Machine learning and computational analyses of adaptive immune receptors

Lonneke Scheffer

Thesis submitted for the degree of Philosophiæ Doctor

Series of dissertations submitted to the
Faculty of Mathematics and Natural Sciences, University of Oslo
No. 2704

ISSN 1501-7710

Cover: UiO.
Print production: Graphic center, University of Oslo.

Abstract

Machine learning is a powerful technology which has revolutionised the analysis of large and complex datasets. This makes machine learning particularly suitable for dissecting high-dimensional sequencing datasets. With the rapid technological advances in high-throughput sequencing, we've gained the capacity to explore large repertoires of adaptive immune receptors. These immune receptors are of interest as they facilitate the targeted recognition of antigens such as viruses, bacteria or cancer cells. Furthermore, immune receptors play a crucial role in maintaining immunological memory. Thus, the repertoire of immune receptors in an individual is a highly personal recording of their disease status. Researchers have turned to machine learning to address two main applications within this data domain: predicting the antigen binding status of individual immune receptors, and predicting the disease status of an individual based on her or his immune receptor repertoire.

Much is still unknown about the underlying mechanisms determining antigen recognition, and while machine learning holds tremendous potential, its use also comes with challenges. Research conducted by different groups may vary in their underlying domain assumptions, evaluation criteria for method performance, and selected datasets. Robust, standardised and fully reproducible workflows need to be in place to determine the optimal machine learning model for a given study. Transparency and interpretability of models are of critical importance to ensure that the learned patterns represent true biological signals rather than artefacts. And the rapid growth of immune repertoire datasets furthermore underlines the need for computational efficiency.

This thesis introduces new computational methods to assist in the analysis of the immune receptor-antigen binding and immune repertoire-disease status prediction problems. The first paper presents immuneML, a platform for the machine learning based analysis of adaptive immune receptors and repertoires. This platform can be used for the comparative evaluation of predictive models for either immune receptor or repertoire classification, accelerating the development of such models. The second paper introduces CompAIRR, a tool for the ultra-fast and efficient computation of overlap between immune repertoires. CompAIRR has been used to speed up various computational analyses of immune repertoires, including several machine learning models in immuneML. The final paper describes an investigation of how well the antigen binding status of immune receptors can be predicted by the presence of short motifs. Improving our understanding of how the performance of simple, straightforward prediction rules can help guide the development of domain specific machine learning algorithms.

Sammendrag

Maskinlæring er en kraftig teknologi som har revolusjonert analysen av store og komplekse datasett. Dette gjør maskinlæring spesielt egnet for dissekering av høydimensjonale sekvenseringsdatasett. Med de raske teknologiske fremskrittene innen high-throughput-sekvensering, har vi fått kapasiteten til å utforske store repertoarer av adaptive immunreseptorer. Disse immunreseptorene er av interesse da de letter målrettet gjenkjennelse av antigener som virus, bakterier eller kreftceller. Videre spiller immunreseptorer en avgjørende rolle for å opprettholde immunologisk hukommelse. Repertoaret av immunreseptorer hos et individ er således en svært personlig registrering av sykdomsstatusen deres. Forskere har vendt seg til maskinlæring for å adressere to hovedapplikasjoner innenfor dette datadomenet: å forutsi antigenbindingsstatusen til individuelle immunreseptorer, og forutsi sykdomsstatusen til et individ basert på deres immunreseptorrepertoar.

Mye er fortsatt ukjent om de underliggende mekanismene som bestemmer antigengjenkjenning, og selv om maskinlæring har et enormt potensial, kommer bruken også med utfordringer. Forskning utført av ulike grupper kan variere i deres underliggende domeneforutsetninger, evalueringskriterier for metodeytelse og utvalgte datasett. Robuste, standardiserte og fullt reproduserbare arbeidsflyter må være på plass for å bestemme den optimale maskinlæringsmodellen for en gitt studie. Åpenhet og tolkning av modeller er av avgjørende betydning for å sikre at de lærte mønstrene representerer ekte biologiske signaler i stedet for spuriøse korrelasjoner. Og den raske veksten av immunrepertoardatasett understreker dessuten behovet for beregningseffektivitet.

Denne oppgaven introduserer nye beregningsmetoder for å bistå i analysen av immunreseptor-antigenbinding og immunrepertoar-sykdomsstatus prediksjonsproblemer. Den første artikkelen presenterer immuneML, en plattform for maskinlæringsbasert analyse av adaptive immunreseptorer og -repertoarer. Denne plattformen kan brukes til komparativ evaluering av prediktive modeller for enten immunreseptor- eller repertoarklassifisering, og akselererer utviklingen av slike modeller. Den andre artikkelen introduserer CompAIRR, et verktøy for ultrarask og effektiv beregning av overlapping mellom immunrepertoarer. CompAIRR har blitt brukt til å fremskynde ulike beregningsanalyser av immunrepertoarer, inkludert flere maskinlæringsmodeller i immuneML. Den endelige artikkelen beskriver en undersøkelse av hvor godt antigenbindingsstatusen til immunreseptorer kan forutsies ved tilstedeværelse av korte motiver. Å forbedre vår forståelse av hvordan ytelsen til enkle, greie prediksjonsregler kan hjelpe til med å lede utviklingen av domenespesifikke maskinlæringsalgoritmer.

Contents

Preface

Introduction

Machine learning (ML) is a subfield of artificial intelligence which concerns the development of self-learning algorithms. Often relying heavily on statistics and mathematical optimisation, ML algorithms can be used to find patterns in data, distinguish groups and make predictions. While artificial intelligence was already an active field of research in the 1950s, initial computational limitations hampered its development and resulted in a loss of interest. However, the recent increase in available data and computational power, as well as technological breakthroughs have resurrected a new interest in this field, in particular in ML algorithms known as deep learning methods. Nowadays, ML algorithms are used in a wide variety of applications, ranging from practical everyday settings to cutting-edge research.

The rekindled interest in ML has revolutionised the analysis of large and complex datasets, including biomedical data. Through the application of ML, we have gained unprecedented insight into biological processes, and accelerated developments in healthcare. Some of the recent breakthroughs include the prediction of 3D protein structures, the discovery of new drugs, vaccine design and the predictive diagnostics for a large range of diseases. One particular type of biomedical data that has recently seen benefits from ML-based analysis is *adaptive immune receptors* (AIRs). AIRs are produced by B and T cells, and serve as our immune system's tool to recognise, neutralise and remember potentially harmful foreign substances or cancer cells. The human body produces a large variety of AIRs, known as the *AIR repertoire* (AIRR), which provides protection against an immeasurable number of different antigens. While AIRs are highly diverse, most of this diversity only occurs within a small region at the tip of the AIR that makes contact with the antigen. This makes AIRs an excellent subject for ML-based classification, by aiming to learn a mapping between this short, information-dense subsequence of the AIR, and its potential antigen target(s). Such classification models could accelerate in silico drug development.

Furthermore, the AIRR continually changes over the course of a person's lifetime in response to exposure to different antigens. This makes the AIRR a highly valuable resource of health information, as infectious diseases, autoimmune diseases or cancer can potentially leave a traceable mark. Recent developments in high-throughput sequencing have for the first time enabled us to sequence substantial portions of AIRRs from large patient cohorts. This offers another opportunity for ML-based classification of adaptive immune data: predicting the disease status of an individual based on their AIRR. An improved understanding of how disease signals may appear as learnable patterns in the AIRR may enable the creation of a universal diagnostic tool.

The complex biology of AIR(R) data warrants the development of domain specific ML methodology for the prediction of AIR-antigen binding and AIRR disease status. Some models have already been developed for this purpose throughout the last decade. However, to be truly useful, it is crucial to understand the individual strengths and weaknesses of models, and ensure that the learned predictive patterns represent true biological markers rather than artefacts. Understanding which ML methods are best for a given AIR(R) study is not trivial, as methods are often evaluated based on different criteria, using different datasets. Transparent evaluation of methods, reproducibility, and interpretability of results are vital for the progression of the AIR(R)-ML field. Moreover, as dataset sizes are rapidly growing, computational efficiency becomes more important. To address these challenges, this thesis introduces several computational methods to support the ML-based analysis of AIR(R) data. The foundation of this thesis is a software platform for the comparative evaluation of AIR(R)-ML classifiers. Furthermore, a component for the ultra-fast and efficient comparison of AIRRs has been developed and used to speed up several applications. Lastly, a method was developed to analyse the antigen binding capabilities of AIRs based on the presence of short motifs.

Acknowledgements

My PhD journey has been an intense and transformative experience, leading to growth both professionally and personally. I deeply appreciate the support and guidance of everyone who has helped me throughout these years.

First and foremost, my warmest thanks go out to my supervisor, Geir Kjetil Sandve, with whom I've had the joy of working not only during my PhD but also throughout my master's and an internship during my first year in Norway. Through many interesting discussions and rounds of feedback – never without a spark of humour – Geir Kjetil has helped me discover and develop myself as a scientist. I would also like to thank my co-supervisor, Victor Greiff, for his detailed feedback and insightful perspectives. Additionally, I extend my sincere appreciation to Bjoern Peters, who unexpectedly joined as a co-supervisor towards the end of my PhD and facilitated a fantastic research stay at the La Jolla Institute for Immunology in San Diego. Another person I'd like to thank for her invaluable contributions in enabling this research stay, as well as her overall support throughout my PhD, is SUURPh coordinator Kimberly McCabe.

