strategies for data partitioning and redundancy reduction, the use of peptide versus pan-specific modeling, the source of negatives, inclusion of CDR1 and CDR2 information, the importance of benchmark comparison of simple sequence similarity-based baseline models, and model performance comparison in the context of distance to training data. In the following, we will briefly summarize our findings and associated conclusions on each of these topics.

Strategies for data partitioning and redundancy reduction

Traditionally, TCR-pMHC specificity data has been focused on CDR3β for reasons previously described. However, the advent of high throughput single cell technologies has resulted in a substantial increase in publicly available data on paired TCR α- and β-chain to cognate pMHC-target data. However, it is evident that these data reflect ongoing research into model organisms or diseases like Influenza A virus, Epstein Barr virus, Melanoma and Human CMV. Furthermore, often only positive observations of TCR-pMHC interaction are reported, biasing the databases. As a reflection of this, the TCR sequences currently available share a high degree of redundancy. Therefore, in order to achieve, to the degree possible, non-biased training and evaluation of the developed models, it is important to address redundancy. This is true particularly for modern modeling frameworks, where parameter space is very large and they are

prone to overfitting. The below described guidelines aim to minimize this risk as much as possible.

Due to the genetic mechanisms underlying TCR generation, classical alignment-based similarity approaches using for instance Blosum matrices and affine gap penalties are nonsensical. Therefore, we propose using alignment-free methods such as the kernel method described by Shen et al. (22) to estimate sequence similarity and then subsequently perform redundancy reduction using e.g. the Hobohm-1 algorithm. Lastly, we recommend performing pre-clustering prior to partitioning the data using e.g. single linkage to ensure the least possible overlap between partitions.

Peptide versus pan-specific modeling

We trained two versions of the NetTCR model; a pan- and a peptide-specific, both trained only including the CDR3 for simplicity Ideally, a pan-specific approach should be more generalizable and rely less on the individual peptides in the training set, aiming at capturing the global signal. The clear advantage is that such a model would be able to make predictions for TCRs specific to any peptide, even for those peptides that are represented by only a small sample in the training data or even absent. Given the data currently available, the peptide-specific models were however found to outperform the pan-specific ones. Using the experiences gained from modeling pMHC-interaction where the early pan-specific model also performed at par or slightly worse than allele-specific (27), this is likely due to the limited volume and coverage of the data volume currently available. We observe that TCRs specific to different peptides do share many features, rendering cross-learning across peptides not achievable at the moment. As more data becomes available, we expect that it will be possible to train pan-specific models.

The source of negatives

A critical point when developing an ML model for binary classification is the definition of negative data. Insufficient consideration of this can lead to biasing (28). The publicly available datasets of TCR-pMHC sequences almost exclusively contain examples of positive binding pairs. Only the recently published 10X Genomics dataset contains both positive and negative data points. Another common approach for generating artificial negatives is to mispair positive peptide-TCR pairs. Here, we have compiled a training data set with both 10X negatives and internal mispairing of peptides and TCRs, referred to as swapped negatives. We investigated the impact of both sources of negatives by training the same neural network on different datasets, including either both sources or negatives or only one of the two. In all these

experiments, the NetTCR CDR3 peptide-specific model was adopted. To better understand how the two negative sets affected the performance, the AUC values were calculated for the positive vs swapped negatives prediction task and for the positives vs 10X negatives. The model trained with only swapped negatives showed a high predictive power when evaluating the positives vs swapped negatives predictive power, as that was specifically the task the model was trained for. However, the performance dropped when evaluating how this model could distinguish between positives and 10X negatives. Vice versa, the model trained on 10X negatives was able to make good predictions for the positives vs 10X negatives task but suffered a major drop in performance when making predictions on the swapped negatives. The model trained on the entire dataset, i.e. positive TCRs, 10X and swapped negatives, showed the ability to make satisfying predictions on both tasks of predicting 10X and swapped negatives. These results suggest that both types of negatives are needed to accomplish more tasks with one unique trained model. Furthermore, given the large drop in performance of the network trained on the 10X negatives on the swapped data, the swapped negatives play a more important role in learning how to differentiate between positive and negative TCRs. This aspect could be a consequence of the fact that, with the mismatched negatives, the network is shown the same TCR sequences that are positive in some cases and negative in others; on the contrary, positive and 10X negative TCR are two disjoint sets. Hence, the network might capture a signal to distinguish positives and negatives that are different in sequences, but not learn the rules that make a TCR positive to one peptide and negative towards others.

Inclusion of CDR1 and CDR2 information

Most of the available models to predict peptide-TCR interaction are focused on CDR3β or paired CDR3αβ sequences, and only a few recently published works have added V/J genes information in the model as one-hot encoded features (6,10). Here, we have developed a neural network that takes as input the full set of the 6 CDR sequences. The full-length TCR was reconstructed from the V/J genes and CDR3 sequence, and the CDRs were annotated using Lyra (see Material and Methods for details). We compared the model trained on the full set of CDRs to the one trained on CDR3αβ data. On average, the model trained on the 6 CDR sequences showed higher AUC values compared to the CDR3αβ model, across the peptide set. For some of the peptides, the inclusion of CDR1 and 2 resulted in a substantial increase in AUC. This is the case, for instance, for the ELAGIGILTV peptide. However, this gain in performance might be driven by a bias in the V gene data. In our data set, 85% of the positive ELA CDR1α and 2α are encoded by the TRAV12-2*01 gene; this gene is present only in a

minor proportion (5%) in the negative set. It is not clear if this bias is due to the data collection or if it is a biological signal.

Benchmark comparison of simple sequence similarity-based baseline models and models comparison in the context of distance to training data

Together with NetTCR-2.1, we have here proposed TCRbase, a similarity-based approach to predict TCR-peptide interaction, under the assumption that TCRs with similar sequences recognize the same epitope. We showed that this model achieved comparable performance to the one of NetTCR, while being very simple. Our results align with previous findings (17,29,26). A closer analysis of our results revealed that TCRbase performed at par with NetTCR when separating positive versus 10X negative TCRs; however, the gap in performance between the two models was enlarged on the positives versus swapped negatives prediction task, where NetTCR significantly outperformed TCRbase. This behavior suggests that the 10X negatives are very different from the positive TCRs, and this dissimilarity makes it trivial for a similarity-based model to distinguish between positives and negatives. This is not the case for the swapped negatives, as they are positive to some other peptide. Here, TCRbase to a higher degree fails in separating the positive and negative set, while NetTCR maintains performance, indicating that the neural network has learned some features beyond sequence similarity. The generalizability of NetTCR is furtherly confirmed when comparing the model's performances in the context of distance to training data. When the evaluation set is allowed to be highly similar to the training data, NetTCR and TCRbase have comparable performance in terms of AUC. As the TCRs similar to the training data are removed, TCRbase suffers a drop in performance, whereas NetTCR is able to maintain the predictive power. We believe this result is essential as a validation of the greater potential for generalization of the NetTCR machine learning-based method over the more simple similarity-based approach, and strongly suggest that such similarity-based models and performance evaluations as a function of distance to training data are included as baselines in future works developing TCR specificity prediction models.

Data Availability Statement

The data used in this study is publicly available and can be found at https://www.iedb.org/, https://vdjdb.cdr3.net/search, http://friedmanlab.weizmann.ac.il/McPAS-TCR/ and https://support.10xgenomics.com/single-cell-vdj/datasets. The curated dataset used for training,

validating and testing NetTCR-2.1 and TCRbase is available at https://github.com/mnielLab/train_NetTCR/tree/main/data.

Code Availability Statement

The NetTCR-2.1 code is available at https://github.com/mnielLab/train_NetTCR. The trained NettCR-2.1 models are available as a web-server tool as https://services.healthtech.dtu.dk/service.php?NetTCR-2.1. TCRbase is also available as a web server at https://services.healthtech.dtu.dk/service.php?TCRbase.

Acknowledgements

The work was supported by the National Institute of Allergy and Infectious Diseases (NIAID), under award number 75N93019C00001

Authors Contribution

AM and MN designed the study. The experimental data used in the study was collected by AM. AM generated the computational results and figures, with contributions from LEJ and MN. All authors contributed to the methodology and provided scientific feedback. All authors wrote and approved the manuscript.

Competing Interests

The authors declare no competing interests.

Bibliography

1. Krogsgaard M, Davis MM. How T cells “see” antigen. *Nat Immunol* (2005) **6**:239–245. doi:10.1038/ni1173

2. Davis MM, Bjorkman PJ. T-cell antigen receptor genes and T-cell recognition. *Nature* (1988) **334**:395–402. doi:10.1038/334395a0

3. Klinger M, Pepin F, Wilkins J, Asbury T, Wittkop T, Zheng J, Moorhead M, Faham M. Multiplex Identification of Antigen-Specific T Cell Receptors Using a Combination of Immune Assays and Immune Receptor Sequencing. *PLoS ONE* (2015) **10**:e0141561. doi:10.1371/journal.pone.0141561

4. Rius C, Attaf M, Tungatt K, Bianchi V, Legut M, Bovay A, Donia M, Thor Straten P, Peakman M, Svane IM, et al. Peptide-MHC Class I Tetramers Can Fail To Detect Relevant Functional T Cell Clonotypes and Underestimate Antigen-Reactive T Cell Populations. *J Immunol* (2018) **200**:2263–2279. doi:10.4049/jimmunol.1700242

5. Lanzarotti E, Marcatili P, Nielsen M. T-Cell Receptor Cognate Target Prediction Based on Paired α and β Chain Sequence and Structural CDR Loop Similarities. *Front Immunol* (2019) **10**:2080. doi:10.3389/fimmu.2019.02080

6. Zhang W, Hawkins PG, He J, Gupta NT, Liu J, Choonoo G, Jeong SW, Chen CR, Dhanik A, Dillon M, et al. A framework for highly multiplexed dextramer mapping and prediction of T cell receptor sequences to antigen specificity. *Sci Adv* (2021) **7**: doi:10.1126/sciadv.abf5835

7. Springer I, Besser H, Tickotsky-Moskovitz N, Dvorkin S, Louzoun Y. Prediction of Specific TCR-Peptide Binding From Large Dictionaries of TCR-Peptide Pairs. *Front Immunol* (2020) **11**:1803. doi:10.3389/fimmu.2020.01803

8. Gielis S, Moris P, Bittremieux W, De Neuter N, Ogunjimi B, Laukens K, Meysman P. Detection of enriched T cell epitope specificity in full T cell receptor sequence repertoires. *Front Immunol* (2019) **10**:2820. doi:10.3389/fimmu.2019.02820

9. Chronister WD, Crinklaw A, Mahajan S, Vita R, Kosaloglu-Yalcin Z, Yan Z, Greenbaum JA, Jessen LE, Nielsen M, Christley S, et al. TCRMatch: Predicting T-cell receptor specificity based on sequence similarity to previously characterized receptors. *BioRxiv* (2020) doi:10.1101/2020.12.11.418426

10. Springer I, Tickotsky N, Louzoun Y. Contribution of T cell receptor alpha and beta CDR3, MHC typing, V and J genes to peptide binding prediction. *Front Immunol* (2021) **12**:664514. doi:10.3389/fimmu.2021.664514

11. Montemurro A, Schuster V, Povlsen HR, Bentzen AK, Jurtz V, Chronister WD, Crinklaw A, Hadrup SR, Winther O, Peters B, et al. NetTCR-2.0 enables accurate prediction of TCR-peptide binding by using paired TCRα and β sequence data. *Commun Biol* (2021) **4**:1060. doi:10.1038/s42003-021-02610-3

12. Bagaev DV, Vroomans RMA, Samir J, Stervbo U, Rius C, Dolton G, Greenshields-Watson A, Attaf M, Egorov ES, Zvyagin IV, et al. VDJdb in 2019: database

extension, new analysis infrastructure and a T-cell receptor motif compendium. *Nucleic Acids Res* (2020) **48**:D1057–D1062. doi:10.1093/nar/gkz874

13. Vita R, Mahajan S, Overton JA, Dhanda SK, Martini S, Cantrell JR, Wheeler DK, Sette A, Peters B. The Immune Epitope Database (IEDB): 2018 update. *Nucleic Acids Res* (2019) **47**:D339–D343. doi:10.1093/nar/gky1006

14. Tickotsky N, Sagiv T, Prilusky J, Shifrut E, Friedman N. McPAS-TCR: a manually curated catalogue of pathology-associated T cell receptor sequences. *Bioinformatics* (2017) **33**:2924–2929. doi:10.1093/bioinformatics/btx286

15. Zhang W, Wang L, Liu K, Wei X, Yang K, Du W, Wang S, Guo N, Ma C, Luo L, et al. PIRD: pan immune repertoire database. *Bioinformatics* (2020) **36**:897–903. doi:10.1093/bioinformatics/btz614

16. Jurtz VI, Jessen LE, Bentzen AK, Jespersen MC, Mahajan S, Vita R, Jensen KK, Marcatili P, Hadrup SR, Peters B, et al. NetTCR: sequence-based prediction of TCR binding to peptide-MHC complexes using convolutional neural networks. *BioRxiv* (2018) doi:10.1101/433706

17. Chronister WD, Crinklaw A, Mahajan S, Vita R, Koşaloğlu-Yalçın Z, Yan Z, Greenbaum JA, Jessen LE, Nielsen M, Christley S, et al. TCRMatch: Predicting T-Cell Receptor Specificity Based on Sequence Similarity to Previously Characterized Receptors. *Front Immunol* (2021) **12**:640725. doi:10.3389/fimmu.2021.640725

18. Sidhom J-W, Larman HB, Pardoll DM, Baras AS. DeepTCR is a deep learning framework for revealing sequence concepts within T-cell repertoires. *Nat Commun* (2021) **12**:1605. doi:10.1038/s41467-021-21879-w

19. Hoof I, Peters B, Sidney J, Pedersen LE, Sette A, Lund O, Buus S, Nielsen M. NetMHCpan, a method for MHC class I binding prediction beyond humans. *Immunogenetics* (2009) **61**:1–13. doi:10.1007/s00251-008-0341-z

20. 10X Genomics. A New Way of Exploring Immunity - Linking Highly Multiplexed Antigen Recognition to Immune Repertoire and Phenotype | Technology Networks A New Way of Exploring Immunity - Linking Highly Multiplexed Antigen Recognition to Immune Repertoire and Phenotype. Available at: https://www.technologynetworks.com/immunology/application-notes/a-new-way-of-exploring-immunity-linking-highly-multiplexed-antigen-recognition-to-immune-repertoire-332554 [Accessed January 20, 2021]

21. Hobohm U, Scharf M, Schneider R, Sander C. Selection of representative protein data sets. *Protein Sci* (1992) **1**:409–417. doi:10.1002/pro.5560010313

22. Shen W-J, Wong H-S, Xiao Q-W, Guo X, Smale S. Towards a Mathematical Foundation of Immunology and Amino Acid Chains. (2012)

23. Klausen MS, Anderson MV, Jespersen MC, Nielsen M, Marcatili P. LYRA, a webserver for lymphocyte receptor structural modeling. *Nucleic Acids Res* (2015) **43**:W349-55. doi:10.1093/nar/gkv535

24. Henikoff S, Henikoff JG. Amino acid substitution matrices from protein blocks.

Proc Natl Acad Sci USA (1992) **89**:10915–10919. doi:10.1073/pnas.89.22.10915

25. Minervina AA, Pogorelyy MV, Kirk AM, Crawford JC, Allen EK, Chou C-H, Mettelman RC, Allison KJ, Lin C-Y, Brice DC, et al. SARS-CoV-2 antigen exposure history shapes phenotypes and specificity of memory CD8+ T cells. *Nat Immunol* (2022) **23**:781–790. doi:10.1038/s41590-022-01184-4

26. Dash P, Fiore-Gartland AJ, Hertz T, Wang GC, Sharma S, Souquette A, Crawford JC, Clemens EB, Nguyen THO, Kedzierska K, et al. Quantifiable predictive features define epitope-specific T cell receptor repertoires. *Nature* (2017) **547**:89–93. doi:10.1038/nature22383

27. Nielsen M, Lundegaard C, Blicher T, Lamberth K, Harndahl M, Justesen S, Røder G, Peters B, Sette A, Lund O, et al. NetMHCpan, a method for quantitative predictions of peptide binding to any HLA-A and -B locus protein of known sequence. *PLoS ONE* (2007) **2**:e796. doi:10.1371/journal.pone.0000796

28. Sidorczuk K, Gagat P, Pietluch F, Kała J, Rafacz D, Bąkała L, Słowik J, Kolenda R, Rödiger S, Fingerhut LCHW, et al. Benchmarks in antimicrobial peptide prediction are biased due to the selection of negative data. *Brief Bioinformatics* (2022) doi:10.1093/bib/bbac343

29. Wong EB, Gold MC, Meermeier EW, Xulu BZ, Khuzwayo S, Sullivan ZA, Mahyari E, Rogers Z, Kløverpris H, Sharma PK, et al. TRAV1-2+ CD8+ T-cells including oligoconal expansions of MAIT cells are enriched in the airways in human tuberculosis. *Commun Biol* (2019) **2**:203. doi:10.1038/s42003-019-0442-2

CHAPTER 6

Benchmark of data-driven filtering approaches for single-cell screening of T cell specificity

In the last years, multiple models have been developed to predict TCR-pMHC binding events [18, 74, 83–89]. However, one of the limitations that emerged across different studies was the quantity and quality of the available data, typically generated by multimer sorting or re-exposure assay, followed by bulk sequencing. Single-cell (SC) technology promises the generation of large amounts of data, in a high-throughput manner. Furthermore, in the context of TCR sequencing, SC allows the generation of paired α and β TCR chains and both binding and non-binding events. However, this technology is not error-free and proper methods to handle the output are needed. De-noising single-cell data composed of T cell specificities is a new and inexperienced field. Here, immunoinformatics methods are crucial to increase the signal-to-noise ratio in the data, so that the community can truly benefit from single-cell assays. Currently, only two frameworks have been proposed that aim to de-noise single-cell data. The two methods are both data-driven but address the filtering in different ways. This chapter presents a benchmark of the two methods, ATRAP and ICON, and highlights the advantages and disadvantages of each approach.