Throughout these years I've worked alongside many great colleagues in the Sandve, Greiff and Peters labs, who I thank for creating a collaborative and supportive work environment. In particular, I'd like to thank Milena Pavlović, with whom I co-developed the immuneML platform and shared an office. Working alongside Milena has been a pleasure and taught me new programming skills. Two other people who deserve a special thanks for brightening up the office are Maria Chernigovskaya and Jahn Zhong. I want to express my gratitude to Marcus Mendez and Eve Richardson for welcoming me into the San Diego office. Additionally, I'm thankful to the SUURPh student group, as well as my friends Forat Seif and Ryan Stall, for providing a strong social network both in Oslo and San Diego. Lastly, I would like to thank all my other friends and family for their support.

List of papers

Paper 1

Pavlović, M.*, Scheffer, L.*, Motwani, K., Kanduri, C., Kompova, R., Vazov, N., Waagan, K., Bernal, F. L. M., Costa, A. A., Corrie, B., Akbar, R., Al Hajj, G. S., Balaban, G., Brusko, T. M., Chernigovskaya, M., Christley, S., Cowell, L. G., Frank, R., Grytten, I., Gundersen, S., Haff, I. H., Hovig, E., Hsieh P.-H., Klambauer, G., Kuijjer, M. L., Lund-Andersen, C., Martini, A., Minotto, T., Pensar, J., Rand, K., Riccardi, E., Robert, P. A., Rocha, A., Slabodkin, A., Snapkov, I., Sollid, L. M., Titov, D., Weber, C. W., Widrich, M., Yaari, G., Greiff, V., Sandve, G. K. (2021). **The immuneML ecosystem for machine learning analysis of adaptive immune receptor repertoires**. *Nature Machine Intelligence*, 3(11), 936–944. https://doi.org/10.1038/s42256-021-00413-z

* co-first authors

Paper 2

Rognes, T.*, Scheffer, L.*, Greiff, V., Sandve, G. K. (2021). **CompAIRR: ultra-fast comparison of adaptive immune receptor repertoires by exact and approximate sequence matching**. *Bioinformatics*, 38(17), 4230–4232. https://doi.org/10.1093/bioinformatics/btac505

* co-first authors

Paper 3

Scheffer, L., Reber, E. E., Mehta, B. B., Pavlović, M., Lê Quý, K., Richardson, E., Akbar, R., Greiff, V., Haff, I. H., Sandve, G. K. **Predictability of antigen binding based on short motifs in the antibody CDRH3**. *Manuscript in preparation*.

Other publications

The following is a list of papers that I have contributed to throughout the Ph.D., but are not included as main papers in the thesis:

- Balaban, G., Grytten, I., Rand, K. D., Scheffer, L., Sandve, G. K. (2021). Ten simple rules for quick and dirty scientific programming. PLoS Comput Biol, 17(3), https://doi.org/10.1371/journal.pcbi.1008549
- Akbar, R., Robert, P. A., Pavlović, M., Jeliazkov, J. R., Snapkov, I., Slabodkin, A., Weber, C. R., Scheffer, L., Miho, E., Haff, I. H., Haug, D. T. T., Lund-Johansen, F., Safonova, Y., Sandve, G. K., & Greiff, V. (2021). A compact vocabulary of paratope-epitope interactions enables predictability of antibody-antigen binding. Cell Reports, 34(11), 108856. https://doi.org/10.1016/j.celrep.2021.108856

- Slabodkin, A., Chernigovskaya, M., Mikocziova, I., Akbar, R., Scheffer, L., Pavlović, M., Bashour, H., Snapkov, I., Mehta, B. B., Weber, C. R., Gutierrez-Marcos, J., Sollid, L. M., Hobæk Haff, I., Sandve, G. K., Robert, P. A., Greiff, V. (2021). Individualized VDJ recombination predisposes the available Ig sequence space. Genome Res, 31, 2209-2224, https://doi.org/10.1101/gr.275373.121

- Dahal-Koirala, S., Balaban, G., Neumann, R. S., Scheffer, L., Aslaksen Lundin, K. E., Greiff, V., Sollid, L. M., Qiao, S. W., Sandve, G. K. (2022). TCRpower: quantifying the detection power of T-cell receptor sequencing with a novel computational pipeline calibrated by spike-in sequences, Brief Bioinform, 23(2), https://doi.org/10.1093/bib/bbab566

- Kanduri, C., Pavlović, M., Scheffer, L., Motwani, K., Chernigovskaya, M., Greiff, V., Sandve, G. K. (2022). Profiling the baseline performance and limits of machine learning models for adaptive immune receptor repertoire classification, GigaScience, 11, https://doi.org/10.1093/gigascience/giac046

- Robert, P. A., Akbar, R., Frank, R., Pavlović, M., Widrich, M., Snapkov, I., Slabodkin, A., Chernigovskaya, M., Scheffer, L., Smorodina, E., Rawat, P., Mehta, B. B., Vu, M. H., Frøberg Mathisen, I., Prósz, A., Abram, K., Olar, A., Miho, E., Trygve Tryslew Haug, D., Lund-Johansen, F., Hochreiter, S., Hobæk Haff, I., Klambauer, G., Sandve, G. K., Greiff, V. (2022). Unconstrained generation of synthetic antibody–antigen structures to guide machine learning methodology for antibody specificity prediction. Nat Comput Sci, 2, 845–865 https://doi.org/10.1038/s43588-022-00372-4

- Akbar, R., Robert, P. A., Weber, C. R., Widrich, M., Frank, R., Pavlović, M., Scheffer, L., Chernigovskaya, M., Snapkov, I., Slabodkin, A., Mehta, B. B., Miho, E., Lund-Johansen, F., Andersen, J. T., Hochreiter, S., Hobæk Haff, I., Klambauer, G., Sandve, G. K., Greiff, V. (2022). In silico proof of principle of machine learning-based antibody design at unconstrained scale, mAbs, 14(1), https://doi.org/10.1080/19420862.2022.2031482

- Kanduri, C., Scheffer, L., Pavlović, M., Dagestad Rand, K., Chernigovskaya, M., Pirvandy, O., Yaari, G., Greiff, V., Sandve, G. K. (2023). simAIRR: simulation of adaptive immune repertoires with realistic receptor sequence sharing for benchmarking of immune state prediction methods, GigaScience, 12, https://doi.org/10.1093/gigascience/giad074

List of abbreviations

Abbreviation	**Meaning**
AIR	Adaptive immune receptor
AIRR	Adaptive immune receptor repertoire
AIRR-seq	Adaptive immune receptor repertoire sequencing
BCR	B cell receptor
CDR	Complementarity determining region
CMV	Cytomegalovirus
CNN	Convolutional neural network
FR	Framework region
HLA	Human leukocyte antigen
IEDB	Immune epitope database
IID	Independent and identically distributed
LJI	La Jolla Institute for Immunology
LSTM	Long short-term memory
MHC	Major histocompatibility complex
MIL	Multiple instance learning
ML	Machine learning
pMHC	Peptide-major histocompatibility complex
RNN	Recurrent neural network
SGD	Stochastic gradient descent
SHM	Somatic hypermutations
TCR	T cell receptor
VAE	Variational autoencoder

Background

This thesis concerns the computational, and specifically ML-based analysis of AIR and AIRR data. The background is structured as follows: first, an overview is given of the research challenges being faced in this thesis – the biology of AIR(R)s, including factors that contribute to the complexity of AIR-antigen binding and the composition of AIRRs. Thereafter, methodological aspects are introduced, covering the relevant ML concepts. Lastly, a review is given of the previous efforts of predicting AIR-antigen binding and AIRR disease status, as well as the benchmarking thereof.

The adaptive immune system

The adaptive immune system is a highly complex system that provides targeted protection against foreign threats and cancers, as well as maintaining an immunological memory that ensures a quick response upon re-exposure to the same substance. Two key cell types make up the adaptive immune system: B and T cells. Both B and T cells are leukocytes (white blood cells), which play a critical role in the identification and destruction of infectious agents, such as viruses and bacteria, or cancer cells. Specifically, any substance that is recognised by the adaptive immune system and able to elicit an immune response is called an *antigen*. Antigens can be bound by the B and T cell receptors (BCRs and TCRs) expressed on the surface of the cell. These BCRs and TCRs are collectively referred to as AIRs, whereas the repertoire of AIRs present in an individual is called immune repertoire or AIRR.

Whereas B cells recognise free-floating antigens in the blood, T cells are responsible for the cell-mediated immune response. For recognition by T cells, antigens must be presented on a major histocompatibility complex (MHC) molecule, which allows a cell to present small protein fragments originating from within the cell to be expressed on its surface [1]. This way, T cells can detect whether antigens are present inside other cells, and subsequently signal for these cells to go into pre-programmed cell death or stimulate other immune cells. B cells do not require antigens to be presented on MHC, and are furthermore able to excrete BCRs, which are in their secreted form known as *antibodies*. Free-floating antibodies play a crucial role in the neutralisation of antigens, by binding to their surface. Antibodies may furthermore agglomerate antigens together, making these antigens an easier target for destruction by phagocytosis [2]. Moreover, antigen-bound antibodies can activate the complement system, leading to direct destruction of the cell walls of foreign cells (e.g., bacterial cells), as well as sending chemical signals to attract more immune cells to the area [2,3].

B or T cells typically express many copies of only one identical AIR. Upon antigen recognition, a B or T cell transitions from its naive state to an activated state. These activated cells rapidly proliferate in order to multiply the number of cells carrying AIRs that are specific to the same antigen [2]. This process is referred to as *clonal expansion*. All B and T cells derived from the same progenitor cell are said to belong to the same *clone*. When proliferating, activated B cells undergo a process called somatic hypermutation (SHM), during which the DNA encoding the immune receptor chains is duplicated with an extremely high mutation rate [4]. Most mutations occur in the regions that are likely to be in contact with the antigen [5]. The resulting daughter cells express AIRs with slight variations in binding affinity to the target antigen, and those with the strongest affinity will also receive the strongest activation signal, contributing to affinity maturation [6]. Some fraction of the activated B and T cells differentiate into long-lived memory cells. These memory cells may survive in the body for many decades, even without re-exposure to the same antigen [7]. When being re-exposed to the same antigen, memory B and T cells enable a quick and strong secondary immune response.

Adaptive immune receptors

BCRs and TCRs share a structural similarity, as they are created in a similar generational process. All AIRs consist of two chains that are paired together (Figure 1). In TCRs, these chains are most often α and β chains, or less commonly γ and δ chains, whereas BCRs typically consist of heavy and light chains, or alternatively heavy and kappa chains. Each of these two chains consists of a constant and a variable region. Within each variable region there are three complementarity-determining regions (CDRs), which are protruding loops that may be in contact with the antigen. The CDRs are flanked by more conserved framework regions (FRs) to maintain the structure of the AIR.

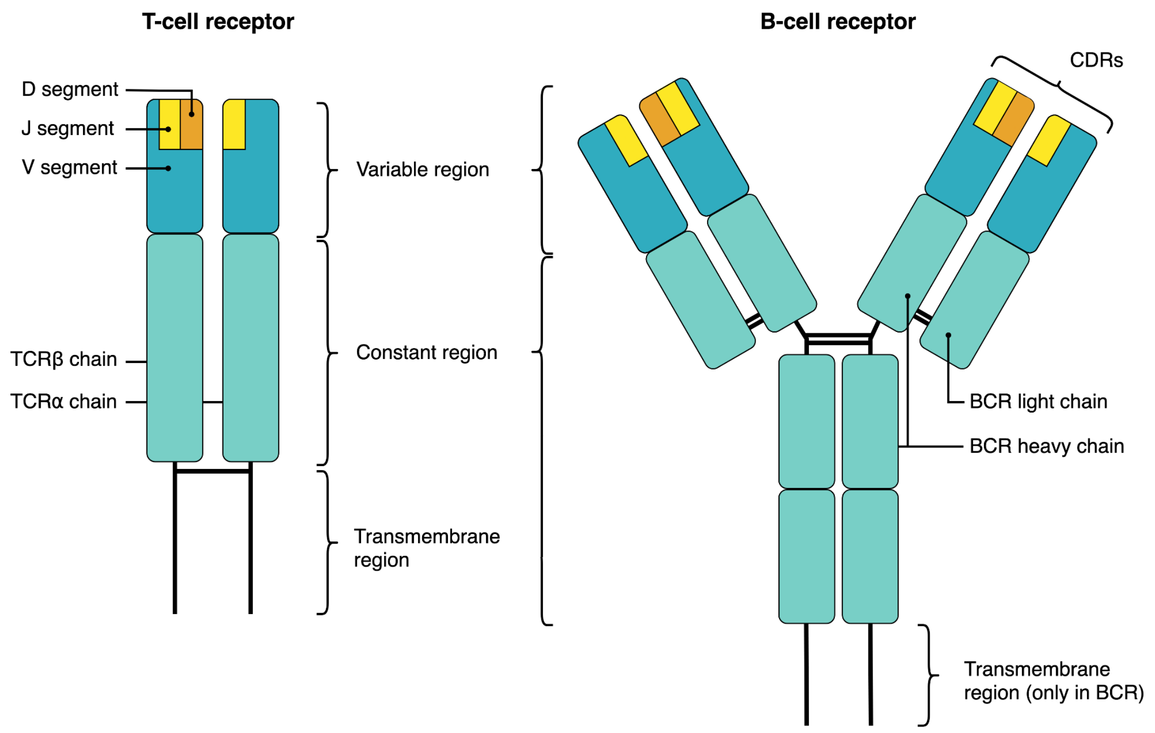


Figure 1: example of a T and B cell receptor. Both types of receptors consist of two chains, here shown are TCR α and β chains, and BCR heavy and light chains. The CDRs in the variable region mediate antigen binding, whereas the constant region provides structural stability. The variability of the tip of the AIR is a result of the AIR chain genes being produced in a process called V(D)J recombination (Figure 2). While TCRs are always membrane-bound, BCRs may either be membrane bound or secreted in the form of antibodies. Figure modified from: Backhaus [8] and Calis and Rosenberg [9]. An earlier version of this figure originally appeared in my master's thesis [10].

Generation of AIR genes through V(D)J recombination

As the total space of antigens is potentially infinite, a large diversity of AIRs is needed in order to provide a broad protection against different antigens. To ensure a broad AIR diversity, the genes encoding the two AIR chains are created in a partially stochastic process called V(D)J recombination (Figure 2) [11]. A multitude of different variable (V), diversity (D) and joining (J) gene segments are encoded in the DNA. When a TCR β or BCR heavy chain gene is created, a randomly chosen pair of D and J gene segments are first recombined, followed by a V gene segment. As the intermediate regions between the gene segments are spliced out, few additional nucleotides may be removed at the ends of the gene segments, and random new nucleotides may be inserted [11]. The TCR α and BCR light chains are created in a similar process, with the exception of a lacking D gene segment. The initial diversity of immune receptor chains is thus ensured in two different ways: *combinatorial diversity* results from the recombination of different gene segments, and *junctional diversity* is created by the random deletion and insertion of nucleotides at the gene segment junctions [12]. As AIRs consist of two separate chains, a third source of diversity is through the pairing of different α/β or heavy/light chains. The total potential diversity of the TCR repertoire has been estimated to be around 10^{15} [13]. For BCRs, the naive repertoire is considered to be around 10^{12}, but after SHM could contain up to 10^{18} unique receptors [14]. In addition to the V, optional D and J segments, the AIR chain gene contains a constant (C) gene segment, which encodes a structural part of the AIR. Different C segments may be used to create antibodies with different immunological functionalities [15].

Due to AIRs being created in a chain of stochastic events, each AIR has a different probability of being generated by a developing immune cell [16,17]. Biases in V(D)J recombination cause certain gene segments to be used preferentially over others, or to be subject to preferential gene segment pairing [18,19]. Random insertions/deletions or SHM occur at biased rates as well. Furthermore, the specific set of gene segments present in the genome may differ across individuals [20], and individual-specific V(D)J recombination biases have been observed [21,22]. As a consequence, the same AIR may have a different probability of being generated by different individuals.

Although B and T cells usually express a single type of AIR resulting from one V(D)J rearrangement for each chain, exceptions exist. Since humans are diploid organisms, it is possible to create up to two V(D)J rearrangements per receptor chain locus in a B or T cell. Having a second opportunity to create a receptor chain gene is advantageous, since many rearrangements fail to create a productive in-frame sequence [23]. Rearrangement is usually halted after a productive receptor chain gene is produced. However, it is occasionally possible for a cell to produce and express multiple productive chains (‘dual’ chains) [24,25], which may result in one immune cell expressing multiple receptors with different antigen specificities.

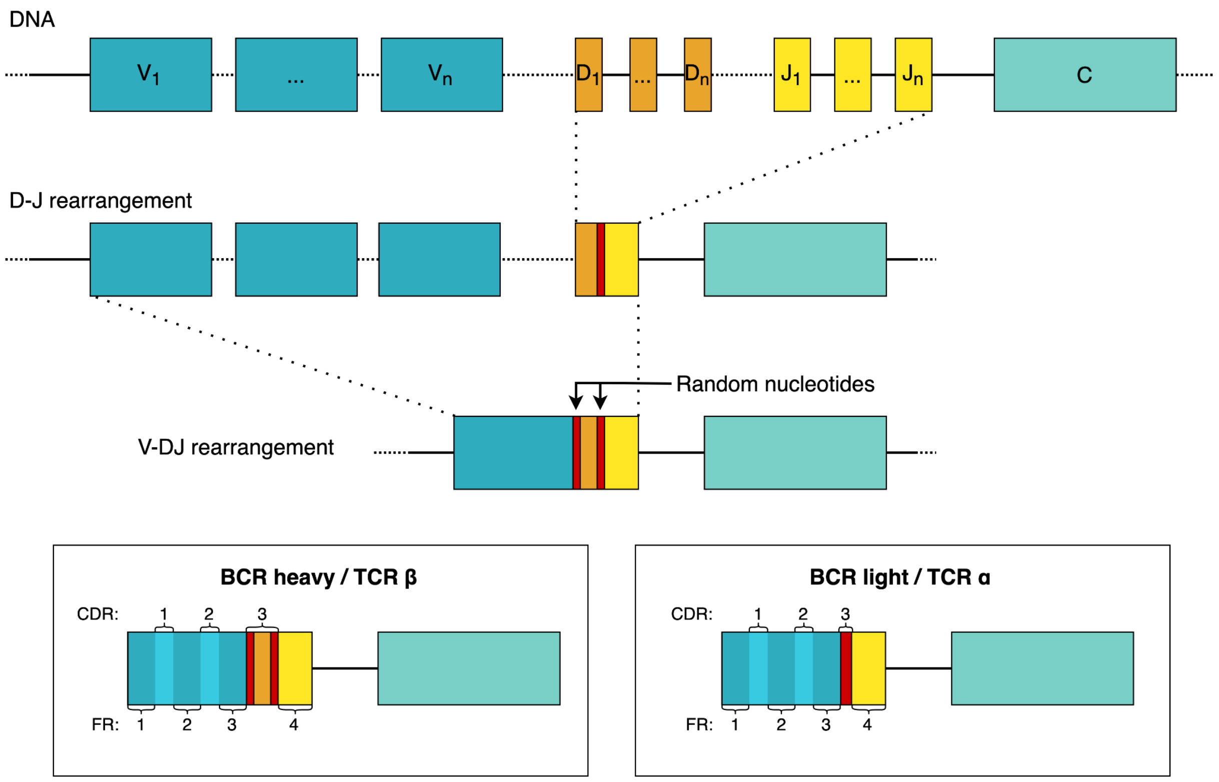


Figure 2 V(D)J recombination of the TCR and BCR genes. V(D)J recombination is a two-step process: first, a D and J gene segment are imperfectly recombined, followed by a V gene segment. Random nucleotide insertions and deletions may occur at the junction sites between gene segments, creating junctional diversity. Only the TCR β and BCR heavy chains contain a D gene segment, for TCR α and BCR light chains, the D segment is omitted and the V and J segments are recombined directly. The genomic location where additional C segments for the BCR heavy and TCR β chains may appear is marked with an asterisk (*). Figure modified from: Backhaus [8]. An earlier version of this figure originally appeared in my master's thesis [10].