The improved quality of the filtered data is assessed by training NetTCR-2.1, a deep learning model, on both raw and filtered data, showing that the predictive performance is increased when the data is de-noised. This suggests that the filtering frameworks remove artifacts and wrong annotations from the data, leading to a more reliable set of TCR-pMHC examples.

The results presented in the chapter come from an ongoing project in collaboration with Helle Rus Povlsen, Leon Eyrich Jessen and Morten Nielsen. My primary contribution to the project was to design a machine learning-based validation for the increased accuracy of de-noised data. The NetTCR-2.1 CDR3$\alpha\beta$ model was trained both on the raw 10x data and on the ICON and ATRAP-filtered datasets, showing improved predictive power of the filtered data both when training the model and on an external, independent dataset.

Benchmark of data-driven filtering approaches for single-cell screening of T cell specificity

Helle Rus Povlsen, Alessandro Montemurro, Leon Eyrich Jessen and Morten Nielsen

Department of Health Technology, Technical University of Denmark, Kgs. Lyngby, Denmark

Abstract

Pairing of T cell receptor (TCR) with its cognate peptide-MHC (pMHC) is a cornerstone in T cell-mediated immunity. Recently, single-cell sequencing coupled with DNA-barcoded multimer staining has made the high-throughput study of T cell specificity available. However, the immense variability of the TCR-pMHC interaction combined with the technology's low ratio of signal-to-noise in the generated data is complicating the study. Several approaches have been proposed for de-noising single-cell TCR-pMHC specificity data. Here, we present a benchmark evaluation of two such computational frameworks, ICON and ATRAP. The methods were applied and evaluated on the publicly available immune profiling data provided by 10x Genomics in terms of both internal metrics developed for the purpose, and by performance on independent data of machine learning methods trained on the raw and denoised 10x data. The conclusion from these benchmarks demonstrates both an increased signal-to-noise ratio in the denoised compared to the raw data, and overall superior performance of the ATRAP method over ICON when it comes to data consistency and performance when training and evaluating predictive models.

Introduction

The specificity of T cells forms the hallmark of cellular immunity. T cell specificity is determined by a triad of interactions between the T cell receptor (TCR), a peptide (p), and its restricting major histocompatibility complex (MHC). The TCR is a heterodimeric protein, typically composed of an α- and β-chain, which are formed during T cell development as a result of stochastic V(D)J gene recombination [90–94]. As a result of the somatic recombination, highly variable joining segments are introduced, facilitating a diverse TCR repertoire that ensures protection from a broad and ever-changing range of pathogens or cancerous mutations [95–97]. The joining segments are contained in a region known as the complementarity determining region 3 (CDR3). CDR1 and CDR2 reside in highly polymorphic regions of the V gene. The three CDRs form flexible loops of the TCR which engage with the peptide-MHC (pMHC) complex and thereby determine the specificity of the T cell [98–101].

Recent studies have elucidated common TCR sequence features of T cells that share specificity, and for selected pMHCs, it has been possible to predict the binding probability to TCRs novel to the trained model [18, 74, 83–89]. The current primary limitation is the lack of both quantity and diversity of training data generated by traditional specificity assays such as multimer sorting and re-exposure assays, followed by bulk sequencing of typically the TCRβ-chain. However, the advent of single-cell sequencing platforms promises high-throughput data which in addition intrinsically provides information of false binding pairs, as well as true pairs [102]. This type of data is expected to accelerate the understanding of TCR specificity.

10x Genomics has specifically developed an immune profiling platform that couples TCR sequencing of both α- and β-chains with DNA barcoded peptide-MHC (pMHC) multimers, DNA barcoded surface marker antibodies, and DNA barcoded cell hashing antibodies. The platform is designed to capture a single cell together with a gel-bead in emulsion (GEM) [103, 104]. Each GEM contains GEM-specific barcoded primers which ensure the back-tracing of transcripts to the cell-of-origin. As the platform promises single-

cell capture, the contents of a GEM should reflect a single cell and its associated barcoded analytes, hence GEM and cell may be used interchangeably. The GEM primers also contain a unique molecular identifier (UMI) which ensures quantification of transcripts unbiased by PCR amplification [105]. Thus, single-cell screening of TCR-pMHC interactions yields the TCR$\alpha\beta$ sequence and the expression level of both chains as well as the count of each unique pMHC binding which might be interpreted as T cell avidity [102].

In 2019, 10x Genomics released a large, state-of-the-art data set [102] which spurred activity within the TCR-pMHC modeling community [18, 74, 86–88, 106]. The 10x Genomics data contain T cell specificities from four healthy donors screened against a panel of 50 pMHCs which includes 44 pMHCs for positive selection and six negative control pMHCs [102]. However, this data presented new challenges. The single-cell platform is generally associated with a poor signal-to-noise ratio, which also affects this specificity data. The challenge was handled in various ways. In NetTCR-2.0, the data was utilized solely to define negative TCR-pMHC pairs, i.e. pairs that were not detected to bind any of the investigated pMHC complexes and thereby avoided handling the noise within the detected binders [18]. Since the true TCR-pMHC pairs are a point of contention, the authors of ImRex purposefully omitted the 10x data [87], while the authors of TcellMatch and DeepTCR relied on the network to extract the salient pMHC specific features of the TCRs [88, 106]. The authors of TCRAI were the first to develop a computational method, named ICON (Integrative COntext-specific Normalization), to discriminate true TCR-pMHC binding signal from nonspecific background noise [86]. Recently, we have proposed an alternative framework for this task called ATRAP (Accurate T cell and Antigen Pairing) [29]. ICON was developed based on 10x Genomics data, utilizing the negative controls to empirically estimate the background binding noise per donor. The UMI counts of pMHCs were then corrected by subtracting the donor-specific estimated background noise. UMI counts were further corrected by penalizing pMHCs multiplets i.e., GEMs containing multiple DNA barcodes corresponding to two or more different pMHCs. The final step of ICON is the normalization of UMI counts across pMHCs and GEMs to make them di-

rectly comparable. As a result, ICON identified a total of 53,062 T cells belonging to 5,722 unique clonotypes.

ATRAP takes a different approach. The framework was developed and tested on in-house single-cell data generated using the 10x Genomics platform similar to the public 10x Genomics data. The ATRAP framework consists of a series of filtering approaches to obtain increasingly accurate TCR-pMHC pairing. The first filtering step was based on identifying expected targets by comparing the UMI distributions of all pMHCs detected within a clonotype consisting of 10 or more GEMs. The key is to study GEMs in an ensemble rather than individually because deviations are averaged out. If a pMHC was distributed with a significantly higher mean UMI in the ensemble, we expected this pMHC to reflect the true target of the clonotype, collectively providing a golden standard. Based on the labeling of true and false targets, we could compute an accuracy score. Thresholds were set on UMI counts to maximize the accuracy. By globally applying the optimal threshold, the remaining clonotypes should ideally represent the same level of accuracy in their pMHC annotations. Another key step of ATRAP filtering is ensuring HLA correspondence between pMHC and the HLA haplotype of the T cell donor. In immune profiling assays, the option to hash cells by donor-of-origin enables the assignment of HLA haplotype restriction to each cell. Correspondence between the allele of pMHC and donor haplotype can be used to verify the assignment of the pMHC, assuming that a T cell is absolutely restricted to the allele for which it was selected during the thymocyte maturation process. In the public 10x data, the cells are not hashed, however, the experiment was run in parallel for each donor, enabling *in silico* hashing of the individual single-cell runs.

In this study, we report a benchmark of the two frameworks to recommend future applications of single-cell specificity data. Both methods are applied to the 10x Genomics data since this is the only data set containing negative controls as is required by ICON. As no external golden standard exists, the performance of the two methods is evaluated on internal performance metrics presented by Povlsen et al. [29]: GEM retention, accuracy, average binding

concordance, and AUC of similarity scores, as well as in terms of predictive performance of machine learning methods trained on the raw and denoised 10x data on independent data.

Materials and Methods

Data retrieval

The 10x Genomics data set used for this study was downloaded from https://support.10xgenomics.com/single-cell-vdj/datasets.

The benchmark data was curated by Zhang et al. employing their method, ICON (Integrative COntext-specific Normalization), for identifying reliable TCR-pMHC interactions. Data was downloaded from http://advances.sciencemag.org/cgi/content/full/7/20/eabf5835/DC1. This set contains 53,062 cells (here referred to as GEMs) that passed the ICON filtering with ICON-corrected pMHC and TCR annotations. The ICON output provided with the publication contains a fifth donor, donor V, which was removed from the set (14,052 GEMs).

Data curation

The data consists of four sets of single-cell RNA sequencing and immune profiling from four healthy donors. The HLA haplotype of each donor was manually added to each set. The sets were concatenated for one combined analysis. Few GEM-specific 10x barcodes (GEM barcodes) were duplicates across the donor sets, therefore the barcodes were additionally suffixed by donor, i.e. AAACCTGTCTAACTTC-6-s2. Cells (referred to as GEMs) were removed if the annotated CDR3αβ sequences were not productive, full length, or contained non-IUPAC characters, resulting in 181,913 GEMs. Differently annotated clonotypes sharing VJ-CDR3αβ annotations were aggregated, as described in [29]. For the GEMs with only one chain annotated, the other chain was imputed from other clonotypes (sharing the same chain) only if one single, non ambiguous match was found (for the missing chain). If no match was found, the clonotype was defined by only the α- or the β-chain available. Finally, some clonotypes contain multiplets

of α- or the β-chain. In this case, only the most abundant chain was selected to represent the clonotype.

Data filtering

The raw 10x dataset was filtered using ATRAP [29] to remove noisy observations. ATRAP consists of different types of filters that can be applied to single-cell immune profiling data to reliably identify TCR-pMHC interactions. The method handles multi-omics single-cell sequencing data generated from a multiplexed multimer binding platform such as 10x Genomics immune profiling. The accepted inputs include single-cell RNA sequencing, targeted T cell receptor sequencing, dCODE-Dextramer sequencing for DNA barcoded pMHC multimers, as well as CITE-seq sequencing of DNA barcoded cell hashing antibodies. The method includes the following major steps as described in [29]:

Step 1: *Correction of 10x annotated clonotypes.* Instead of limiting clonotypes to groups of GEMs with exact nucleotide sequence identity, clonotypes were defined based on VJαβ-gene annotation and the CDR3αβ amino acid sequences. Clonotypes for GEMs containing only one TCR chain were imputed if the chain matched only one pre-established clonotype. GEMs containing multiple chains were annotated by the most abundant chain by UMI count.

Step 2: *Filtering based on data-driven thresholds.* For clonotypes consisting of more than 10 GEMs, the expected target is identified if a pMHC has significantly higher UMI distribution than other pMHCs also captured in GEMs of the given clonotype. Significance is tested by Wilcoxon, $\alpha = 0.05$. The pMHCs not declared as target are considered background noise. An accuracy score was obtained based on the fraction of target pMHCs over background pMHCs. The optimal UMI threshold was selected as the UMI value that maximized the accuracy score.

Step 3: *Match pMHC HLA allele with donor haplotype.* The HLA-A, -B, and -C haplotypes were provided by an application note following the release of the single-cell sequencing of the four healthy individuals. Since the samples were sequenced individually the haplotypes were easily added to the data sets. GEMs consisting of mismatch between donor haplotype and pMHC were discarded.

Step 4: *Selecting GEMs with paired aβ chains.* GEMs with only a single chain were removed. For GEMs with multiple α- and/or β-chains, the ones with the highest UMI counts were assigned to each GEM.

Step 5: *Filtering specificity singlets.* If a TCR-pMHC pair was only observed once, it was discarded to increase confidence in matches.

Step 6: *Selecting 10x annotated cells.* Application of the 10x provided filter "is_cell" [28].

Benchmark

The impact of above-mentioned filters was compared to the ICON framework. ICON was applied on the public 10x data sets and the result thereof was provided by the authors via the publication [86]. The annotation for each GEM between the two approaches was traced per donor via the 10x barcode, omitting the well suffix of the barcode. The two approaches were compared based on the number of retained GEMS, accuracy, average binding concordance across clonotypes, and AUC of kernel similarity scores.

The fraction of retained GEMs quantifies how many observations were removed by a filter. Accuracy measures the proportion of GEMs where highest abundance pMHC annotation corresponded to the expected target of large clonotypes (>10 GEMs).

Binding concordance is defined per clonotype as the distribution of GEMs annotated with varying pMHCs, as described in [29]. In a clonotype, the more GEMs annotated with the same pMHC, the larger the concordance for that specificity. The average concordance is a single measure of how much cross-binding the full data contains. The last metric used to compare the two filtering approaches is the AUC on kernel similarity scores, as described in [29]. Kernel similarity scores [107] were computed for sets of TCRs binding the same pMHC (intra-specificity) and sets of TCRs binding different pMHCs (inter-specificity). Under the assumption that TCRs with same specificity have higher intra-similarity than inter-similarity, an AUC value was obtained considering intra-specificities as true positive observations and inter-specificities as true negatives [29]. This AUC metric quantifies how well TCRs

sharing the same specificity can be separated by TCRs with binding different epitopes.

TCR Specificity Prediction

In order to quantify the benefit of removing noisy observations from the original 10x dataset, we trained the NetTCR-2.1 CDR3$\alpha\beta$ model [70] on *i*) the unfiltered 10x data, *ii*) the ATRAP-filtered data using optimal UMI threshold and donor HLA matching, *iii*) ICON-filtered data, with the setup recommended by the authors [86]. A set of positive training TCR-peptide pairs was built from the raw and filtered datasets. For each clonotype, the most frequent pMHC across GEMs (for that clonotype) was selected to be the target pMHC. To validate the trained models, an external evaluation set was retrieved from VDJdb [24]. This dataset consisted of 927 TCR sequences relative to 4 epitopes (GILGFVFTL, GLCTLVAML, ELAGIGILTV, IVTDFSVIK). Also, the training set was restricted to the set of 4 peptides, to ensure overlap between the training and evaluation set. For both data sets, negative peptide-TCR pairs were artificially generated by pairing the positive TCRs with the other 3 peptides different from their target cognate. To investigate performance inflation due to a similarity overlap between training and evaluation sets, TCRs from the evaluation data that had a kernel similarity value above 0.9 to the training TCRs were removed. The training set was randomly split into 5 partitions and the models were trained using 5-fold nested cross-validation. The resulting trained models were used in an ensemble to get predictions over the TCRs in the evaluation set.

Results

Summary of the public 10x data

In the public data set made available by 10x Genomics, a total of 181,913 GEMs were detected containing at least one TCR-pMHC pair. The data set is the result of screening $CD8^+$ T cells from four healthy donors against a panel of 50 pMHC DNA barcode-labeled multimers. Donors were selected by HLA haplotype to ensure overlap with the HLA alleles of the pMHC panel. 44

of the multimers contain antigenic peptides derived from CMV, EBV, influenza, HTLV, HPV, HIV and known cancer antigens. It should be noted that the donors were all seronegative for HIV, HBV, and HBC. The remaining six multimers contained negative control peptides restricted by five HLAs, selected without further elaboration or reasoning. The specificities from each of the four donors were screened in parallel i.e., of four different experimental runs. Therefore, unique GEM-specific 10x barcodes (GEM barcodes) were in some cases observed in replicas across runs. In order to distinguish these evidently distinct GEMs, an extra suffix was added denoting the donor (sample ID). The unfiltered output is portrayed in Figure 6.1, which clearly demonstrates the issue of noise, as every GEM contains multiple pMHCs. Most GEMs contain TCRs annotated with a unique α- and β-chain, however, 10% are annotated with multiple α- or β-chains, which further challenges the investigation of specificity.

Alignment of ICON- and 10x-assigned GEMs reveals inconsistent annotations

In order to compare the ICON and ATRAP filtering frameworks, the outputs from each method were aligned based on the GEM barcode, consisting of 16 nucleotides, a suffix pertaining to the sequencing well, and a sample ID suffix. ICON reported retention of 53,062 GEMs out of the total set of 181,913 GEMs. However, ICON only contains 5031 GEMs that match the original data based on the full GEM barcode, due to inconsistencies in the suffix annotation. When stripping the barcode down to only the 16 nucleotides, we were able to align 39,806 GEM barcodes, as exemplified in Figure 6.2a. We also observed inconsistencies of TCRαβ annotations in 3391 GEMs, as illustrated in Figure 6.2b+c. 1854 GEMs were missing either an α- or a β-chain in the 10x data, but not in the ICON set, while 1537 GEMs were fully annotated, but had inconsistent TCR annotations between ICON and the 10x data. The inconsistencies in TCRαβ annotations may have arisen from imputations based on the 10x-provided clonotype summary. However, imputation is risky because the same CDR3 may form part of several different clonotypes. The example given in 6.2b represents an imputation likely based on the CDR3β sequence. In

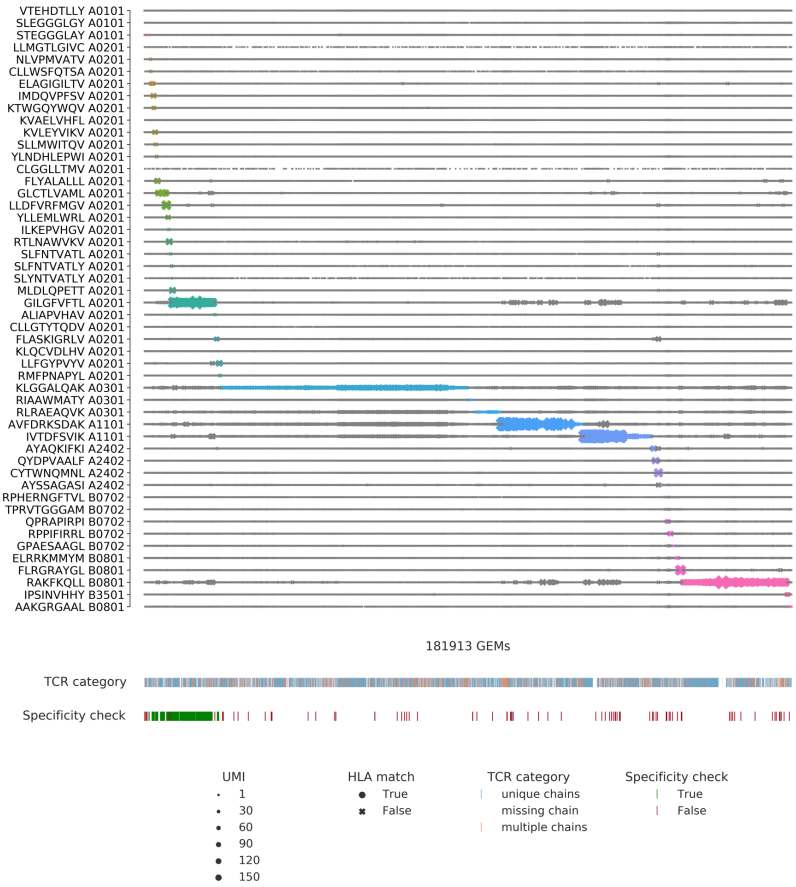


Figure 6.1: Scatterplot of all detected pMHC barcodes (y-axis) within each of the 181,913 GEMs (x-axis). In each GEM the most abundant pMHC is marked by a color, while the remaining pMHCs in the GEM are gray. The marker size reports the UMI count of the given pMHC and the shape recounts whether the HLA allele of the pMHC matches the HLA haplotype of the donor, which is provided in the experimental report [28]. The first color bar indicates the type of TCR chain annotation; whether the TCR has a unique αβ-pair, is missing a chain, or consists of multiple chains. The second color bar is a specificity check against the specificity databases IEDB [23] and VDJdb [24]. Colors highlight the GEMs where the CDR3αβ sequences are contained in the databases. The green color represents a match between the database pMHC and the detected pMHC, while red indicates a mismatch.

this example the CDR3β sequence is part of 42 distinct 10x clonotypes, all carrying the same CDR3β sequence, but paired with different CDR3α sequences. The same case is made for 6.2c and all the other inconsistent GEMs. Imputation by 10x clonotypes is further made difficult as their clonotype definition actually allows multiple α- or β-chains in one clonotype, perhaps a reflection of incomplete allelic exclusion. Thus, 116 of the fully annotated GEMs with mismatching TCRαβ annotations between ICON and 10x can be explained by a switch from one chain to the other, still within the same clonotype definition. This non-conformity has challenged the benchmark, however, we have proceeded assuming that there is a reasonable, however undocumented, explanation for their GEM assignments.

ATRAP - Revisiting clonotype assignment

Of the complete data provided by 10x Genomics, we initially reduced the set to only include IUPAC encoded amino acids within CDR3 sequences and further only considered GEMs which contained both TCR and pMHC annotations, resulting in 181,913 GEMs. Redefining 10x clonotypes resulted in 76,627 unique combinations of V, J genes and CDR3 sequences from α and β chains. Of these clonotypes, 1151 were represented by 10 or more GEMs, and for 1107 of them we were able to annotate an expected binder. The derived optimized UMI thresholds set a cutoff at a minimum UMI of 5 for any pMHC. For pMHC multiplets, the most abundant pMHC must be 1.2 times greater in UMI counts than the second most abundant pMHC. A minimum of 1 UMI is required for TCR α- and β-chains. By this filter, the data set is reduced to 91,652 GEMs and 27,925 unique clonotypes. Additionally, filtering on matching HLA serves as the recommended minimum of filters for ATRAP.

The optimized ATRAP threshold on UMI counts

Of the complete data provided by 10x Genomics, we initially reduced the set to only include IUPAC encoded amino acids within CDR3 sequences and further only considered GEMs which contained both TCR and pMHC annotations, resulting in 181,913

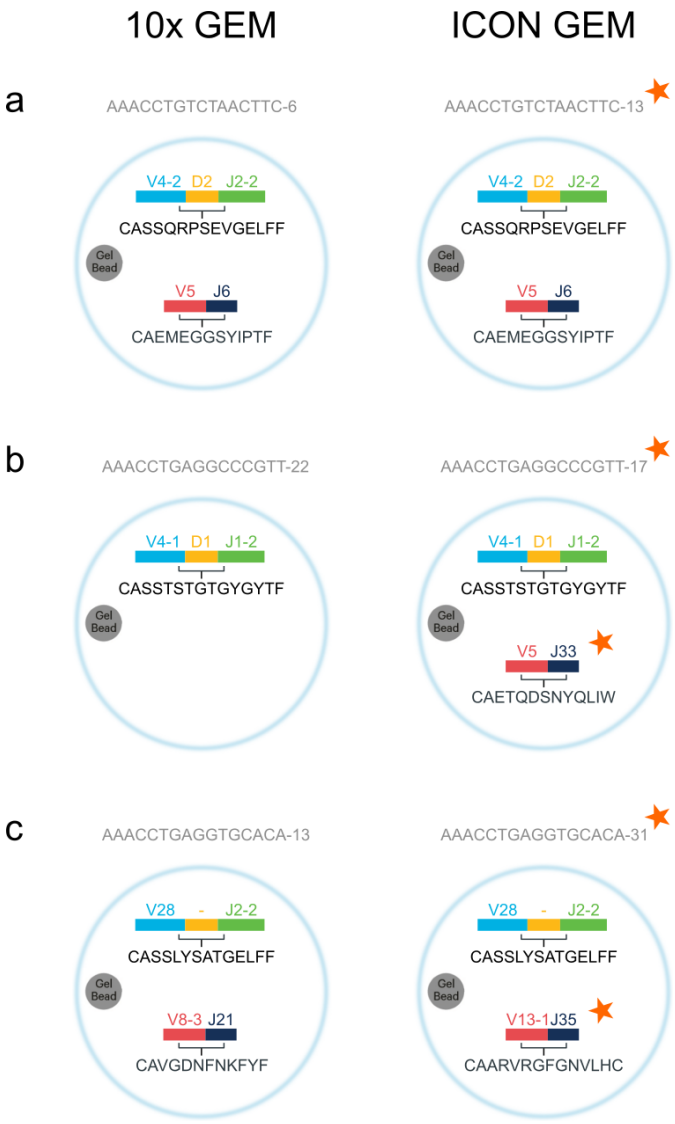


Figure 6.2: Illustrations of annotation inconsistencies. The figure shows examples of GEMs and their TCR annotations from 10x and ICON, respectively. The observed inconsistencies are grouped into three major groups. The inconsistencies are highlighted with a red star in each group. (a) 33,342 GEMs were mapped from the ICON set with inconsistent GEM barcode suffixes. Mapping was based on the GEM barcode nucleotide sequence and TCR annotations. (b) 1854 GEMs were missing either an α- or a β-chain in the 10x data, but not in the ICON set. (c) 1537 GEMs were fully annotated, but the TCR annotations were inconsistent between ICON and the 10x data.

GEMs. Redefining 10x clonotypes resulted in 76,627 unique Vαβ, Jαβ and CDR3αβ combinations. Of those clonotypes, 1151 were represented with 10 or more GEMs, and for 1107 of them we were able to annotate an expected binder. The derived optimized UMI thresholds set a cutoff at minimum 5 UMI for any pMHC. For pMHC multiplets, the most abundant pMHC has to be at least 1.2 times greater in UMI counts than the second most abundant pMHC. A minimum of 1 UMI is required for TCR α- and β-chains. By this filter, the data set is reduced to 91,652 GEMs and 27,925 unique clonotypes. Additionally, filtering on matching HLA serves as the recommended minimum of filters for ATRAP.

Benchmark of ICON and ATRAP

The two filtering frameworks were benchmarked on four metrics, as described by Povlsen et al. [29]: fraction of retained GEMs, accuracy of specificity, average binding concordance across all clonotypes, and AUC based on CDR3αβ similarities. Accuracy is computed as the fraction of GEMs where the most abundant pMHC (by UMI counts) corresponds to the expected binder of a clonotype. An expected binder is defined for each clonotype as the pMHC which is distributed with a mean UMI count significantly higher (Wilcoxon, $\alpha = 0.05$) than the other pMHCs detected as binders for the given clonotype. Binding concordance is computed as the fraction of GEMs within a clonotype that binds a given pMHC and describes the dispersion of pMHC annotations within the clonotype. In a data set where no cross-reactivity is expected, the average binding concordance should be 100%. Finally, the similarity between two TCRs is defined as the summed score of the pairwise CDR3α and CDR3β similarities each calculated using the kernel similarity method described in Shen et al. [107]. The AUC metric is computed based on the hypothesis that different TCRs binding the same pMHC (intra-specificity) are more similar to each other than to TCRs of other specificities (inter-specificity). The performance metrics are presented in Figure 6.3.

The metrics presented in Figure 6.3 reveal good performance from both frameworks. Figure 6.3a+b show the distribution of similarity scores between intra- and inter-specificity TCRs for each

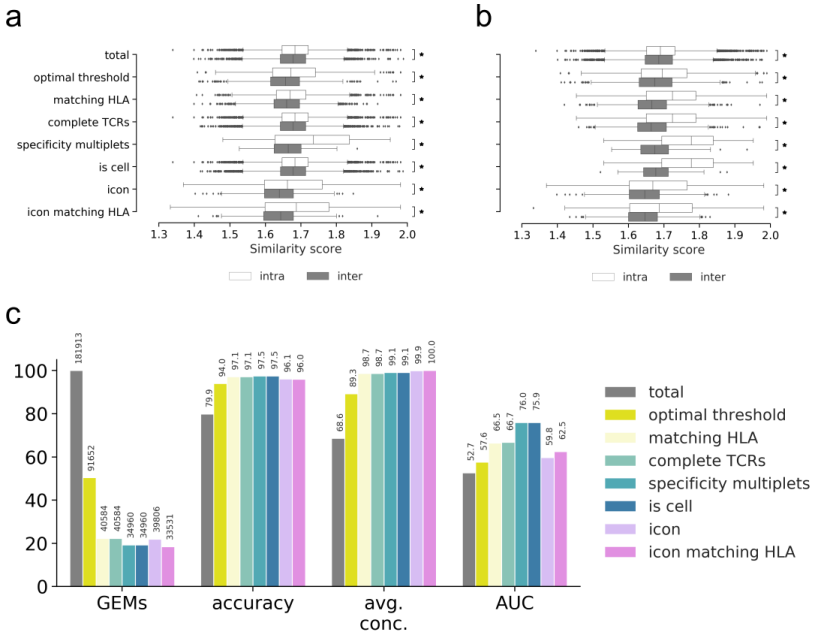


Figure 6.3: Performance metrics for evaluating the filtering steps of ATRAP with ICON. The ATRAP filtering steps consist of total (raw, unfiltered data), optimal threshold obtained from grid search, matching HLA, complete TCRs with a unique set of α- and β-chain, specificity multiplets i.e., TCR-pMHC pairs observed in two or more GEMs, and "is_cell" defined by 10x Genomics Cellranger. ICON yields a single output, however, an addendum has been made to also filter ICON output on HLA match between pMHC and HLA haplotype of the donor. (a) The boxplots show kernel similarity scores between CDR3β sequences of intra- (white) and inter- (dark) specificity for each of the filtering steps. A significant difference (Wilcoxon, $\alpha = 0.05$) of mean between inter- and intra-specificity is marked with an asterisk to the right (b) Here the boxplots show the cumulative effect of ATRAP filters on similarity scores. (c) Performance is measured and summarized by a number of metrics: ratio of retained GEMs (GEMs), accuracy defined by the proportion of GEMs where most abundant pMHC matches the expected binder (accuracy), average binding concordance (avg. conc.) and AUC of similarity scores (AUC). The ATRAP filters are also here cumulatively added to show increasing improvement in performance.

filtering step. Figure 6.3a shows the individual effects of each filter, revealing that filtering specificity singlets away to only retain specificity multiplets yields the greatest separation between intra- and inter-specificity distributions of all filtering steps. We define a

specificity singlet as a TCR-pMHC pair only detected with a single GEM, which makes the pairing more susceptible to artifacts. The combined effect of each filter is visualized in Figure 6.3b, which clearly shows how the separation of inter- and inter-specificity improves as more filters are applied. To quantify the separation of distributions, we compute an AUC score from the principles that perfect intra-specificity scores are close to a maximum value of 2, while inter-specificity resembles completely different TCRs of similarity close to 0. The exact numerical values of the individual specificities are not of interest and they do not affect the AUC. Note that AUC here does not translate into a predictive performance, but rather reflects the extent to which intra-similarity can be distinguished from inter-similarity values.