Antigen presentation by the major histocompatibility complex

MHC molecules mediate cellular immunity by presenting antigen fragments to TCRs. There are two antigen-presenting subgroups of MHC molecules: MHC classes I and II. All nucleated cells as well as platelets express MHC class I, while MHC class II is presented only on so-called professional antigen-presenting cells, which are able to ingest antigens through phagocytosis [2]. In either case, small protein fragments (peptides) are bound to the MHC molecule and subsequently expressed on the surface of the cell [26]. This allows the MHC expressing cell to communicate to other cells what kinds of proteins are present inside. MHC class I is primarily used to express fragments of proteins circulating freely in the cellular fluid. Peptides that are commonly found in healthy cells are usually not recognised by T cells, as such self-reactive T cells are eliminated during maturation [27]. However, peptides expressed by infected or cancerous cells may be recognised by cytotoxic T cells, which will signal to the dysregulated cell to go into apoptosis. The MHC class II pathway instead concerns the expression of peptides from phagocytosed antigens, activating T helper cells, which further stimulate and regulate other immune cells [26].

The human variant of MHC is sometimes referred to as the human leukocyte antigen (HLA) complex. Both for MHC classes I and II, there exist several different HLA subtypes. The ability for a given TCR to bind to a peptide is dependent on the specific HLA subtype [28]. This concept is called MHC restriction [29]. MHC restriction complicates AIRR research, as identical TCRs may have different antigen binding capabilities across individuals with different HLA subtypes. The HLA subtype furthermore restricts what peptides can be presented, thereby further shaping the TCR repertoire [26]. For MHC class II, certain HLA subtypes are associated with higher risk of developing autoimmune diseases such as celiac disease [30] and type 1 diabetes [31].

Immune receptor-antigen binding

AIRs bind to antigens in 3D space. Antigen binding *specificity* is determined by the 3D shape of the AIR CDRs, as well as biophysicochemical interactions [32,33]. The part of the antigen that is recognised by the AIR is called the *epitope* [34], whereas the amino acids inside the AIR that make contact with the epitope are called the *paratope* [35]. Paratopes usually involve short stretches of contiguous amino acids in the CDR3, but may occasionally contain few gap positions [36–38].

The CDR3 of the BCR heavy or TCR β chain is the most variable region of the AIR. This is due to this CDR3 being located at the junction of the V, D and J gene segments, whereas the BCR light or TCR α chains do not utilise the D gene segment. While all of the 6 AIR CDRs may be in contact with the antigen, the CDR3 of the antibody heavy chain or TCR β chain is considered to be the most important CDR for determining antigen binding [39,40]. Analyses of AIR-antigen crystal structures have shown that the CDR3 is nearly always in contact with the antigen [36–38,41], whereas this is not necessarily true for the other CDRs.

Despite the overall similarity in the structure and function of BCRs and TCRs, there are some crucial differences in how they recognise antigens. Since TCRs recognise peptides presented on MHC, their epitopes (or at least, the peptide portion within the epitope) are necessarily linear. Contrary, BCR epitopes are usually confirmational, meaning they are composed of several components that are physically close in 3D space, but may be far apart in the amino acid sequence of the antigen [34]. The TCR does not only recognise the presented antigen fragment, but recognises the peptide-MHC (pMHC). Typically, the TCR CDR3s are in contact with the peptide, whereas CDRs 1 and 2 contact the MHC [28]. For BCRs, all CDRs can be involved in antigen contact [38].

AIRs are not specific to an individual epitope, but are highly cross-reactive and can bind many different epitopes [42–44]. Similarly, an epitope can be bound by many different AIRs. AIR-antigen binding can be experimentally tested, but depending on the experimental setup, the exact epitope information may remain unknown, for example when the antigen target for BCRs carries multiple epitopes [45]. Furthermore, AIR-antigen binding is not a binary value. Instead, the *affinity* of the AIR-antigen interaction describes the binding strength of an individual AIR paratope, whereas the *avidity* is the total antigen binding strength of a BCR, which carries two identical paratopes [2]. However, even when an antigen target of an AIR is known, its affinity or avidity is not always experimentally verified. In practice, AIRs are often simply labelled as ‘binders’ to a particular target, whereas non-binding is often not explicitly confirmed [46].

The adaptive immune receptor repertoire

The AIRR is shaped by the immune responses an individual has experienced throughout their lifetime, and therefore contains a wealth of information about their health [47]. Changes in the AIRR could potentially be used as biomarkers for past and ongoing infections, and may be used to diagnose infectious diseases, autoimmune diseases or cancer long before clinical symptoms appear [48], [49]. Apart from being a natural diagnostic, the disease-associated AIRs within an AIRR also effectively function as therapeutics against infectious agents, and may be a valuable resource for the development of therapeutic drugs like monoclonal antibodies [50,51].

Immune repertoire diversity and overlap

While the potential diversity of AIRs is enormous (10^{15} for TCRs [13], 10^{18} for BCRs [14]), the number of unique AIRs in the human body is several orders of magnitude smaller, with estimates around 10^8 – 10^{10} [52–54]. The clonal counts within an AIRR (number of cells belonging to the same clone) follow a heavy-tailed frequency distribution, meaning that the vast majority of clones have a very low clonal count, and only a small minority of highly abundant clones exist [55–57]. This poses challenges in AIRR research, as low-frequency clones are unlikely to appear consistently across multiple biological samples, even when taken from the same individual. Our ability to observe AIRR overlap is thus highly dependent on sample size [58,59]. Many analytical strategies for quantifying AIRR diversity and overlap have been borrowed from ecological studies of biodiversity [55,60–64]. However, since the clonal frequency distributions tend to be more extreme than species frequency distributions, the degree of undersampling is much greater in AIRR data than in species frequency data [60].

The fraction of so-called *public* AIRs occurring in the AIRRs of multiple individuals is thought to be around 1% [14,65,66], whereas the rest of the AIRs are *private*. Despite only a small fraction of the AIRR being public, the overlap is still high compared to the expected overlap based on the total potential diversity and actual AIRR size [58]. The presence of public AIRs can be explained by several different processes, including *convergent recombination* and *convergent selection* [67,68]. Convergent recombination refers to the fact that certain AIRs have a higher generation probability than others. Such ‘common’ AIRs may be created many times independently within and across individuals, and therefore likely to show up in AIRR sequencing experiments [67]–[71]. Conversely, convergent selection is the process where the same AIR is abundantly present in multiple individuals due to the presence of a common antigen [36,67,68,72]. Knowledge about the V(D)J recombination probabilities can thus help identify those AIRs in a repertoire that are actively involved in an immune response, for example by identifying AIRs that are more abundantly present than expected by their recombination probability [67], or by identifying clusters AIRs with more high similarity neighbours than expected by chance [73].

Immune repertoire sequencing and data sharing

Recent developments in high-throughput sequencing technologies have allowed researchers for the first time to investigate the complexity and diversity of the AIRR at a large scale [74]. Generally, AIRR sequencing (AIRR-seq) can be divided into bulk sequencing and single-cell sequencing [75]. Single-cell sequencing allows for a more detailed investigation of not only the frequency distribution of AIRs, but also additional per-cell transcriptomic information [75,76]. However, these techniques are significantly less high-throughput and more expensive than so-called 'bulk' sequencing. Therefore, bulk sequencing still tends to be the more commonly used sequencing method for AIRRs [77]. Because the two chains encoding each AIR are encoded on separate genomic loci, it is impossible to retain the chain pairing information when bulk sequencing is used [78]. Therefore, bulk sequencing datasets often contain exclusively TCR β or BCR heavy chain sequences. Single-cell sequencing can be used to maintain the pairing between the two AIR chains, as the individually sequenced loci can be traced back to the same cell.

Software like MiXCR [79] is typically used to translate a set of raw sequencing reads into a set of quantitated clonotypes. A clonotype is then represented as a CDR3 amino acid sequence combined with a V and J gene annotation (either for a single chain or for both paired chains) and clonal count. Such a representation of AIRR data can be used to share AIRR datasets for various computational analyses, and the AIRR community has defined a standardised format for such data [80]. Datasets can be shared and queried through several different databases. VDJdb [81,82] and McPAS-TCR [83] both contain TCRs annotated with their cognate antigen. The immune epitope database (IEDB) [84–87] is an epitope-centric database, but contains a collection of curated epitope-specific AIR sequences as well. The iReceptor platform [88] contains a collection of BCR and TCR repertoire datasets from various repositories, which may be annotated with immunological status.