The summary of both filtering frameworks across our selected performance metrics is presented in Figure 6.3c. Both ICON and the combined ATRAP filters discard a large number of GEMs. The recommended filtering steps for ATRAP consist of filtering on UMI thresholds and matching HLA between annotated pMHC and HLA haplotype of the donor, which yields 40,584 GEMs, which is slightly more than ICON (39,806). Filtering away specificity singlets only removes 5624 GEMs extra, but yields a gain in AUC, as we also saw in 6.3a+b. However, many of those GEMs represent unique clonotypes, so this filter also vastly reduces the total number of clonotypes.

As mentioned, ICON does not discard GEMs based on HLA match between pMHC and donor haplotype. However, we have tested the impact of adding that filter to ICON, which reduces the yield to 33,531 GEMs. The performance measured by accuracy and average concordance is generally very high. ICON scores almost perfect binding concordance at every clonotype, as this was the task it was essentially designed for. Hence, we assume that the corrections of pMHC UMI counts and imputations of CDR3 sequences play a major role in this result. However, the slightly lower AUC of similarity scores of ICON suggest that some imputations might have been incorrect. Based on the AUC of similarity scores, the ATRAP-filters yield a slightly better performance, however, ICON yields specificity annotations of very little ambiguity, where each

clonotype is assigned to only one pMHC.

Visual inspection of ICON and ATRAP outputs

The differences in binding concordance between ATRAP and ICON are clearly visualized in Figure 6.4 and Figure 6.5. Figure 6.4 presents the ATRAP-filters of UMI threshold, HLA matching, and complete TCRs i.e., unique pairing of α- and β-chain.

With an average binding concordance of 98.7, we observe 407 GEMs with a binding concordance <50%, which we will refer to as outliers. A substantial proportion of these cross-binding events are across different HLA alleles. This contradicts the prevailing belief that T cells are restricted to the HLA for which they were positively selected during maturation. We thus suspect that some of these events are a result of random capture of ambient multimer barcode.

In 65 GEMs of the 407 outliers, an expected pMHC target had not been identified, due to the small sizes of the clones. Of the remaining 320 outliers, 76 GEMs exhibit a pattern that aligns with potential cross-reactivity.

Typically a TCR will have a single, preferred target while allowing binding of other pMHCs to a lesser extent, i.e. clones of a clonotype may display a single dominant pMHC response of high binding concordance with few smaller responses of low binding concordance. For the clonotypes of these 76 GEMs, the dominant high-concordance pMHC coincides with the expected target of the individual clonotypes. In 18 of these GEMs, the corresponding clonotypes showed divergent HLA restriction between the annotated low-concordance pMHC and the expected target for the given clonotype. In all of the 76 GEMs, the expected target was detected albeit at a lower UMI count than the annotated pMHC.

The remaining set of 266 GEMs consists of 80 clonotypes exhibiting highly dispersed binding to many different pMHCs, all with low binding concordance. All of these GEMs also contain multiplets of pMHCs. Based on these observations, we conclude that the majority of the 407 outliers are likely artifacts that have es-

caped the ATRAP filtering steps and thus not true cross-binding events.

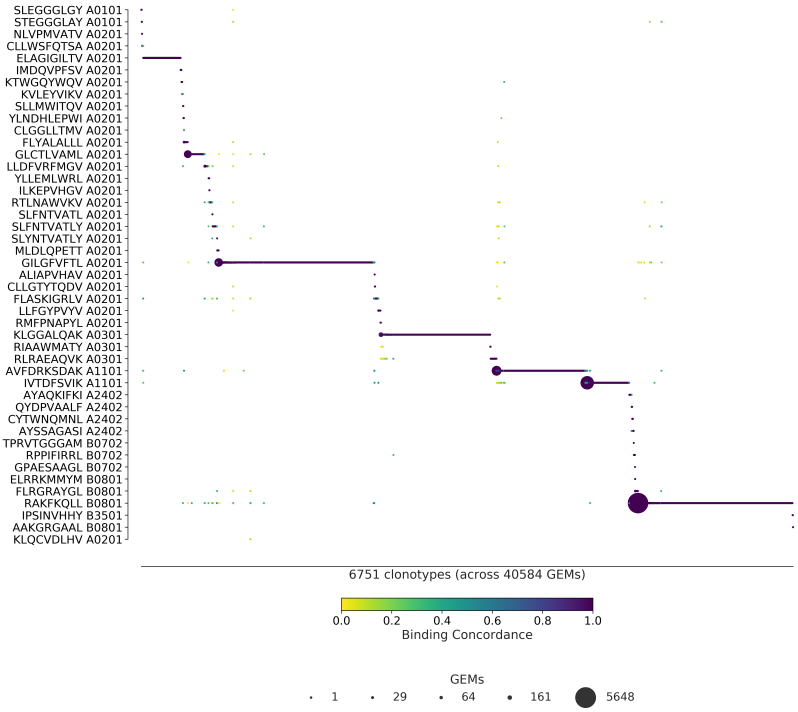


Figure 6.4: ATRAP derived specificity per clonotype. ATRAP-filters consist of UMI threshold, HLA matching, and complete TCRs i.e., a unique pairing of α- and β-chain. The library peptides are listed on the y-axis and each clonotype is represented on the x-axis. Below the x-axis is annotated the total number of clonotypes and GEMs in the presented data. The marker size shows the number of GEMs supporting a given specificity. The color indicates the binding concordance which is calculated as the fraction of GEMs within a clonotype that supports a given pMHC. The higher the concordance, the larger the fraction of supporting GEMs.

Figure 6.5 presents the ICON retrieved specificities. With an average binding concordance of 99.9%, most clonotypes are paired with a single specificity, and only 24 GEMs are categorized as outliers. 13 of the outliers are annotated with a pMHC that does not match the allele of the donor. 4 of the outliers contain CDR3 sequences that differ from the 10x annotation and may be a result

of imputation.

Finally, a key difference between the two methods is that ATRAP retains 45 pMHCs from the staining whereas ICON retains 34 pMHCs. The 11 peptides retained by ATRAP and not ICON elicit small and few responses, but are primarily not involved in cross-binding events. With both filtering frameworks, the largest responses are toward KLG HLA*A-03:01, RKA HLA*B-08:01, and GIL HLA*A-02:01. ICON retains more GEMs and more clonotypes within these peptides, at the expense of other specificities, than ATRAP does.

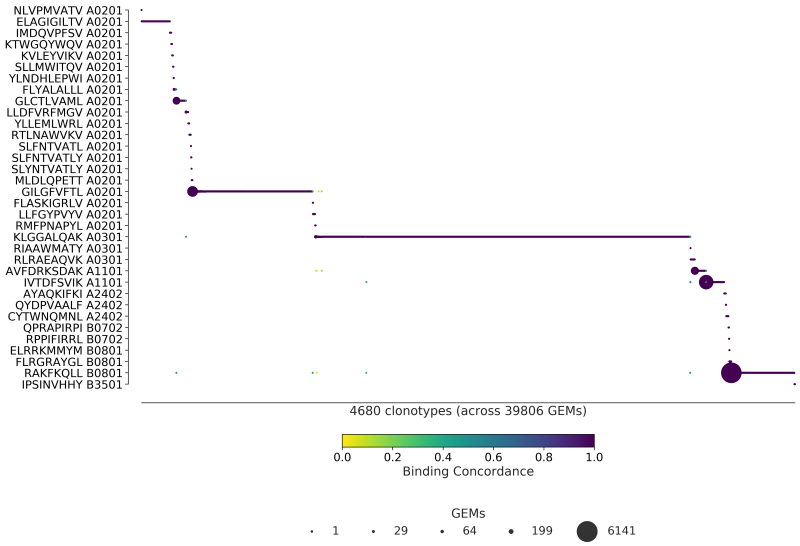


Figure 6.5: ICON derived specificity per clonotype. The library peptides are listed on the y-axis and each clonotype is represented on the x-axis. Below the x-axis is annotated the total number of clonotypes and GEMs in the presented data. The marker size shows the number of GEMs supporting a given specificity. The color indicates the binding concordance which is calculated as the fraction of GEMs within a clonotype that supports a given pMHC. The higher the concordance, the larger the fraction of supporting GEMs.

Predicting TCR specificity with ATRAP- and ICON-filtered data

To quantify the potential predictive performance gain derived from filtering the raw TCR data, we trained NetTCR-2.1 [70] on the

raw 10x data and on the ICON and ATRAP-filtered datasets. Note that the data split for training was done randomly for the three data sets, likely inflating the reported cross-validation performance. We evaluated the performance of the three models on an independent dataset derived from VDJdb [24]. The evaluation set consisted of 927 positive TCRs relative to the 4 peptides in consideration. The number of positive TCRs for each peptide was distributed as follows: 649 for GILGFVFTL, 213 for GLCTLVAML, 57 for ELAGIGILTV 57 and 8 for IVTDFSVIK. To ensure the least possible overlap between training and evaluation set, TCRs from the evaluation set with more than 0.9 kernel similarity to training TCRs were removed. After this filtering, the number of positives was reduced to 219, 122, 46, and 6, respectively.

The results of the experiment are shown in Figure 6.6. The cross-validation performance refers to the performance on the concatenated test sets while the predictions on the evaluation set were calculated as an ensemble of the predictions of the 20 trained models. For the evaluation predictions, we reported the AUCs on the full evaluation set (middle panel) and on the similarity reduced set (lower panel). For each trained model, the AUC was reported on a peptide level. An overall performance value was also given by averaging AUCs across peptides. We reported the average AUCs both as a mean value of the AUCs from each peptide and as a weighted average of the peptide AUCs, weighted by the number of positive TCRs for that specific peptide in the dataset. Both in cross-validation and on the external data, the models trained on ICON and ATRAP datasets outperformed the models trained on unfiltered data. Interestingly, the ICON outperformed ATRAP on almost all the peptides in cross-validation. This can be explained by looking at the similarity between the test and training partitions in cross-validation. Figure 6.7 shows, for each peptide, the distribution of kernel similarities between the positive TCRs in the test set and their nearest neighboring positive TCR in the training. For the GIL and IVT peptides, ICON has a higher median similarity between training and test set, leading to a higher AUC value in cross-validation for these two peptides.

In the external evaluation, the models trained with ICON and

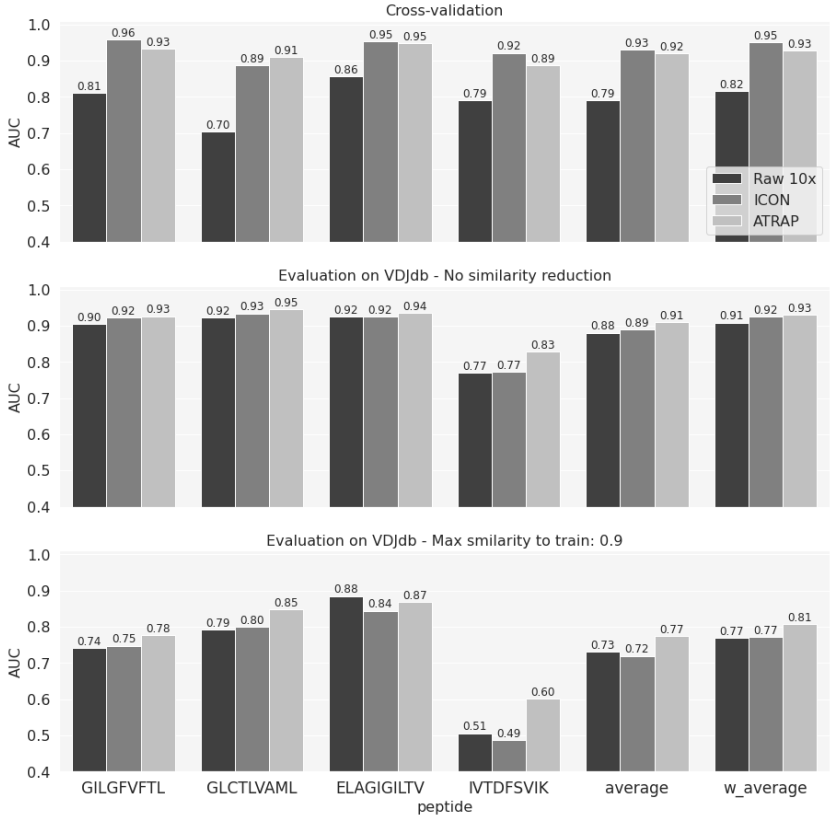


Figure 6.6: Performance of NetTCR-2.1 in terms of AUC on the raw 10x data and on the filtered datasets. The AUC is given on the concatenated test sets from cross-validation and on the external evaluation set from VDJdb (before and after removing evaluation TCRs similar to sequences in the training set). "average" refers to the mean of the AUC values across peptides; "w_average" is a weighted average of AUCs across peptides, weighted by the number of positive TCRs for the peptides in the dataset in consideration.

ATRAP datasets showed better performance (except for the ELA peptide) compared to the one trained on the raw data. Furthermore, the models trained on ATRAP-filtered data generalize better on the external dataset, outperforming ICON across all peptides. For the similarity-reduced evaluation set, all the differences in AUC between ICON and ATRAP are significantly different for all peptides except IVT ($p - \text{value} < 0.05$, bootstrap test on the

AUC with 100 repetitions). This is also confirmed by the improved average and weighted average performance of the model trained on ATRAP data. Furthermore, the gap in performance between the two methods increases when the overlap between the training and evaluation set is reduced.

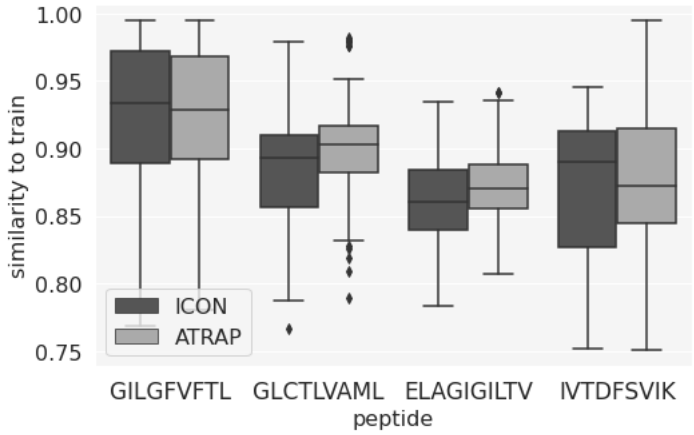


Figure 6.7: Kernel similarity values between positive TCRs in the test set and their nearest neighbor in the positive set of training TCRs. For each test TCR, the similarity to training is calculated as the minimum similarity between the test TCRs and all the TCRs in the training set. The similarity distributions for ATRAP- and ICON-filtered dataset are shown.

Discussion

Single-cell screening assays may pave the way for a better understanding of T cell specificity. The technology enables the study of binders, decisive non-binders and even cross-binding. However, de-noising single-cell specificity data is a critical bottleneck in studying T cell specificity. Here, we evaluate two methods, ATRAP and ICON, both aiming to resolve this bottleneck, filtering noise and putative artifacts from true binding events. Since no golden standard exists, the methods are evaluated via metrics designed for the purpose.

The two filtering frameworks both show very good performance, but with substantially different advantages and disadvantages.

ICON excels at reducing ambiguous specificity annotations, such that the majority of clonotypes are annotated with exactly one pMHC target. The efficient reduction of outliers may, however, also become a hindrance to detecting cross-reactivity. The ATRAP method includes more GEMs across more pMHCs. A larger proportion of GEMs represent binding events that resemble cross-reactivity, although, after careful scrutiny, the majority of these are noisy observations having escaped filtering.

The filtering frameworks were evaluated on four metrics: retention of GEMs, binding accuracy guided by expected targets, average binding concordance, and AUC of kernel similarity scores. ATRAP achieves the highest accuracy score. However, binding accuracy may be a biased metric in this context as ATRAP was specifically designed to maximize this score. Similarly, we see ICON showing superior average binding concordance, favoring low dispersion of specificity within a clonotype, which ICON was purposefully designed to reduce. The AUC of kernel similarity scores is the only method-independent metric, which however does not account for outliers, in favor of ATRAP.

Each framework has a set of requirements for the method to work optimally. ATRAP heavily relies on cell hashing, where HLA typing of donors is known, to validate specificities. In contrast, ICON relies on gene expression data to remove duplicates and negative control pMHC multimers to correct binding signals of positive pMHCs. The impact of gene expression data was previously tested for ATRAP, which showed only minute added performance [29]. Due to the low impact and the high expense of running gene expression sequencing, this filtering step was deprioritized. Cell hashing, of course, also confers an additional cost; however, it further enables the study of immunodominant epitopes and individual T cell repertoires. The use of negative control pMHCs allows ICON to set a cutoff for pMHC UMI counts, similar to the accuracy optimizing threshold in ATRAP. The weakness of negative control pMHCs is that no one can yet define true negative targets. To circumvent this, utilizing empty multimer scaffolds containing only the DNA barcode as negative controls would reveal the level of ambient barcodes polluting the assay without risking rare but

true binding.

Both frameworks assume that the pMHC UMI count reflects the likelihood of a TCR-pMHC pair, and use the count either directly (ATRAP) or corrected and normalized (ICON) to filter away GEMs. However, it is important to note that the UMI count actually refers to the number of pMHC multimers captured together with a T cell in a GEM. The count may be affected by the extent of ambient multimers, T cell expression of TCRs, and binding affinity. Thus to improve the filtering strategies of ATRAP or ICON future methods may implement adjusted TCR-pMHC pairing scores.

Pairing of TCR and pMHC is further made difficult in the cases where a presumed single cell expresses two different α- or β-chains. The dual expression cannot simply be written off as capture of multiple cells, as multiple GEMs exhibit the same dual TCR profile, and is a known phenomenon [108–110]. Neither ICON nor ATRAP seeks to investigate the impact on specificities, but simply annotate the most abundantly expressed chain. To improve specificity detection, this aspect should be investigated further. Moreover, CDR3 α and beta β are not unique, but exist in various combinations, despite the stochastic process under which they are produced. Therefore, imputing CDR3 chains for GEMs with either multiple chains or GEMs missing a chain, will often not result in a unique pairing. We speculate that ICON has attempted this since we have observed discrepancies in CDR3 annotations between 10x and ICON. The comparison was further complicated by inconsistent GEM barcodes between ICON and the 10x data. The alteration of barcodes is unaccounted for by the authors of ICON.

To quantify how the two filtering approaches increase the signal-to-noise ratio, we trained NetTCR-2.1 on *i*) the raw 10x dataset, *ii*) the ATRAP-filtered data (using UMI threshold and HLA matching criteria), and *iii*) ICON-filtered data. The results showed that both ICON and ATRAP-filtered data sets lead to improved performance, compared to the raw 10x data. This further confirms that both methods filter out artifacts from the datasets, increasing the signal-to-noise ratio. The two models performed comparably

in cross-validation, However, ATRAP demonstrated better generalizability compared to ICON on a novel set of TCRs, independent from the training data.

In conclusion, both ICON and ATRAP successfully remove potential artifacts from the 10x dataset. Overall, the two frameworks perform on par. ICON provides high specificity at the expense of sensitivity, whereas ATRAP provides high sensitivity to allow detection of cross-binding events, but at the expense of specificity.

CHAPTER 7

Epilogue

The research projects presented in this thesis were centered around the investigation of TCR-pMHC recognition. The main scope was to build deep learning models capable of learning patterns from the input sequences and predict whether a given TCR would recognize a specific pMHC complex, triggering an immune response.

In the first project, we developed NetTCR-2.0, a neural network-based model to predict interactions between TCRs and pMHC complexes. NetTCR uses convolutional neural networks to scan the input TCR and peptide and extract latent features from the sequences. This learned representation is subsequently used to predict whether the TCR would recognize the epitope in consideration. We focused our study mainly on two aspects, namely input selection and data curation. The TCR is a heterodimer, consisting of a α and β chains. Traditionally, β chain was thought to be the main driver of the interaction between the TCR and the pMHC. For this reason, the majority of the available models were trained on CDR3β only data. We investigated the impact of adding the CDR3α as input to the model and showed that the inclusion of the α chain leads to improved performance over the model trained only on the β data, across all the analyzed peptides. These results were validated on a set of TCRs generated in-house. Additionally, we demonstrated that a considerable amount of TCRs for each peptide is needed to build a reliable classifier; we quantified this

number to be approximately 150 positive TCRs. If the dataset contains very few observations regarding the TCRs binding to a specific epitope, a model trained on such data will be exposed to very few examples of binding for this epitope and will not be able to extract some rules for generalization. The second key point of the study was to show that a proper dataset curation in terms of data redundancy is needed before training any models. We approached this problem by removing from the dataset TCRs with high sequence similarity, and splitting the data in training, validation and test making sure that these partitions did not have any overlap of TCRs in terms of similarity. Such overlap would produce an inflated performance due to data leakage between training and test sets.

The second research project naturally followed the first. In this study, we investigated some fundamental aspects involved in the development of a TCR-pMHC predictor. Rather than comparing our model to other existing tools, we trained NetTCR-2.1 to address these questions and define a set of suggested rules that could help the research community. We investigate whether a peptide- or a pan-specific approach serves best to model the data and we quantify the impact of integrating the CDR1 and 2 loops in the model. In our study, we also address some of the challenges involved in building a suitable dataset, namely data redundancy and negative data generation. Based on the currently available data, our study concludes that TCR specificity is best modeled using peptide-specific approaches, integrating information from all 6 CDR loops, and with negative data constructed from a combination of true and mislabeled negatives.

Lastly, we proposed TCRbase, a straightforward similarity-based model to predict TCR specificity, showing that such a modeling approach achieves good performance while being very simple. It is highly advisable to include such a baseline model in the evaluation of machine learning-based models to appreciate the improved generalizability of the more complex neural network models. We demonstrated this by showing the performance of both NetTCR and TCRbase as a function of the ”distance” to training data. If the training and test data are allowed to be similar in sequences,

then the two models' performances are comparable; as the similarity is reduced, TCRbase suffers a drop in performance, while NetTCR is able to maintain its predictive power. This is an indication that the neural network has learned some features from the inputs beyond sequence similarity.

The first two projects proposed different models aiming to predict TCR-peptide interactions. These machine learning models were trained on publicly available datasets. One of the main challenges involved in the development of these models was related to the data, specifically data scarcity and data quality. The first major limitation was that the vast majority of the data referred to one MHC molecule, the HLA-A*02:01, as it is one of the most frequent in humans. Moreover, only for a few epitopes the amount of data was sufficient to train a model. Secondly, the quality of the data is uncertain, as bulk-sequencing of TCRs paired with their target epitope is prone to wrong annotations. Furthermore, bulk techniques allow sequencing of only one of the two chains of the TCRs.

Single-cell sequencing technologies hold the promise of addressing the above-mentioned data-related problems, enabling the generation of large amounts of paired TCR data, while ensuring high data quality. However, T cell characterization through scRNA-seq is a relatively new field and new methods for processing the resulting data are needed.

In the third project, we applied two tools, ICON and ATRAP, on the single-cell 10x dataset aiming to remove potential artifacts and wrongly annotated TCRs, to increase the signal-to-noise ratio. To validate our finding, we trained NetTCR-2.1 on the raw 10x data and on the filtered versions. The model trained on the filtered data outperformed consistently the one trained on the full dataset, confirming that ATRAP and ICON successfully denoised the 10x data. As large single-cell databases will be generated, these data-driven filtering approaches will be a fundamental step in processing future data.

In conclusion, we have shown that it is, to some extent, possible to predict TCR-pMHC interaction. At the moment, however, the capabilities of the existing models are heavily limited by the current

state of the data. In our work, we have proposed fairly simple neural network architectures to accomplish this prediction task. Novel and more complex deep learning models are constantly being developed, enabling the characterization of complex relationships in complex data. However, even these sophisticated models cannot show their true potential without high-quality training data. Given the development of new data generation techniques and the advancement of deep learning modeling frameworks, the future perspectives on T cell epitope prediction are encouraging.

Bibliography

[1] K. Murphy and C. Weaver. *Janeway's Immunobiology*. Ed. by K. Murphy. 9th edition. |New York, NY : Garland Science/Taylor & Francis: Garland Science, 2016.

[2] F. A. Bonilla and H. C. Oettgen. "Adaptive immunity." In: *The Journal of Allergy and Clinical Immunology* 125.2 Suppl 2 (Feb. 2010), S33–40.

[3] D. B. Roth. "V(D)J recombination: mechanism, errors, and fidelity." In: *Microbiology spectrum* 2.6 (Dec. 2014).

[4] R. S. Akondy et al. "Origin and differentiation of human memory CD8 T cells after vaccination." In: *Nature* 552.7685 (Dec. 2017), pp. 362–367.

[5] B. Youngblood et al. "Effector CD8 T cells dedifferentiate into long-lived memory cells." In: *Nature* 552.7685 (Dec. 2017), pp. 404–409.

[6] M. De Simone, G. Rossetti, and M. Pagani. "Single cell T cell receptor sequencing: techniques and future challenges." In: *Frontiers in immunology* 9 (July 2018), p. 1638.

[7] J. M. Volpe and T. B. Kepler. "Large-scale analysis of human heavy chain V(D)J recombination patterns." In: *Immunome research* 4 (Feb. 2008), p. 3.

[8] J. J. Miles, D. C. Douek, and D. A. Price. "Bias in the αβ T-cell repertoire: implications for disease pathogenesis and vaccination." In: *Immunology and Cell Biology* 89.3 (Mar. 2011), pp. 375–387.

[9] K. S. Kobayashi and P. J. van den Elsen. "NLRC5: a key regulator of MHC class I-dependent immune responses." In: *Nature Reviews. Immunology* 12.12 (Dec. 2012), pp. 813–820.

[10] S. Paul et al. "Determination of a predictive cleavage motif for eluted major histocompatibility complex class II ligands." In: *Frontiers in immunology* 9 (Aug. 2018), p. 1795.

[11] A. Sette et al. "Capacity of intact proteins to bind to MHC class II molecules." In: *Journal of Immunology* 143.4 (Aug. 1989), pp. 1265–1267.

[12] K. Deroost and J. Langhorne. “Gamma/delta T cells and their role in protection against malaria.” In: *Frontiers in immunology* 9 (Dec. 2018), p. 2973.

[13] C. Zou et al. “ δT cells in cancer immunotherapy.” In: *Oncotarget* 8.5 (Jan. 2017), pp. 8900–8909.

[14] M. G. Rudolph, R. L. Stanfield, and I. A. Wilson. “How TCRs bind MHCs, peptides, and coreceptors.” In: *Annual Review of Immunology* 24 (2006), pp. 419–466.

[15] M. M. Davis and P. J. Bjorkman. “T-cell antigen receptor genes and T-cell recognition.” In: *Nature* 334.6181 (Aug. 1988), pp. 395–402.

[16] M. Krogsgaard and M. M. Davis. “How T cells ’see’ antigen.” In: *Nature Immunology* 6.3 (Mar. 2005), pp. 239–245.

[17] W. Zhang et al. “A framework for highly multiplexed dextramer mapping and prediction of T cell receptor sequences to antigen specificity.” In: *Science Advances* 7.20 (May 2021).

[18] A. Montemurro et al. “NetTCR-2.0 enables accurate prediction of TCR-peptide binding by using paired TCRα and β sequence data.” In: *Communications Biology* 4.1 (Sept. 2021), p. 1060.

[19] A. K. Sewell. “Why must T cells be cross-reactive?” In: *Nature Reviews. Immunology* 12.9 (Sept. 2012), pp. 669–677.

[20] J. J. Adams et al. “T cell receptor signaling is limited by docking geometry to peptide-major histocompatibility complex.” In: *Immunity* 35.5 (Nov. 2011), pp. 681–693.

[21] K. M. Armstrong, K. H. Piepenbrink, and B. M. Baker. “Conformational changes and flexibility in T-cell receptor recognition of peptide-MHC complexes.” In: *The Biochemical Journal* 415.2 (Oct. 2008).

[22] K. M. Armstrong, F. K. Insaidoo, and B. M. Baker. “Thermodynamics of T-cell receptor-peptide/MHC interactions: progress and opportunities.” In: *Journal of Molecular Recognition* 21.4 (Aug. 2008), pp. 275–287.

[23] R. Vita et al. “The Immune Epitope Database (IEDB): 2018 update.” In: *Nucleic Acids Research* 47.D1 (Jan. 2019), pp. D339–D343.

[24] D. V. Bagaev et al. “VDJdb in 2019: database extension, new analysis infrastructure and a T-cell receptor motif compendium.” In: *Nucleic Acids Research* 48.D1 (Jan. 2020), pp. D1057–D1062.

[25] N. Tickotsky et al. “McPAS-TCR: a manually curated catalogue of pathology-associated T cell receptor sequences.” In: *Bioinformatics* 33.18 (Sept. 2017), pp. 2924–2929.

[26] W. Zhang et al. “PIRD: pan immune repertoire database.” In: *Bioinformatics* 36.3 (Feb. 2020), pp. 897–903.

[27] M. S. Bardi et al. “HLA-A, B and DRB1 allele and haplotype frequencies in volunteer bone marrow donors from the north of Parana State.” In: *Revista brasileira de hematologia e hemoterapia* 34.1 (2012), pp. 25–30.

[28] 10xGenomics. “A new way of exploring immunity: linking highly multiplexed antigen recognition to immune repertoire and phenotype”. In: *Application Note* (2020).

[29] H. Povlsen et al. “ATRAP - Accurate T cell Receptor Antigen Pairing through data-driven filtering of sequencing information from single-cells [Manuscript submitted for publication]”. unpublished. 2022.

[30] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[31] W. McCulloch and W. Pitts. “A logical calculus of the ideas immanent in nervous activity. ”. In: *Bulletin of Mathematical Biology* 52.1-2 (1990), 99–115, discussion 73.

[32] G. Cybenko. “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of Control, Signals, and Systems* 2.4 (Dec. 1989), pp. 303–314.

[33] Y. Bengio and Y. Lecun. “Convolutional Networks for Images, Speech, and Time-Series”. In: (Nov. 1997).

[34] J. L. Elman. “Finding Structure in Time”. In: *Cognitive science* 14.2 (Mar. 1990), pp. 179–211.

[35] S. Hochreiter and J. Schmidhuber. “Long short-term memory.” In: *Neural Computation* 9.8 (1997), pp. 1735–1780.

[36] A. Krogh. “What are artificial neural networks?” In: *Nature Biotechnology* 26.2 (Feb. 2008), pp. 195–197.

[37] C. Hsu et al. “Learning protein fitness models from evolutionary and assay-labeled data.” In: *Nature Biotechnology* 40.7 (July 2022), pp. 1114–1122.

[38] S. Malhotra and R. Walters. “Secondary Protein Structure Prediction Using Neural Networks”. In: *arXiv* (2022).

[39] M. Nielsen and M. Andreatta. “NNAlign: a platform to construct and evaluate artificial neural network models of receptor-ligand interactions.” In: *Nucleic Acids Research* 45.W1 (July 2017), W344–W349.

[40] J. Zhang. “Protein-length distributions for the three domains of life.” In: *Trends in Genetics* 16.3 (Mar. 2000), pp. 107–109.