Machine learning

The field of ML concerns self-learning algorithms which can be used to find patterns in datasets, and use these learned patterns to categorise or make predictions about items in new datasets. While the term 'machine learning' was already coined in the late fifties [89], ML research has had a tremendous boost throughout recent decades [90]. In particular, the development of methods to feasibly train so-called 'deep' neural networks have enabled us to construct methods for a wide range of complex prediction tasks, ranging from everyday applications such as personalised content recommendations and voice assistants, to the discovery of new drugs and analysis of medical data. ML has become an integral part of modern day life, and bioinformatics sequence analysis has been shaped by the rise of ML as well [91–94].

The purpose of this chapter is to provide a general introduction to ML, outline the terminology, and discuss some specific ML concepts that are relevant to AIRR research. An in-depth description of specific ML algorithms is out of scope for this thesis. For such technical details about ML, I kindly refer you to the books 'Deep Learning' [95] and 'The Elements of Statistical Learning' [96].

Supervised and unsupervised machine learning

Two of the main subcategories of ML are supervised and unsupervised learning [95,96]. Both supervised and unsupervised learning can be used to analyse a dataset consisting of *examples*. Supervised learning involves training ML models using labelled data, in order to make predictions for new data. The *labels* represent the outcome that we would like to be able to predict with a trained ML model. For instance, a dataset could consist of many AIRs, which are the examples, and the label could represent the antigen binding status of each AIR to an antigen of interest. Classification is a type of supervised learning where the labels are categorical values (*classes*), whereas regression aims to predict continuous numeric values. Depending on the particular ML algorithm, classification may be defined as a binary problem or a multi-class classification problem if more than two classes are present.

Unsupervised learning can be used to find relationships and patterns in data without any particular target label. Clustering is a form of unsupervised learning, which places relatively similar examples together in clusters [97]. Typically, a type of similarity metric is defined to compute the distance between various examples in the dataset. Note that although the unsupervised learning algorithm does not make use of labels, these labels may still be used to evaluate the performance of the algorithm afterwards [98]. One can for instance compute how often the examples in a dataset are clustered together with other examples that share the same label.

Multiple instance learning

Multiple instance learning (MIL) is a special case of supervised learning where each example (named *bag*) consists of a set of *instances*, and a label is provided for the entire bag. The class of a bag is dependent on one or several instances inside it. While MIL was originally formalised for the prediction of drug activity [99], it has since been used for a wide range of applications [100], including the classification of AIRRs [37,101–103]. In this case, the AIRRs are considered to be bags and the AIRs within are the instances.

Traditionally, MIL functions under the standard assumption that in order for a bag to be positive, it must contain at least one positive instance. These positive instances are called witnesses, and the proportion of positive instances in a bag is the *witness rate*. Under this assumption, the negative bags contain only negative instances. As long as one positive instance is found, the bag can be correctly classified as positive, and it is thus not necessary to identify all positive instances. Alternatively, the collective MIL assumption states that more than one instance is needed to identify a positive bag. For example, if the bag label relies on a combination, distribution or accumulation of positive instances [100,104].

Carbonneau et al. [100] propose to categorise the different challenges related to MIL into four groups: prediction level, bag composition, data distribution and label ambiguity. Prediction level refers to whether the objects of classification are the bags or the instances within. Instance level classification is very challenging, since labels are only available on the bag level. Under the standard MIL assumption, a perfect bag classifier may not be able to classify every instance correctly, whereas a perfect instance classifier also classifies the bags correctly. Secondly, bag composition describes the witness rate and possible relations between instances in a bag. A higher witness rate results in an easier classification problem. However, in some cases the witness rate of the negative bags is not zero, thereby adding an extra source of noise. Furthermore, the presumed instance labels may be dependent on meaningful relations between instances in a bag. For AIRR data for example, an AIR may be more likely to be part of an active immune response when a cluster of highly similar AIRs is found [73]. Thirdly, both positive and negative instances follow some kind of data distribution. In the simplest scenario, the positive instances could all bear some degree of similarity to each other, and thus all cluster together. Alternatively, there could be many highly dissimilar clusters of positive instances, making it more challenging to learn the defining features of a positive instance. The distribution of the negative data may not be clearly defined, and some methods therefore only use an internal model of positive instances. Finally, the label may be ambiguous, for example due to noisy label definitions or instances that do not fall into clearly defined categories.

Training supervised machine learning models

A supervised ML model can be viewed as a function mapping a vector of input values to a predicted output value: $f(x) \rightarrow \hat{y}$, where $\hat{y}$ aims to approximate the true label y [95]. The behaviour of this function is determined by a set of internal *parameters* (sometimes called weights). When training a supervised ML model, the goal is to find a set of values for these internal parameters that make the function predict output values that are as close to the true target labels (y) as possible. A *loss function* can be defined to represent how close the predicted value was to the actual value. For example, for numeric values, a simple loss function could be the square of the difference between actual and predicted: $(y - \hat{y})^2$. The aggregated loss function for a set of training data is called the *cost function*.

Since the parameter space can be extremely large, it is not feasible to test out every possible combination of parameters to find the optimal parameter set for the model. Optimisation algorithms can be used to explore the parameter space in a more computationally efficient manner. One of the most commonly used optimisation algorithms in modern ML for minimising the cost function is stochastic gradient descent (SGD) [95]. With SGD, the model parameters are usually initialised with some random values. At each step, the gradient (slope) of the cost function is computed based on the training data or a random subset thereof. The parameters of the model are then nudged a small step in the direction where the gradient has the steepest descent, i.e., the direction that minimises the cost function. This process is repeated many iterations until the algorithm converges. The learning rate determines how much the model parameters can be changed at each iteration of SGD. A faster learning rate may help the model converge faster, but a too fast learning rate causes the model parameters to be changed too much, potentially overshooting the minimum of the cost function. SGD does not guarantee to find the global minimum of the cost function, but is very likely to at least converge to a local minimum [105]. For the ML model, this means it may not be possible to find the overall optimal set of parameters, but a comparatively good set of parameters is usually found.

Aside from the internal parameters of the model, which are optimised during training, the model also has *hyperparameters*. Whereas the model parameters directly determine the output of the model, hyperparameters define the type of ML model and control the learning process. For example, when defining a neural network, there are several hyperparameters that need to be set in order to define the architecture of the network, such as the number of hidden layers or the number of nodes per hidden layer. The learning rate of SGD is a hyperparameter as well.

Encoding data for machine learning

ML models require the examples in the dataset to be represented in a numerical form. Translating raw data into such a numerical form is called an *encoding*. Encoded data consists of a vector of feature values, which are typically binary or floating point values. For example, an amino acid sequence (such as a CDR3 sequence) can be translated into a numerical representation through one-hot encoding. Since there are 20 amino acids in total, a one-hot encoding entails translating each amino acid to a vector consisting of 19 zeroes and a single one, where the position of the one is unique for each amino acid. Another type of encoding commonly used for biological sequences is k-mer frequency based encoding [106]. K-mers are subsequences of length k, and k-mer based encodings represent a sequence, or set of sequences, by the relative frequency of each possible k-mer (given some value k) occurring in the example. An encoded dataset is typically represented as a *design matrix* where each row corresponds to one example, and each column corresponds to one feature [96]. The choice of the type of encoding is of equal importance to the performance of the model as the choice of ML algorithm. Similarly to ML algorithms, encodings too have hyperparameters, such as for example the size of k for k-mer based encodings.

Performance evaluation and selection of machine learning models

To assess the overall performance of an ML model, a performance metric is computed. Examples of such performance metrics are the root mean squared error for regression, or accuracy for classification problems. However, for an ML model to be truly useful it should not only have a high performance on the same data it was trained on, but generalise to new, unseen data. The ability for an ML model to generalise can be tested by using a held out test set. The examples in the training and test data are typically assumed to be independent and identically distributed (IID) [95]. Identically distributed refers to the data being drawn from the same underlying data-generating process, whereas independent means that examples are independently drawn from the underlying distribution, and the presence of one example thus does not imply any information about which other examples are present in a dataset. The test set performance provides an estimate of the performance of the model on other unseen IID data.

When training an ML model, there is a balance between underfitting and overfitting (Figure 3). An underfitted model has not (yet) learned the patterns in the data, resulting in poor performance on the training set. If the model that is being trained is too simple, it will not be able to learn the patterns of interest in the data, no matter how much training data is supplied. In this case, the *capacity* of the model to learn the underlying function of interest can be increased by for example increasing the number of model parameters [95]. Contrary, an overfitted model may have excellent performance on the training set, but generalises poorly to new data. Overfitting can occur when an ML model is trained for too long, or when a too complex model (too high capacity) is used. In these cases, the model 'memorises' properties of the training data that are not representative of other IID data. Overfitting is tightly linked to the concept of bias-variance tradeoff [95]. The two sources of error in a predictive model are bias and variance. The bias relates to the assumptions made by the model. A model with high bias will produce an average prediction that differs substantially from the target prediction. Variance on the other hand relates to how different the model predictions are as a consequence of fluctuations in the training set. Underfitted models have high bias, whereas overfitted models have high variance.

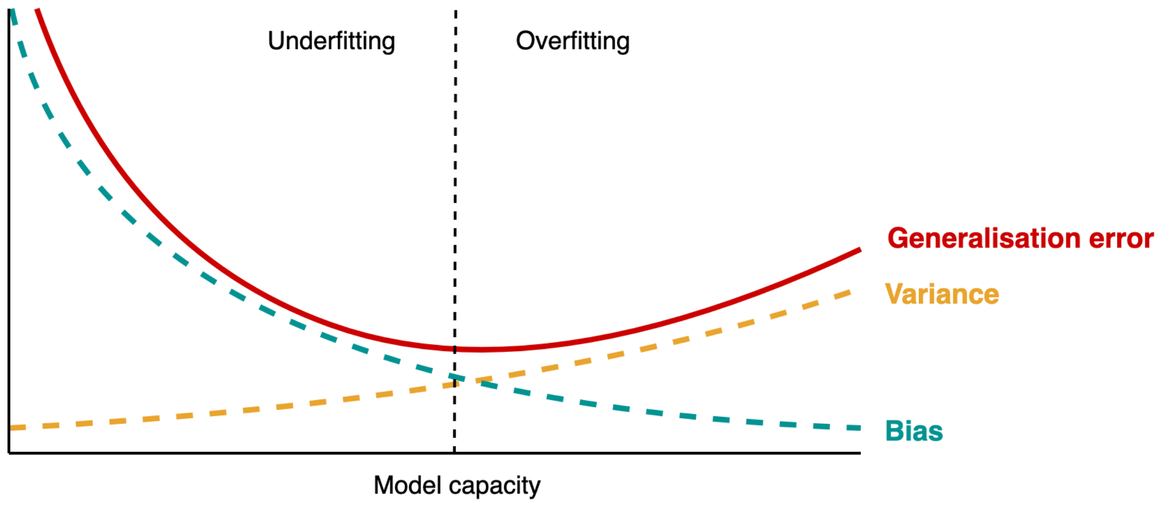


Figure 3: The bias-variance tradeoff. The generalisation error of a model can be decomposed into two sources of reducible error: bias and variance. As the capacity of the model increases, the bias tends to decrease while the variance tends to increase. Underfitting happens when the capacity of the model is too low to learn the patterns of interest, whereas a model with a too high capacity is prone to overfitting. Figure modified from: Goodfellow and Bengio [95].

Regularisation

To combat overfitting, a wide variety of different regularisation techniques can be applied [95]. Regularisation refers to any modification we can make to the ML algorithm to improve generalisation to new data, without trying to improve the performance on the training data. Applying regularisation also causes the learned solutions to be simpler. For example, some regularisation techniques introduce a weight penalty to the cost function, meaning that large weights are penalised. As a result, the trained models tend to learn smaller parameter values. Some parameters may even be entirely reduced to zero, effectively removing the influence from associated features to the model prediction. This penalty term serves as a hyperparameter that determines the degree of parameter shrinkage.

Another form of regularisation which aims to combat overfitting due to overtraining is early stopping using a validation set. This validation set is independent from the training and test sets, and allows for the generalisation error to be estimated without consulting the test data.

Throughout training, the performance on the validation set is tracked, and once the validation error starts to increase compared to the training error, training is halted.

In a similar manner, validation sets can be used to compare the performance of models trained with different hyperparameters, in order to select the optimal hyperparameters for the model. Since hyperparameters often control the learning process of an ML algorithm, their optimal values cannot be estimated from the training set, as this is likely to result in choosing hyperparameters that maximise overfitting to the training data. While the model parameters are not directly estimated using the validation set, it is important to note that because the validation set is used to select the hyperparameters, the performance on this set is an overly optimistic estimate of the true performance of the model on new data [95]. A separate test set should be used for this purpose.

Cross-validation and nested cross-validation

When splitting a dataset into subsets for training and testing, the majority of the data is typically used for training, and a rule of thumb is to set aside around 20% of the entire dataset for testing. However, if the total number of examples in the dataset is small, the performance results obtained on such a test set may be statistically uncertain. One possible solution for this problem is to repeat training and testing of ML models several times, using different partitions of the data as test sets, such that an average test error can be computed across all test sets. This strategy is called cross-validation [95]. A commonly used variant is k-fold cross-validation, where the total dataset is split into k subsets (for example 5), and each subset is used as a test set once, while all other k–1 subsets are used as training data. With k-fold cross-validation, each example is guaranteed to occur in the test set exactly once. Alternatively, data splitting can be done by selecting a fixed percentage of random examples as test data. If the dataset is particularly small, leave-one-out cross-validation can be used, in which case a single example is used as a test set for each iteration.

Cross-validation is appropriate when evaluating the performance of individual models with a preselected set of hyperparameters. However, when hyperparameters should be optimised in addition, standard cross-validation can lead to overoptimistic results, as the selected optimal hyperparameters may be biased towards the test set. To overcome this issue, nested cross-validation should be used instead, which consists of an inner “selection” loop, and an outer “assessment” loop. In the selection loop, the training data is split further into several training and validation sets. After training models with a range of different parameters on each training set, the optimal hyperparameters are determined based on the validation set performance. Using these optimal hyperparameters, a new model is trained using data from the combined training and validation set. In the assessment loop, the performance of all retrained models is evaluated on the test sets. By separating the hyperparameter selection from performance assessment, a nearly unbiased estimate of the model performance can be obtained [107].

Machine learning analysis of immune receptor and repertoire data

AIRR sequencing datasets can be large, containing hundreds of thousands or even millions of sequences per individual. Furthermore, the underlying mechanisms determining the antigen binding capabilities of individual AIRs are not yet thoroughly understood [47,108]. ML is particularly well-suited to analyse such large and highly complex datasets. Broadly, the field of AIR(R)-ML can be divided into making predictions about individual AIRs, or making predictions on the full AIRR. In principle, ML can be used to predict any property of an AIR or AIRR as long as a suitable labelled dataset is available. But most typically, the goal is to learn whether AIRs bind to a particular antigen target of interest, or predicting the immunological (disease) status of an individual based on their AIRR. ML can thereby further the development of therapeutics [51] and diagnostics [109].

Predicting antigen binding of immune receptors

The AIR-antigen binding prediction problem can be formulated in a variety of different ways. While AIR-antigen binding is technically defined by binding affinity (a continuous value), in practice it is usually treated as a classification task. This can be specified as a binary classification task, where binders need to be distinguished from non-binders, or a multi-label classification task where the most probable target is predicted from a multitude of epitopes [110]. The definition of negative data in a binary classification task furthermore has a large impact on what can be learned. For instance, one could learn to distinguish AIRs that bind to two different targets (one versus one), or retrieve the binding AIRs from a pool of mixed specificity background AIRs (one versus many) [111].

Both supervised and unsupervised learning methods have been applied to the AIR-antigen binding problem [112]. Unsupervised methods aim to cluster sequences together in order to find shared-specificity clusters. One of the challenges in predicting AIR-antigen binding is however that AIR specificity is not solely defined by overall sequence similarity, as highly dissimilar sequences have also been observed to bind to the same target [36,108,113]. Supervised methods on the other hand aim to learn a more refined decision boundary between AIRs belonging to different classes. Some of the key methods that have been developed for unsupervised or supervised prediction of AIR specificity are described in the next two sections.

Unsupervised clustering of immune receptors

In the absence of epitope specificity information for each individual AIR in a dataset, clustering of AIR sequences can be done in order to identify shared specificity clusters within a set of AIRs [108]. The inference of antigen binding specificity through unsupervised clustering has primarily been applied to TCR rather than BCR data. TCRs may be easier to cluster due to the absence of SHM. BCR clustering is instead often used for the prediction of clonal lineages [114].

In order to find clusters of TCRs with a high likelihood of sharing antigen specificity, Glanville et al. [36] developed GLIPH, which clusters TCRs based on the similarity of the CDR3 in the TCR β chain. Furthermore, HLA restriction can be predicted based on the identified clusters. GLIPH employs two complementary strategies to cluster sequences: local and global similarity-based clusters. Global similarity clusters are defined as a sequence containing a single flexible amino acid. Contrary, local similarity clusters are only required to share a single contiguous k-mer, which is intended to capture the paratope within the CDR3. The first version of GLIPH was known to suffer from the so-called ‘small-world’ effect when too large datasets were analysed. An improved version of the algorithm, GLIPH2, was introduced later by Huang et al. [115], which was designed to be able to process larger datasets.

Dash et al. [116] developed TCRdist, which serves as a direct distance metric between paired α-β chain TCR sequences. TCRdist does not only take the CDR3 sequence information into account, but also CDR1 and CDR2, as well as an additional variable loop which was identified between CDR2 and CDR3. Still, a higher weight was given to the CDR3 compared to the other CDRs, due to its important role in antigen binding. By using several sets of TCRs with known specificity, TCRdist could be used as a distance metric for a nearest-neighbour classifier, where the TCR specificity is predicted to match the specificity of the set with most similar receptors. Mayer-Blackwell et al. [117] improved upon the TCRdist algorithm and introduced TCRdist3, where the key new feature is the use of meta-clonotypes. These meta-clonotypes are defined by a centroid TCR, a TCRdist radius around this centroid, and optionally a CDR3 motif. Meta-clonotypes can be used to search for same-specificity TCRs in bulk TCR repertoires, which may be used as biomarkers for antigen-specific immune responses similarly to the specificity clusters found by GLIPH and GLIPH2.

As TCR clustering is increasingly applied to larger datasets, not only the biological similarity of the AIRs but also the efficiency of clustering algorithms needs to be prioritised. The TCR CDR3 clustering tool iSMART [118] was developed as a faster alternative to GLIPH and TCRdist. However, iSMART was outperformed in running time by GLIPH2 [115,119]. ClusTCR is an ultrafast TCR clustering tool which implements a multi-step algorithm where TCRs are first divided into superclusters based on physicochemical properties, which are later refined to more accurate specificity groups [119]. ClusTCR was shown to run 50 times faster than GLIPH2 on a dataset of 1 million TCRs. In their study, Valkiers et al. [119] assessed the quality of the produced specificity clusters based on a multitude of metrics, and found that no individual TCR clustering algorithm distinctly outperformed all others, as each algorithm had its own advantages. GIANA is another computationally efficient TCR clustering tool which outperformed GLIPH2 in speed (approximately 10 times faster), but was not directly benchmarked against ClusTCR [120].