[41] B. Daş and S. Toraman. “Classifying protein sequences using convolutional neural network”. In: *Bitlis Eren Üniversitesi Fen Bilimleri Dergisi* 9.4 (Dec. 2020), pp. 1663–1671.

[42] V. Jurtz et al. “NetMHCpan-4.0: Improved Peptide-MHC Class I Interaction Predictions Integrating Eluted Ligand and Peptide Binding Affinity Data.” In: *Journal of Immunology* 199.9 (Nov. 2017), pp. 3360–3368.

[43] W. Torng and R. B. Altman. “3D deep convolutional neural networks for amino acid environment similarity analysis.” In: *BMC Bioinformatics* 18.1 (June 2017), p. 302.

[44] J. Y. Lee and F. Dernoncourt. “Sequential Short-Text Classification with Recurrent and Convolutional Neural Networks”. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2016, pp. 515–520.

[45] M. P. Perrone et al. “Optimal Mini-Batch Size Selection for Fast Gradient Descent”. In: *arXiv* (2019).

[46] D. Masters and C. Luschi. “Revisiting Small Batch Training for Deep Neural Networks”. In: *arXiv* (2018).

[47] D. P. Kingma and J. Ba. “Adam: A Method for Stochastic Optimization”. In: *arXiv* (2014).

[48] D. Opitz and R. Maclin. “Popular ensemble methods: an empirical study”. In: *Journal of Artificial Intelligence Research* 11 (Aug. 1999), pp. 169–198.

[49] L. Rokach. “Ensemble-based classifiers”. In: *Artificial Intelligence Review* 33.1-2 (Feb. 2010), pp. 1–39.

[50] A. P. Bradley. “The use of the area under the ROC curve in the evaluation of machine learning algorithms”. In: *Pattern recognition* 30.7 (July 1997), pp. 1145–1159.

[51] D. M. W. Powers. “Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation”. In: *arXiv* (2020).

[52] M. Bassani-Sternberg et al. “Deciphering HLA-I motifs across HLA peptidomes improves neo-antigen predictions and identifies allostery regulating HLA specificity.” In: *PLoS Computational Biology* 13.8 (Aug. 2017), e1005725.

[53] J. Racle et al. “Robust prediction of HLA class II epitopes by deep motif deconvolution of immunopeptidomes.” In: *Nature Biotechnology* 37.11 (Nov. 2019), pp. 1283–1286.

[54] R. A. Koup and D. C. Douek. “Vaccine design for CD8 T lymphocyte responses.” In: *Cold Spring Harbor perspectives in medicine* 1.1 (Sept. 2011), a007252.

[55] R. A. Morgan et al. “Cancer regression in patients after transfer of genetically engineered lymphocytes.” In: *Science* 314.5796 (Oct. 2006), pp. 126–129.

[56] P. A. Ott et al. “An immunogenic personal neoantigen vaccine for patients with melanoma.” In: *Nature* 547.7662 (July 2017), pp. 217–221.

[57] R. W. Hamming. “Error Detecting and Error Correcting Codes”. In: *Bell System Technical Journal* 29.2 (Apr. 1950), pp. 147–160.

[58] V. I. Levenshtein et al. “Binary codes capable of correcting deletions, insertions, and reversals”. In: *Soviet physics doklady*. Vol. 10. 1966, pp. 707–710.

[59] L. Y. Yampolsky and A. Stoltzfus. “The exchangeability of amino acids in proteins.” In: *Genetics* 170.4 (Aug. 2005), pp. 1459–1472.

[60] S. Henikoff and J. G. Henikoff. “Amino acid substitution matrices from protein blocks.” In: *Proceedings of the National Academy of Sciences of the United States of America* 89.22 (Nov. 1992), pp. 10915–10919.

[61] W.-J. Shen et al. “Towards a Mathematical Foundation of Immunology and Amino Acid Chains”. 2012.

[62] U. Hobohm et al. “Selection of representative protein data sets.” In: *Protein Science* 1.3 (Mar. 1992), pp. 409–417.

[63] J. Chen, Z.-R. Xie, and Y. Wu. “Understand protein functions by comparing the similarity of local structural environments”. In: *Biochimica et Biophysica Acta (BBA)-Proteins and Proteomics* 1865.2 (2017), pp. 142–152.

[64] V. Sangar et al. “Quantitative sequence-function relationships in proteins based on gene ontology”. In: *BMC bioinformatics* 8.1 (2007), pp. 1–15.

[65] A. Vaswani et al. “Attention Is All You Need”. In: *arXiv* (2017).

[66] W. D. Chronister et al. “TCRMatch: Predicting T-cell receptor specificity based on sequence similarity to previously characterized receptors”. In: *BioRxiv* (Dec. 2020).

[67] P. Dash et al. “Quantifiable predictive features define epitope-specific T cell receptor repertoires.” In: *Nature* 547.7661 (July 2017), pp. 89–93.

[68] J. Glanville et al. “Identifying specificity groups in the T cell receptor repertoire.” In: *Nature* 547.7661 (July 2017), pp. 94–98.

[69] H. Huang et al. “Analyzing the Mycobacterium tuberculosis immune response by T-cell receptor clustering with GLIPH2 and genome-wide antigen screening.” In: *Nature Biotechnology* 38.10 (Oct. 2020), pp. 1194–1202.

[70] A. Montemurro, L. E. Jessen, and M. Nielsen. “NetTCR-2.1: Lessons and guidance on how to develop models for TCR specificity predictions”. 2022 (Submitted for publication).

[71] E. Lanzarotti, P. Marcatili, and M. Nielsen. “T-Cell Receptor Cognate Target Prediction Based on Paired α and β Chain Sequence and Structural CDR Loop Similarities.” In: *Frontiers in immunology* 10 (Aug. 2019), p. 2080.

[72] J.-W. Sidhom et al. “DeepTCR is a deep learning framework for revealing sequence concepts within T-cell repertoires.” In: *Nature Communications* 12.1 (Mar. 2021), p. 1605.

[73] D. P. Kingma and M. Welling. “An introduction to variational autoencoders”. In: *Foundations and Trends®in Machine Learning* 12.4 (2019), pp. 307–392.

[74] A. Weber, J. Born, and M. Rodriguez Martínez. “TITAN: T-cell receptor specificity prediction with bimodal attention networks.” In: *Bioinformatics* 37.Suppl_1 (July 2021), pp. i237–i244.

[75] P. Moris et al. “Current challenges for unseen-epitope TCR interaction prediction and a new perspective derived from image classification.” In: *Briefings in Bioinformatics* 22.4 (July 2021).

[76] I. Springer et al. “Prediction of Specific TCR-Peptide Binding From Large Dictionaries of TCR-Peptide Pairs.” In: *Frontiers in immunology* 11 (Aug. 2020), p. 1803.

[77] I. Springer, N. Tickotsky, and Y. Louzoun. “Contribution of T cell receptor alpha and beta CDR3, MHC typing, V and J genes to peptide binding prediction.” In: *Frontiers in immunology* 12 (Apr. 2021), p. 664514.

[78] J. Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *arXiv* (2018).

[79] K. E. Wu et al. “TCR-BERT: learning the grammar of T-cell receptors for flexible antigen- binding analyses”. In: *BioRxiv* (Nov. 2021).

[80] S. Gielis et al. “Detection of enriched T cell epitope specificity in full T cell receptor sequence repertoires.” In: *Frontiers in immunology* 10 (Nov. 2019), p. 2820.

[81] E. Jokinen et al. “TCRGP: Determining epitope specificity of T cell receptors”. In: *BioRxiv* (Feb. 2019).

[82] Y. Tong et al. “SETE: Sequence-based Ensemble learning approach for TCR Epitope binding prediction.” In: *Computational biology and chemistry* 87 (June 2020), p. 107281.

[83] J. Glanville et al. “Identifying specificity groups in the T cell receptor repertoire”. In: *Nature 2017 547:7661* 547.7661 (June 2017), pp. 94–98.

[84] P. Dash et al. “Quantifiable predictive features define epitope-specific T cell receptor repertoires”. eng. In: *Nature* 547.7661 (July 2017), pp. 89–93.

[85] W. D. Chronister et al. “TCRMatch: Predicting T-Cell Receptor Specificity Based on Sequence Similarity to Previously Characterized Receptors”. In: *Frontiers in Immunology* 12 (Mar. 2021), p. 673.

[86] W. Zhang et al. “A framework for highly multiplexed dextramer mapping and prediction of T cell receptor sequences to antigen specificity”. In: *Science advances* 7.20 (May 2021).

[87] P. Moris et al. “Current challenges for unseen-epitope TCR interaction prediction and a new perspective derived from image classification”. In: *Briefings in Bioinformatics* 22.4 (July 2021).

[88] J. W. Sidhom et al. “DeepTCR is a deep learning framework for revealing sequence concepts within T-cell repertoires”. In: *Nature Communications 2021 12:1* 12.1 (Mar. 2021), pp. 1–12.

[89] I. Springer, N. Tickotsky, and Y. Louzoun. “Contribution of T Cell Receptor Alpha and Beta CDR3, MHC Typing, V and J Genes to Peptide Binding Prediction”. In: *Frontiers in Immunology* 12 (Apr. 2021), p. 1436.

[90] M. S. Krangel. “Mechanics of T cell receptor gene rearrangement.” In: *Current Opinion in Immunology* 21.2 (Apr. 2009), pp. 133–139.

[91] E. Mahe, T. Pugh, and S. Kamel-Reid. “T cell clonality assessment: past, present and future.” In: *Journal of Clinical Pathology* 71.3 (Mar. 2018), pp. 195–200.

[92] N. R. J. Gascoigne et al. “TCR signal strength and T cell development.” In: *Annual Review of Cell and Developmental Biology* 32 (Oct. 2016), pp. 327–348.

[93] D. Jung and F. W. Alt. “Unraveling V(D)J recombination; insights into gene regulation.” In: *Cell* 116.2 (Jan. 2004), pp. 299–311.

[94] K. J. L. Jackson et al. “The shape of the lymphocyte receptor repertoire: lessons from the B cell receptor.” In: *Frontiers in immunology* 4 (Sept. 2013), p. 263.

[95] M. M. Davis and P. J. Bjorkman. “T-cell antigen receptor genes and T-cell recognition”. In: *Nature* 334.6181 (1988), pp. 395–402.

[96] V. I. Zarnitsyna et al. “Estimating the diversity, completeness, and cross-reactivity of the T cell repertoire.” In: *Frontiers in immunology* 4 (Dec. 2013), p. 485.

[97] Y. Elhanati et al. “repgenHMM: a dynamic programming tool to infer the rules of immune receptor generation from sequence data.” In: *Bioinformatics* 32.13 (July 2016), pp. 1943–1951.

[98] P. Marrack et al. “T cell receptor specificity for major histocompatibility complex proteins”. In: *Current Opinion in Immunology* 20.2 (Apr. 2008), pp. 203–207.

[99] J. Hennecke and D. C. Wiley. “T Cell Receptor–MHC Interactions up Close”. In: *Cell* 104.1 (Jan. 2001), pp. 1–4.

[100] M. Wieczorek et al. “Major histocompatibility complex (MHC) class I and MHC class II proteins: Conformational plasticity in antigen presentation”. In: *Frontiers in Immunology* 8.3 (Mar. 2017), p. 292.

[101] N. L. La Gruta et al. “Understanding the drivers of MHC restriction of T cell receptors”. In: *Nature Reviews Immunology 2018 18:7* 18.7 (Apr. 2018), pp. 467–478.

[102] S. C. Boutet et al. “Scalable and comprehensive characterization of antigen-specific CD8 T cells using multi-omics single cell analysis”. In: *The Journal of Immunology* 202.1 Supplement (2019).

[103] A. M. Klein et al. “Droplet Barcoding for Single-Cell Transcriptomics Applied to Embryonic Stem Cells”. In: *Cell* 161.5 (May 2015), pp. 1187–1201.

[104] E. Z. Macosko et al. “Highly Parallel Genome-wide Expression Profiling of Individual Cells Using Nanoliter Droplets”. In: *Cell* 161.5 (May 2015), pp. 1202–1214.

[105] T. Kivioja et al. “Counting absolute numbers of molecules using unique molecular identifiers”. In: *Nature methods* 9.1 (Jan. 2011), pp. 72–74.

[106] D. S. Fischer et al. “Predicting antigen specificity of single T cells based on TCR CDR3 regions”. In: *Molecular Systems Biology* 16.8 (Aug. 2020), e9416.

[107] W.-J. Shen et al. “Towards a Mathematical Foundation of Immunology and Amino Acid Chains”. In: (May 2012). arXiv: 1205.6031.

[108] J. I. Elliott and D. M. Altmann. “Dual T cell receptor alpha chain T cells in autoimmunity”. In: *The Journal of Experimental Medicine* 182.4 (Oct. 1995), p. 953.

[109] H. T. Petrie et al. “Multiple rearrangements in T cell receptor alpha chain genes maximize the production of useful thymocytes”. In: *The Journal of Experimental Medicine* 178.2 (Aug. 1993), p. 615.

[110] N. J. Schuldt and B. A. Binstadt. “Dual TCR T Cells: Identity Crisis or Multitaskers?” In: *The Journal of Immunology* 202.3 (Feb. 2019), pp. 637–644.

APPENDIX A

Paper I Appendix

Supplementary Information for

"NetTCR-2.0 enables accurate prediction of TCR-peptide binding by using paired TCRα and β sequence data"

Alessandro Montemurro, Viktoria Schuster, Helle Rus Povlsen, Amalie Kai Bentzen, Vanessa Jurtz, William D. Chronister, Austin Crinklaw, Sine R. Hadrup, Ole Winther, Bjoern Peters, Leon Eyrich Jessen, and Morten Nielsen[$]

[$] Corresponding author: morni@dtu.dk

Supplementary Note 1

pHMM based k-mer method for CDR3β loop sequence excision

As the IEDB is based on collecting published sequence data, the raw CDR3β data downloaded from the IEDB contained not only the CDR3 segment of the VDJ-recombination, but also in some cases included flanking parts of various lengths spanning into the V- and J-segments. To remove these potential excess flanking residues, a profile Hidden Markov Model (pHMM) k-mer based scoring method was developed to extract the correct CDR3β-sequence. Here, the sequenced CDR3β repertoire from 20 healthy donors included in Savola et al.[S1] was used. This data set consists of a total of 487,787 CDR3 sequences of which 405,588 are non-NA, 398,139 of these contain only the 20 standard proteogenic amino acids. A further 352,116 of these are unique and of these 348,249 matched the canonical CDR3 motif "Cxxx...xxx[FW]". Removing the c-terminal "C" and the N-terminal "[FW]", resulted in an average length of 12.7 with a standard deviation of 1.8. The final data set was then created by randomly selecting 100,000 sequences to be used for training, leaving the remaining 248,249 for evaluation. Using this setup, a profile Hidden Markov Model was trained using the Baum–Welch algorithm implemented in the Aphid package[S2]. The resulting pHMM model was then used to score each of the 248,249 evaluation sequences using the Viterbi algorithm, resulting in a k-dependent score distribution (Supplementary Figure 9a) reflecting the underlying CDR3β length distribution (Supplementary Figure 9b). Next, the raw IEDB data containing a total of 25,300 CDR3β-pMHC data points of which 13,274 were specific for HLA-A*02:01, further subsetting to 9-mer peptides, yielded 12,353 data points. Removing CDR3β-sequences containing non-standard 1-letter amino acid symbols resulted in 12,223 data points and finally non-trimmed CDR3β-sequences were required to have a length of at least 5, yielding a final data set of 12,222 data points. As a first step, 3,400 CDR3β-sequences with a N-terminus "C" and a C-terminus "F" or "W" were stripped of said flanks. The remaining 8,822 sequences were digested into all possible nested k-mers (5 <= k <= CDR3β-length) and Viterbi-scored using the trained pHMM. The best scoring k-mer was recorded. Finally, in case the k-mer had a higher Viterbi-score than that of the original full length CDR3β, the trimming was accepted replacing the original CDR3β-sequence. This procedure resulted in trimming 911 sequences corresponding to ~7.5% of the 12,222 sequences in the IEDB CDR3β data set. Selecting the unique CDR3β-peptide pairs from this trimmed data resulted in 11,845 data points and removing promiscuous TCRs yielded 11,122 unique CDR3β-sequences and 163 unique peptides. Finally, requiring CDR3β-sequences to have a length of at least 8 amino acids 18 at most (corresponding to 99% of TCRs in the Savola

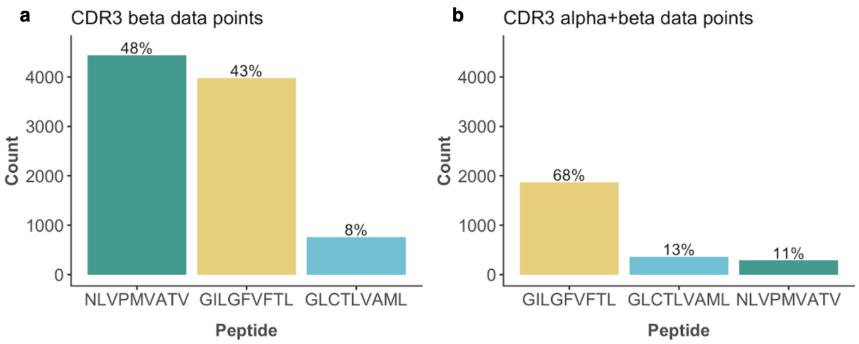
et al. data set (Supplementary Figure 9b)), resulted in 10,987 CDR3β-sequences covering 163 peptides.