The success of TCR CDR3 similarity-based clustering methods inspired the development of TCRMatch, which was not designed as a clustering algorithm itself, but rather assigns CDR3 sequences a putative epitope based on the existence of a similar TCR in the IEDB [121].

Supervised methods for predicting antigen specificity

One of the early applications of deep learning to predict the antigen binding capabilities of TCRs was the method DeepTCR [103]. In order to learn meaningful features of TCRs, DeepTCR employs variational autoencoders (VAEs). These VAEs learn a function which first maps each input TCR to a latent representation of this TCR in a dimensionality reduced space (encoder), and then reconstructs the original TCR from this latent representation (decoder). Thus, the latent representation contains informative features of the TCR, but in a compressed format. The trained ‘encoder’ part of the network is then used to compute latent TCR representations. These latent representations are used as a starting point for unsupervised clustering and used as an input for ML models predicting TCR-antigen binding.

In order to train deep learning models to predict antibodies binding to a breast cancer associated antigen, Mason et al. [113] designed a large mutagenesis-based antibody dataset containing over 10.000 binders and 25.000 non-binders. This dataset was created in a two-step process. First, a previously developed therapeutic antibody was mutated in 10 positions throughout the heavy chain CDR3 in order to find out which mutated variants retained their ability to bind. Next, the mutagenesis dataset was created by sampling from the distribution of positional amino acids that tended to occur among the binders. This resulted in a large dataset of sequences varying exclusively across the 10 positions in the CDR3, while the rest of the antibody was kept constant. Mason et al. trained a variety of different ML models, ranging from shallow models such as logistic regression or k-nearest neighbours, to deep learning models like long short-term memory recurrent neural networks (LSTM-RNN) and convolutional neural networks (CNNs). The CNN-based models outperformed all other models.

All supervised and unsupervised specificity prediction methods discussed up to this point exclusively focus on the TCR sequences, without taking additional information about the properties of the epitope in account. Contrary to this, some ML methods have been developed which incorporate epitope sequence information. Examples of this include: NetTCR, which uses a shallow one-dimensional CNN [122,123]; ERGO, which uses LSTM and autoencoders [124,125]; ImRex, which creates image-like interaction maps between TCR and epitope sequences that are classified using a two-dimensional CNN [126]; and TITAN, a bimodal neural network [127]. Rather than treating the epitope as a binary binding versus non-binding label, these models aim to learn the underlying interactions between TCR and its target. The underlying thought is that such a model may be able to generalise to be able to predict whether TCRs bind to novel, unseen epitopes. However, a lack of representative data, and in particular an underrepresentation of the variety of different epitopes, is often reported to be the main challenge for this prediction task [112,122,127], as well as poor quality of the existing data [123]. Furthermore, constructing an appropriate set of negative (non-binding) examples of TCR-epitope pairs is not trivial, yet is crucial for the model performance [46].

Another way to incorporate epitope information into AIR-antigen binding prediction is by considering 3D structural data of AIRs bound to their cognate epitope. Akbar et al. [38] showed that antibody-antigen binding adheres to a limited set of structural paratope-epitope interaction motifs, suggesting that antibody-antigen binding in 3D space adheres to learnable rules. Several methods for AIR-antigen binding prediction based on 3D structural data have been created [128,129]. It should however be noted that the number of 3D crystal structures of antigen-bound AIRs remains very limited, as the experimental methods for collecting such data are low-throughput, time consuming and expensive [38,130,131].

Predicting disease status for immune repertoires

Since any immunological response leaves a unique mark on the AIRR, it has been suggested that AIRR-seq data could be used as a universal diagnostic test for infectious diseases, autoimmune diseases or cancer [109]. Using a single blood test, a variety of different immune states could potentially be identified [132]. One of the main challenges of ML-based AIRR diagnostics has been the massive diversity and lack of overlap between AIRRs [47]. Other challenges include technological artefacts and confounding factors that may hamper the ability of the ML model to learn the correct disease-related signals [47,133,134].

To avoid the issue of lack of overlap between AIRR samples, some of the early approaches for the ML-based classification of AIRRs instead relied on AIR sequence-independent features to make predictions about immunological status. For example, the clonal frequency distribution of an AIRR can be described by diversity profiles, which have been used as features for ML classification into healthy, vaccinated and infected categories [135]. Here, diversity refers to Hill-based diversity metrics [136], which have been commonly used in biodiversity studies and were later introduced in AIRR research [60]. Such diversity metrics rely on a weighting parameter α, and by computing the diversity for a range of different α values, a profile can be constructed. The size of the diversity profile is determined by the number of α values, rather than the number of clonotypes in the repertoire, thus allowing for direct comparison between clonal frequency distributions even if the number of clones differs [135]. Other works have computed k-mer frequency distributions of AIRRs and used these as a basis for AIRR classification. Even if AIRRs do not share any public AIRs in their entirety, it may be possible to find shared k-mers that are associated with a particular immunological status [137–139].

As only a fraction of the AIRs inside an AIRR are responsive to the antigens associated with the disease of interest, AIRR classification can be viewed as a classical MIL problem [140], [141], where the AIRR is a bag and the instances are the AIRs within. In particular, the AIRR MIL problem is challenging due to the very large number of instances and a low witness rate. Furthermore, it is possible for there to be disease-associated (antigen binding) AIRs present in a healthy AIRR, for example if these AIRs are cross-reactive with another epitope.

Emerson et al. [72] sequenced 786 TCR β repertoires with either positive or negative serostatus for cytomegalovirus (CMV), which is still one of the largest AIRR-seq datasets to date. The large dataset size, both in terms of number of individuals and per-repertoire sampling depth, allowed for the development of a MIL classifier relying on public TCRs. They built a probabilistic model that detects public TCRs which are associated with CMV positive serostatus, and classify new repertoires based on the relative abundance of such CMV-associated TCRs. This method could also be used to predict the HLA alleles of subjects. An independent study furthermore exemplified the high generalisability of the CMV-associated TCR β sequences, by successfully classifying the CMV serostatus of a new cohort of 33 individuals [142]. It should be noted that because this method relies on the presence of public disease-associated sequences, it is highly dependent on a sufficient sampling depth, number of AIRRs, as well as a high fraction of disease-responsive AIRs. CMV is known to induce a particularly strong immune response, thus leaving a clear mark on the AIRR [143], which may not be the case for other diseases.

In order not to rely on public full AIR sequences, Ostmeyer et al. [37] developed a MIL method which classifies TCR repertoires based on the presence of one or few tumour-associated k-mers (subsequences of length k). Here, instead of considering full AIRs to be instances in the MIL setting, k-mers are treated as instances. Since many different AIRs could share the same k-mer, there naturally exists a larger number of public k-mers compared to public AIRs. K-mer length 4 was chosen as it corresponds to the length of the paratope in a set of 3D crystal structures of TCRs contacting peptide-presenting MHC. The k-mers in this study are represented by their biophysicochemical properties, such that amino acids with comparable properties are considered synonymous. A modified logistic regression model is used which is modified such that it requires the k-mers to occur at least once in every tumour TCR repertoire, and be absent from all healthy repertoires. In a later study [102], the method was updated to consider k-mers containing a single gap position, and applied to a new cancer dataset. While the method obtained a high classification accuracy (93-95% when assessed with patient-holdout cross-validation), it should be noted that the dataset sizes were small, and the method performed notably worse in a later benchmark [101].

The deep learning model DeepRC was developed to classify AIRRs, and internally uses a transformer-like attention mechanism to learn the importance of the individual AIRs within [101]. This importance can be represented as a weight per AIR, which are not only used by the model to make predictions, but also improve interpretability. DeepRC outcompeted both Ostmeyer [37] and Emerson [72] MIL models on simulated data as well as the experimental CMV dataset.

The previously mentioned tool DeepTCR supports in addition to receptor-level classification also repertoire-level classification through a MIL approach [103]. Here, the same VAE-based latent TCR representation is used to represent each of the TCRs in the repertoire, and an attention mechanism is used to infer the important subset of TCRs to classify the repertoire. In the original study, DeepTCR was not used as a disease classifier, but rather to classify in vivo repertoires derived from T cells that were cultured together with antigens, in order to learn antigen-specific signatures. In a later study, DeepTCRs MIL classifier was furthermore shown to be able to classify severity of SARS-CoV-2 infection based on TCR repertoires [144].

Benchmarking of ML methods using simulated data

While few large experimental AIRR datasets exist, simulated datasets should be considered equally important for the benchmarking of computational methods [145]. Besides being able to simulate datasets of unconstrained size, the benefit of simulated data is that the ground truth signals can be defined, which are unknown for experimental data. Knowledge of ground truth signals allows us to benchmark not only the predictive performance of ML models, but also investigate whether the learned predictive patterns match the signal we intended the model to learn. Furthermore, with simulated data the parameters of the ground truth signal can be controlled, allowing us to benchmark ML models over a range of different immune signal complexities [146]. This makes it possible to test the limits of what different types of ML methods can learn given different parameterisations of the ground truth signal.

Immune signals

In AIR(R)-ML, the term immune signal is used to denote the set of features that defines the immunological status of an AIRR, or the antigen binding status of an AIR [47]. This immune signal is the ultimate underlying pattern that we aim to learn through ML. However, the shape and complexity of the immune signal are still not fully understood. We know that AIR-antigen binding is largely determined by the BCR heavy or TCR β chain CDR3 [39,40], and takes place in 3D space [32,33]. Yet more research is needed to better understand what types of patterns in the CDR3 amino acid sequence are most clearly associated with antigen binding status when structural information is not available, and the degree of dependency on other properties of the AIR besides the CDR3.