Supplementary References

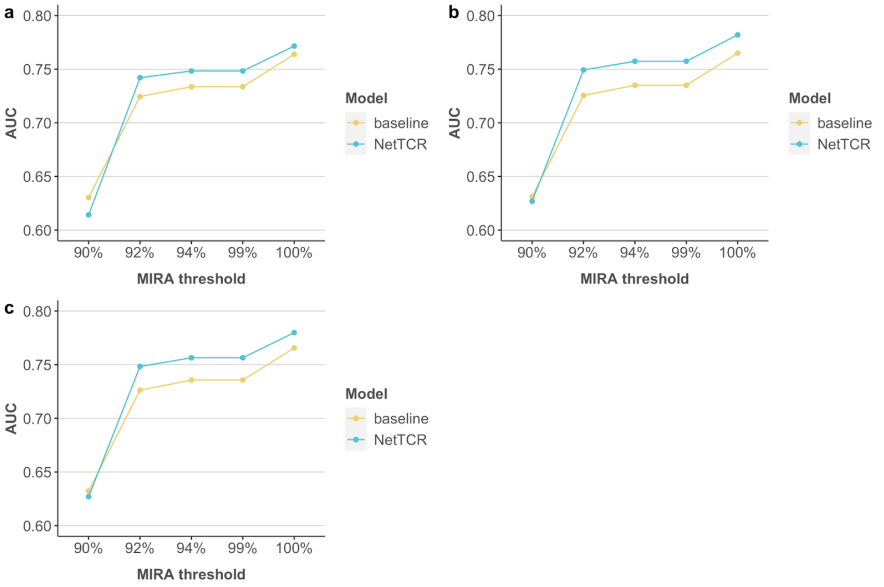
S1. Savola, P. *et al.* Somatic mutations in clonally expanded cytotoxic T lymphocytes in patients with newly diagnosed rheumatoid arthritis. *Nat. Commun.* **8**, 15869 (2017).

S2. Wilkinson, S. P. aphid: an R package for analysis with profile hidden Markov models. *Bioinformatics* **35**, 3829–3830 (2019).

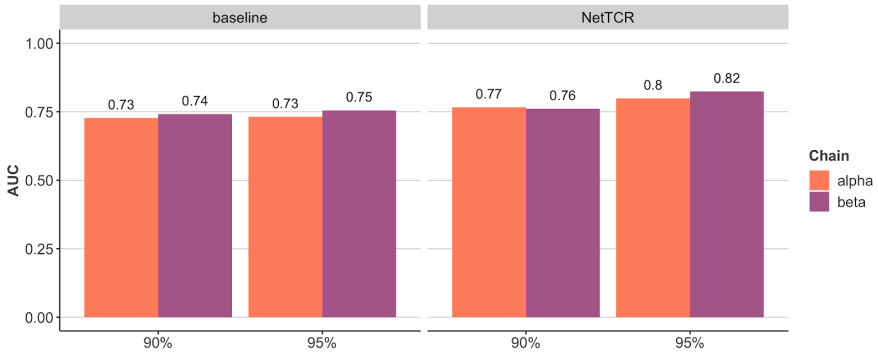
Supplementary Figures



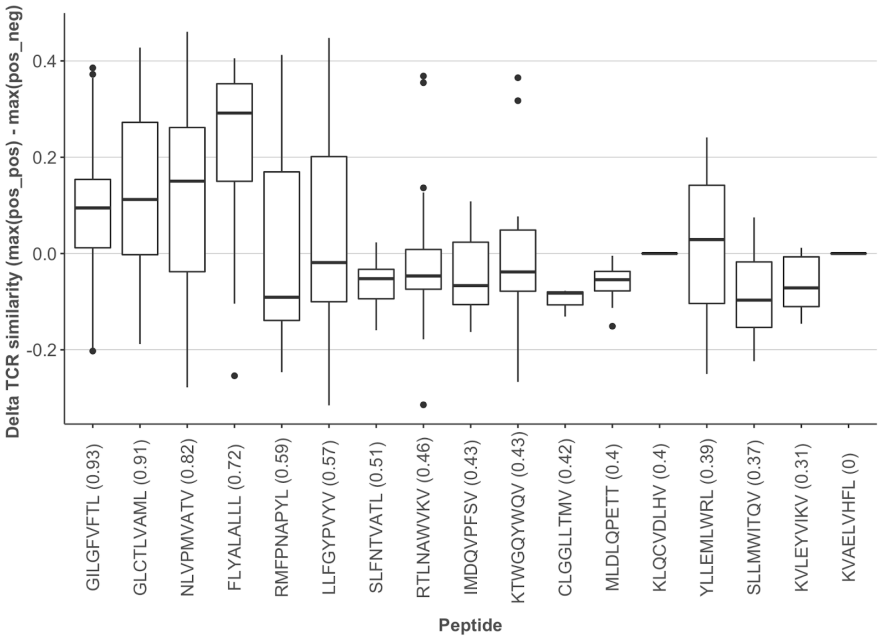
Supplementary Figure 1. Counts of unique data points per peptide. Count for the data sets consisting of only CDR3 β chains (**a**) and both CDR3 α and β chains (**b**). The percentages above bars indicate the representation of peptides in the data.



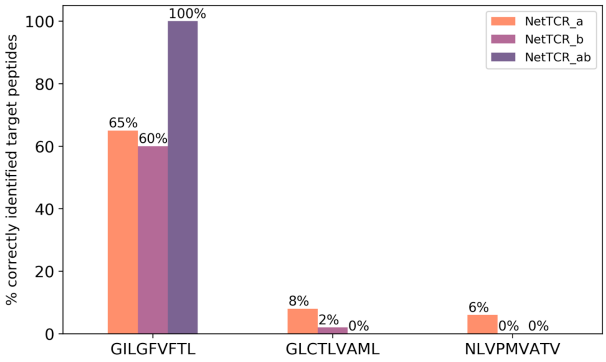
Supplementary Figure 2. Overall AUCs of the CDR3 beta models on the external evaluation MIRA data at different redundancy thresholds of the models trained on the (**a**) 90%, (**b**) 92% and (**c**) 99% partitioned training set.



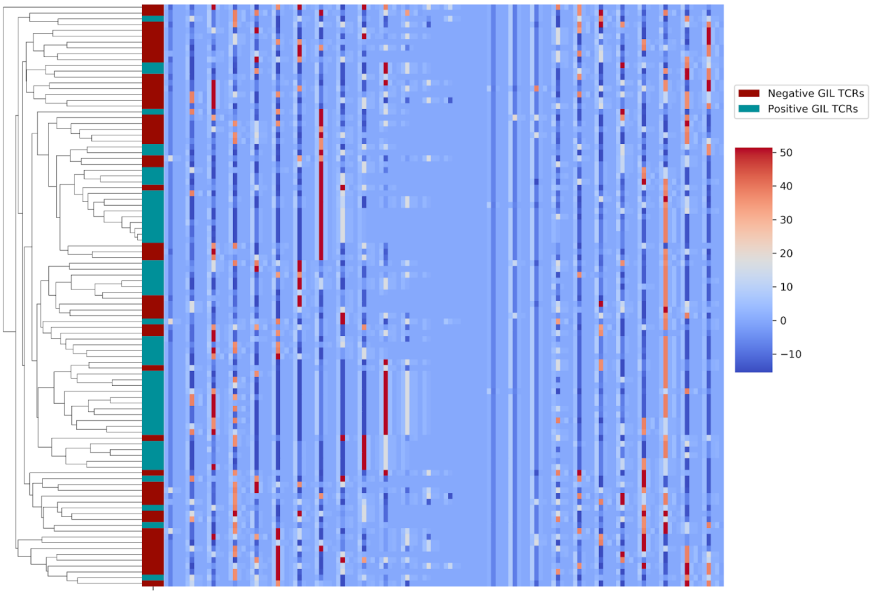
Supplementary Figure 3. Performance of models trained on single-chain data. Overall AUCs evaluated via cross-validation for the different partitioning thresholds. The single-chain data sets were partitioned using a chain-specific partitioning approach.



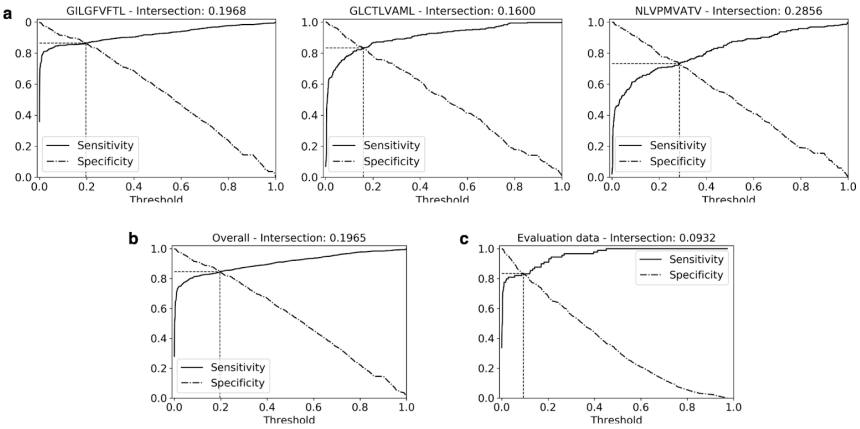
Supplementary Figure 4. Correlation between the performance of the paired chain NetTCR model and the difference between positive and negative data points. The x-axis presents peptides sorted by AUC of the paired chain NetTCR model from 95% partitioning (AUC values indicated next to the peptide). The boxplot shows differences in similarity per peptide between different partitions (see text).



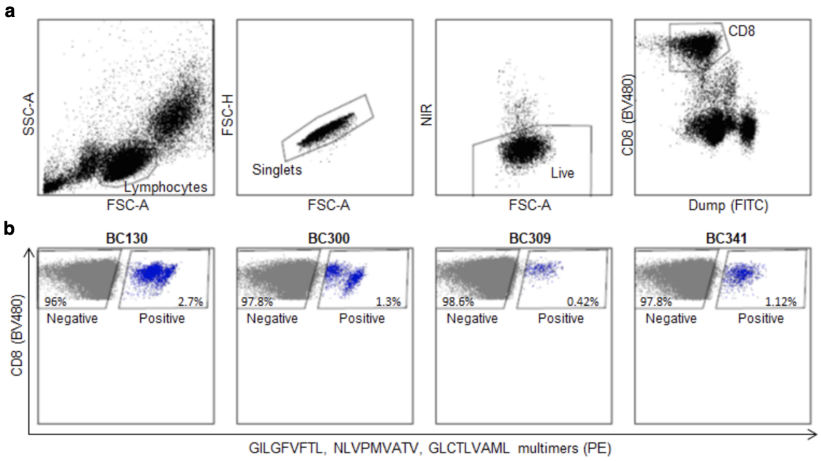
Supplementary Figure 5. Peptide ranking analysis for the paired-chains model trained with a wrong TCR-peptide combination. The TCRs in the training set were paired with a wrong peptide and a model was trained on the mismatched dataset. After, each TCR positive to GIL, GLC, or NLV peptide was paired to the other two peptides and a binding prediction was obtained. The percentages show the proportion of TCRs for which the predicted lowest-ranking peptide matched with the "true" target peptide.



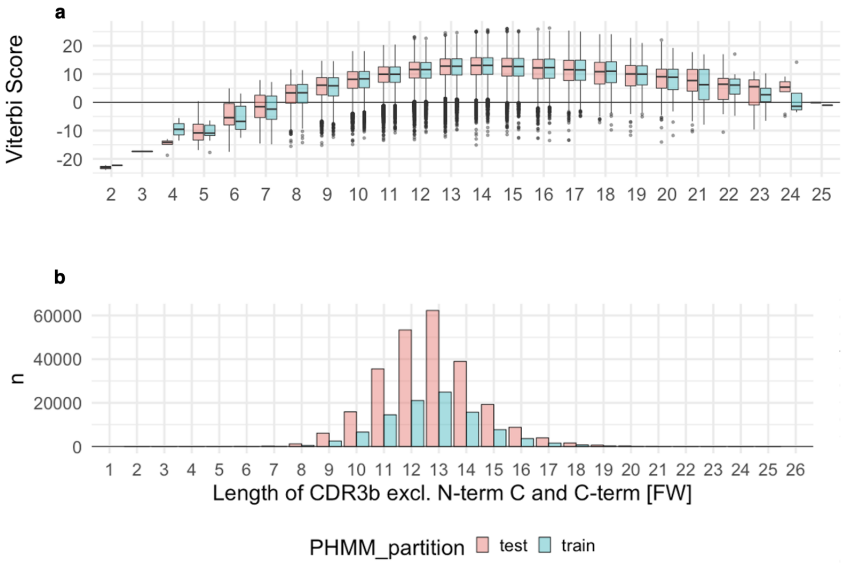
Supplementary Figure 6. Hierarchically-clustered heatmap of a random set of 50 positive and 50 negative TCRs GIL TCRs encoded using the physico-chemical features.



Supplementary Figure 7. Sensitivity and specificity curves as a function of the decision thresholds for NetTCR_αβ. The curves were plotted using peptide-specific percentile rank scores (**a**), all the percentile scores from cross-validation (**b**), and the scores from the external evaluation predictions (**c**).



Supplementary Figure 8. Gating strategy and sorted populations used for generating the novel independent paired TCR dataset. (a) Shows an example of the initial gating of CD8+ T cells (BC341). **(b)** Shows the sorted positive and negative populations of total CD8+ T cells from all four samples included.



Supplementary Figure 9. (**a**) K-dependent distribution of pHMM derived Viterbi scores. (**b**) The length distribution of CDR3β-sequences. All stratified on the pHMM-test/training partition.

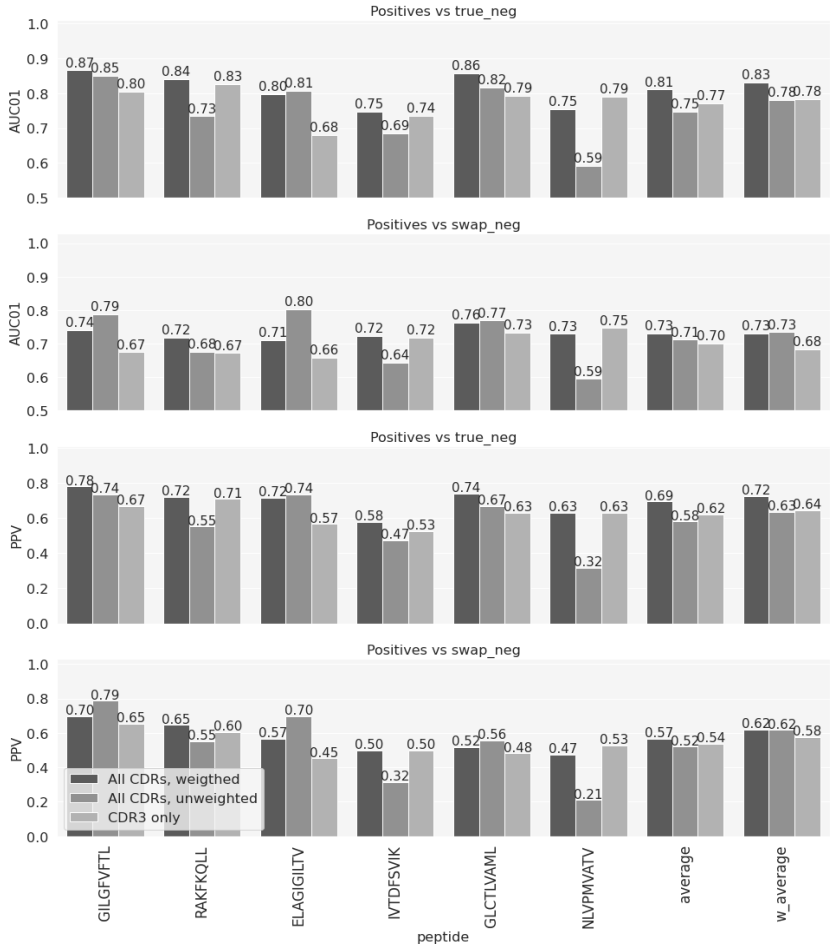
	Predetermined responses			# of sorted cells	
	GILGFVFTL	**GLCTLVAML**	**NLVPMVATV**	**Positive subset**	**Negative subset**
BC130			2.1%	4698	75000
BC300	0.7%	0.9%	0.1%	2469	75000
BC309			0.3%	839	75000
BC341	1.3%			3744	75000

Supplementary Table 1. Information on samples used for generating novel independent paired TCR dataset. All sorted cells in the positive subset were loaded in one lane and 17,000 cells from the negative subset were loaded in another lane. Both were processed using the 10x Chromium pipeline. Percentages are % of total CD8 T cells

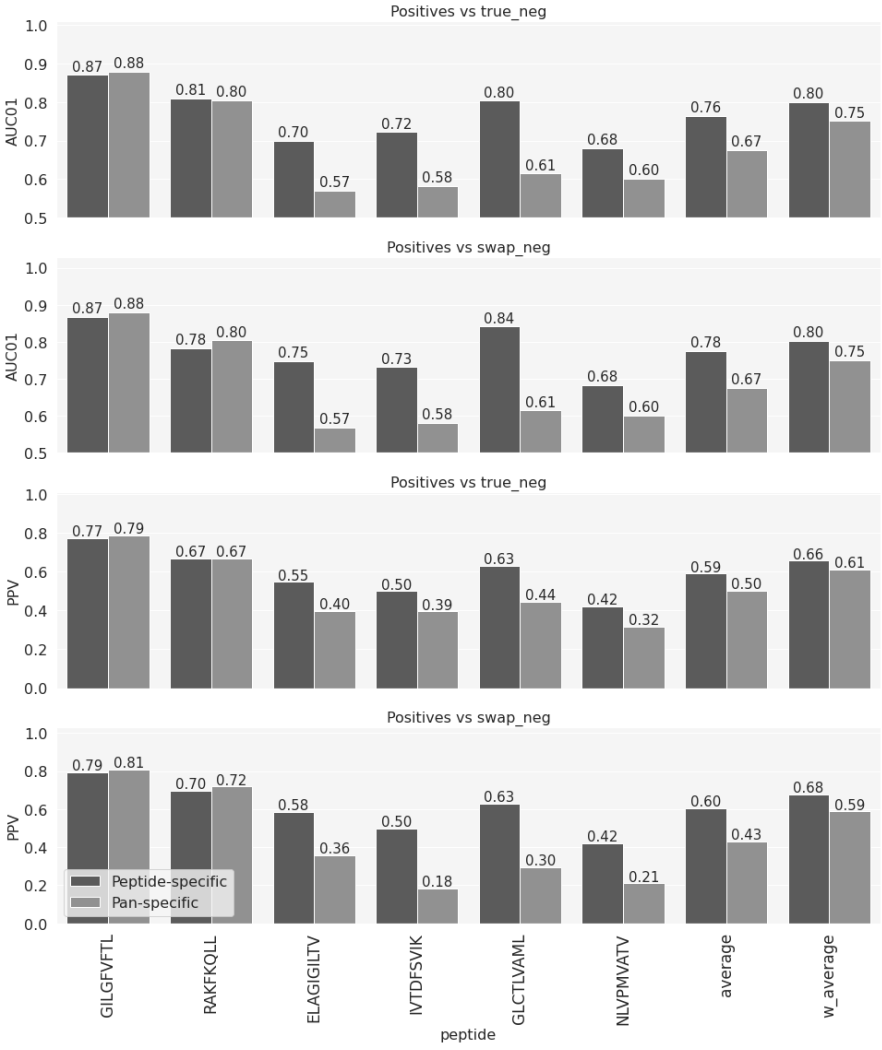
APPENDIX B

Paper II Appendix

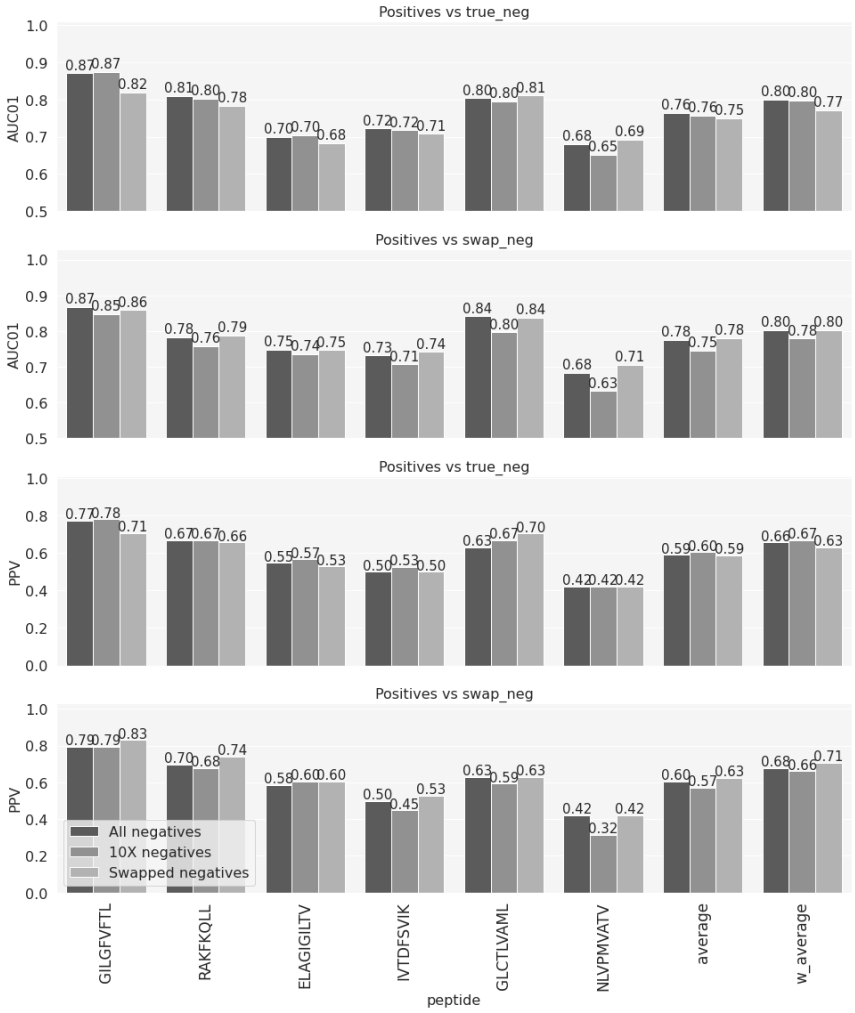
Supplementary Materials



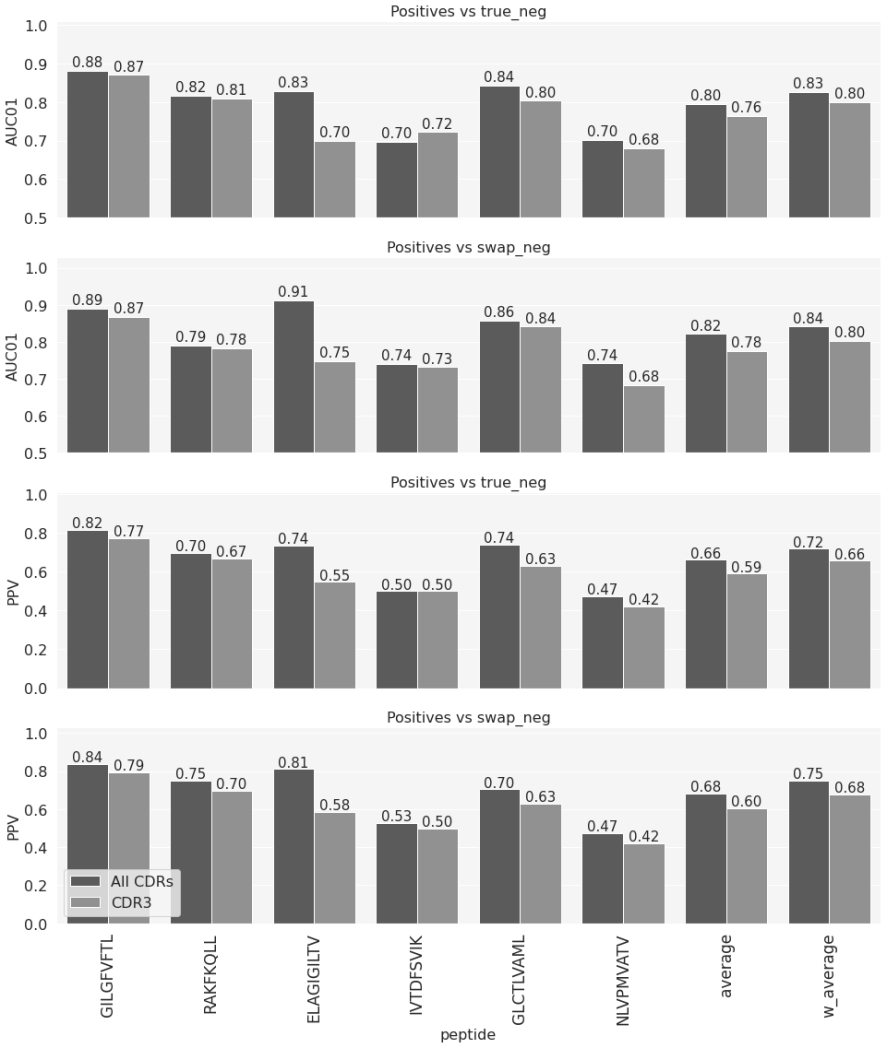
Supplementary Figure 1: Baseline model performance comparison in terms of AUC01 and PPV. The baseline model was used i) with weights [1, 1, 4] on the CDRs; ii) with equal weights on the CDrs; iii) using only CDR3s. The values are given for each peptide, and on the positives vs 10X negatives and positives vs swapped negatives prediction tasks. average and w_average refer to the average and weighted average of the AUC01 (and PPV) across the six peptides.



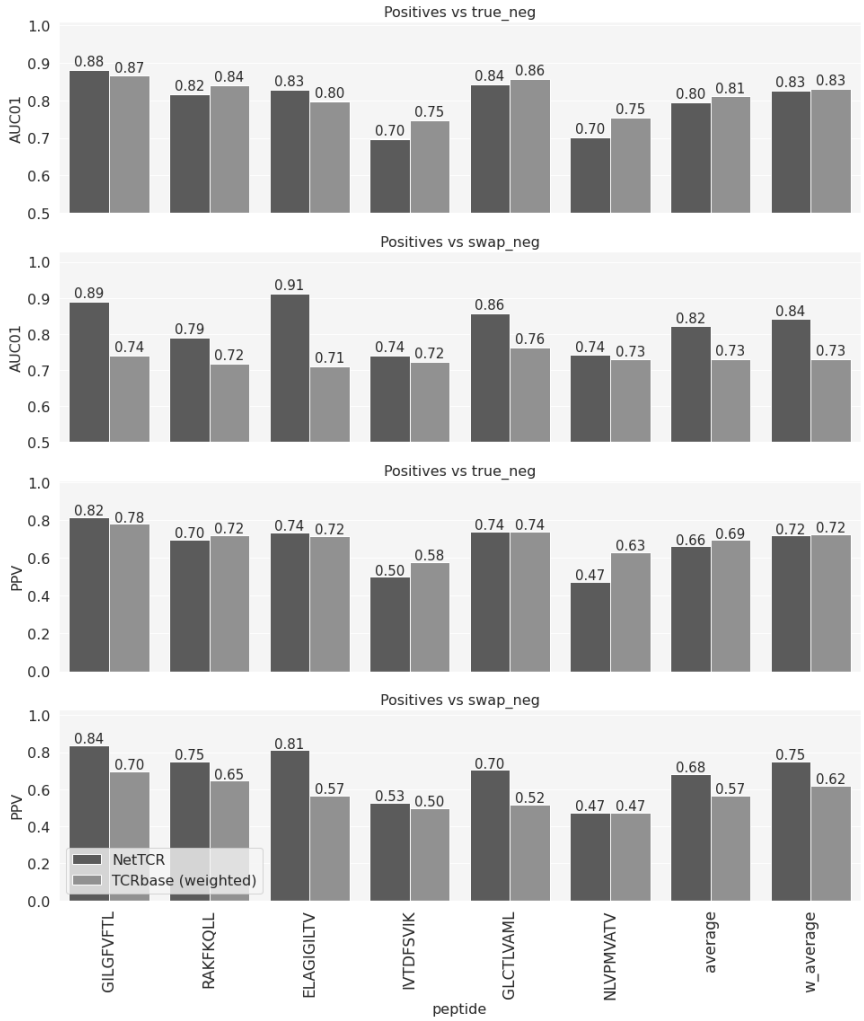
Supplementary Figure 2: AUC01 and PPV values comparison of the NetTCR model trained in a peptide-specific or a pan-specific manner. Performance reported for each peptide, and for positives vs. 10X negatives and positives vs swapped negatives task. Average and weighted average (weighted by the number of positive TCRs for each peptide) performances are also reported.



Supplementary Figure 3: Analysis of the different sources of negatives. AUC01 and PPV values for the NetTCR-CDR3 model trained on i) the full dataset, including positives, 10x negative and swapped negatives; ii) positives and 10x negatives only, iii) positives and swapped negatives only. AUC01 and PPV are reported in a peptide-specific manner; the values are also differentiated based on positives versus 10X/swapped negatives predictions. "average" refers to the mean values of AUC01 (and PPV) from each peptide; "w_average" is a weighted average (weighted by the number of positive TCRs) of the values.



Supplementary Figure 4: Peptide-specific AUC01 and PPV values comparison of the NetTCR models trained using the set of all CDRs or CDR3 only. The predictive power is evaluated for each peptide (average and w_average refer to an average and weighted average, respectively, of the peptide-specific scores). The performance is also differentiated based on the positives vs. 10X/swap negatives predictions.



Supplementary Figure 5: NetTCR versus TCRbase. Performance comparison in terms of AUC01 and PPV. The values are reported for each peptide, and differentiated according to the two prediction tasks, positives vs 10x negatives and positives vs swapped negatives. "average" is calculated as an average of the AUC01 (and PPV) of the peptide-specific scores; "w_average" is a weighted average (weighted by the number of positive TCRs) of the peptide-specific scores.

Technical University of Denmark
Health Technology
Section of Bioinformatics

Kemitorvet 204, 257
2800 Kgs. Lyngby

www.healthtech.dtu.dk