The AIRs present in an AIRR are selected and clonally expanded over the course of an individual's lifetime in response to the immunological events that the individual has experienced. It can therefore be assumed that at the AIRR level, the immune signal presents itself in a subset of disease-associated AIRs. These AIRs may be characterised by their ability to bind to one of multiple disease-related antigens, although the particular antigens need not be known. The degree of clonal expansion of individual AIRs, and the presence of clusters of highly similar AIRs could thus be part of an AIRR-level immune signal [73]. Furthermore, the genetic background of an individual shapes their AIRR to a large degree. For example, MHC restriction or personal biases in V(D)J recombination contribute to a large degree to which AIRs are likely to be present in an individual's repertoire [133]. This may result in differences in the immune signal across different individuals. An improved understanding of immune signals can help us create more realistic simulations and therefore better benchmarking of ML models, as well as inspire us to design feature encodings and ML architecture that better capture the immune signal.

Methods for the simulation of immune receptors and repertoires

Simulation of AIRs can be done by modelling the V(D)J recombination events. For this purpose, IGoR was developed [16], which learns the V(D)J recombination and SHM statistics from unproductive genomic rearrangements in AIRR-seq data. Since these sequences do not produce a functional AIR, they represent the distribution of AIR sequences prior to selective pressure. In practice, AIRR data is typically analysed at the amino acid rather than the nucleotide level. OLGA was later developed as a faster alternative to IGoR [17]. OLGA directly predicts the recombination events based on amino acid sequences, rather than nucleotide sequences, thus being more computationally efficient as one amino acid sequence could be derived from many potential nucleotide sequences. Both IGoR and OLGA can be used to simulate new AIR sequences, as well as to compute the generation probabilities of existing AIRs [16,17]. These models can be further extended to simulate AIRs which have undergone selective pressure. Software tools SONIA and soNNia internally use IGoR, and add a second component for modelling antigen binding selection using a linear model and deep neural network, respectively [147,148]. To calibrate these models, distributions of unproductive (and thus unselected) sequences are compared to the productive sequences in a given AIRR dataset.

Besides aiming to simulate AIRs according to their observed distributions in AIRR datasets, several other simulation tools have been developed for the benchmarking of specific computational methods. IgSimulator simulates both antibody repertoires and the sequencing reads derived from them, in order to benchmark the reconstruction of antibody repertoires from raw sequencing data [149]. AbSim simulates the evolution of clonal lineages in antibody repertoires to test clonal lineage reconstruction methods [150]. immuneSIM provides a broad and flexible framework where the user has full control over the parameters shaping the antigen-inexperienced repertoire, and allows for the implanting of sequence motifs to represent disease status [151]. This allows for immuneSIM to be used for a range of different benchmarking use cases, most notably, for ML-based prediction of immunological status represented by ground truth sequence motifs. To benchmark ML methods for the prediction of antibody-antigen binding based on 3D structural data, the Absolut! software can be used to generate synthetic antibody-antigen complexes [131]. These synthetic complexes are based on a simplified lattice structure, and provide ground truth information about the epitope, paratope and binding affinity, according to this lattice structure model.

Present work

Research aims and context

The application of ML methods for the prediction of AIR-antigen binding status or the disease status of an AIRR has shown promising results. Nevertheless, the field of AIR(R)-ML is also subject to challenges. Independently developed models are not consistently benchmarked against each other, and reproducibility of studies is not always achievable. Differences in performance evaluation strategies, along with a lack of interoperability between tools further complicate the comparison of ML model performance results. The underlying signals determining the antigen binding status of AIRs or disease status of an AIRR are highly complex and not yet thoroughly understood. Therefore, not only the development of ML models with a good prediction performance is of interest, but understanding the underlying signals learned by the models is of equal importance. Furthermore, with the increase of available data, computational efficiency can become a bottleneck. The overall aim of this thesis is to provide new computational methods to aid the ML-based analysis of AIR(R) data. Three sub-goals are defined as follows:

Developing a platform for the machine learning analysis of adaptive immune receptors and repertoires. While AIR(R)-ML methods have been proposed, they have often been developed under different domain assumptions, and their performance is presented using different evaluation strategies or datasets, hampering our understanding of which method is ideal for a given study. New AIR(R)-ML methods should be benchmarked against a wide variety of competing or baseline methods using both experimental and simulated datasets, but this can be a challenging and time consuming task. **Paper 1** therefore presents immuneML, a platform implementing all functionality needed for the thorough evaluation and comparison of AIR(R)-ML models.

Developing an ultra-fast tool for the comparison of sets of adaptive immune receptors. Many computational applications of AIR(R) data, including AIR(R)-ML, involve computing the number of matching sequences between AIRRs or large sets of reference AIRs from databases. Advancements in high-throughput sequencing have resulted in a rapid increase in the number and sizes of AIRR datasets, yet previously developed methods for quantifying overlap between AIRRs scale poorly with increasing dataset sizes. **Paper 2** therefore introduces CompAIRR, an ultra-fast tool for computing exact or approximate sequence matches across AIRRs.

Investigating the ability of short motifs to distinguish antigen binding from non-binding antibodies. Several AIR studies have been performed under the implicit or explicit assumption that AIR-antigen binding is largely determined by the presence of a short motif in the CDR3. In order to directly examine this assumption, **Paper 3** introduces a method for the discovery of short motifs that are highly predictive of antigen binding. This method is applied to an antibody binder/non-binder dataset, to investigate to what extent antigen binding can be predicted using a set of short CDR3 motifs.

Paper summaries

Paper 1: The immuneML ecosystem for machine learning analysis of adaptive immune receptor repertoires

While there already exist ML software packages such as scikit-learn [152], PyTorch [153] and Tensorflow [154], they provide functionalities for general ML usage. As ML is increasingly applied in (biological) sciences, domain-tailored ML platforms for genomics [155], biomedicine [156] and chemistry [157] have been created. The complex biology of AIRR data has inspired the development of domain specific ML methods to predict the antigen binding status of AIRs or immune status of AIRRs [47]. However, these studies differ in the datasets that are used, ML method evaluation strategies and domain assumptions, hampering our ability to evaluate which methodology is optimal for a given prediction task. Paper 1 presents immuneML, an open source platform for the ML-based analysis of AIR and AIRR data.

immuneML implements all steps of the AIR(R)-ML process, such as data preprocessing and encoding, training and evaluation of ML models, and interpretability analysis. The immuneML platform is extensible, such that AIR(R)-ML researchers can easily integrate their own methods into the platform and re-use standard workflows such as nested cross-validation for ML method hyperparameter optimisation and comparison. To benchmark new methodology, a set of baseline ML methods are already provided, as well as various previously published ML methods for AIR and AIRR classification [37], [72], [101], [117], [135]. Furthermore, immuneML can simulate benchmarking datasets with implanted ground truth signals, and provide various graphical reports that can be used to investigate properties of data or of trained ML models. immuneML is available both as a python package and through a Galaxy web interface. All analysis details are specified in a YAML specification file, which is compatible with both the command line and Galaxy interface, ensuring reproducibility and shareability of analyses.

The applicability of immuneML is shown in three use cases. (i) Firstly, it is demonstrated that immuneML can be used to reproduce the AIRR CMV status prediction study by Emerson et al. [72]. In order to do this, the proposed probabilistic binary classifier was re-implemented and integrated into immuneML. Since the success of this method is highly dependent on dataset size, it is shown how immuneML can be used to automatically repeat the classification task with a smaller number of AIRRs in order to assess the robustness of a classifier. (ii) Secondly, the extensibility of the platform is demonstrated by integrating a new deep learning method for classifying paired chain AIRs. The new method is benchmarked against a shallow baseline ML method as well as TCRdist3, and learned motifs are compared to motifs learned by TCRdist3 and GLIPH2, to show that newly integrated methods can easily be compared to already-available methods. (iii) And thirdly, it is shown that immuneML can be used for large-scale benchmarking of ML methods. For this use case, the immuneML simulation functionality is used to implant ground truth motifs into a set of synthetic AIRRs generated by OLGA [17] to simulate disease signals with varying complexity. Several different encodings and ML methods are benchmarked for the prediction of these simulated diseases, and it is shown how graphical analysis reports can be used to get additional insight into how well the learned patterns overlap with the implanted ground truth motifs.

Paper 2: CompAIRR: ultra-fast comparison of adaptive immune receptor repertoires by exact and approximate sequence matching

In Paper 2 we present CompAIRR, an ultra-fast tool for identifying (nearly) identical AIRs across large AIR sets such as AIRRs. Computing overlap between AIRRs is a computational process that is part of almost any AIRR analysis. However, all-against-all AIR matching can be computationally expensive as the number of comparisons grows rapidly with larger AIRR datasets. As our sequencing technologies are becoming more high-throughput, the speed and memory efficiency of AIRR analysis algorithms become increasingly important.

Prior to CompAIRR, several computational suites (immunarch [158], immuneREF [159] and VDJtools [160]) already provided functionality for counting overlapping AIRs across AIRRs. However, these tools internally rely on straightforward implementations that do not scale well to large datasets. Furthermore, previously available implementations allowed AIRR overlap to be computed exclusively based on identical AIR sequence matches. Since similar AIRs are often thought to have similar antigen binding capabilities, identifying near-identical AIRs could be of interest as they may be involved in immune responses to the same antigen. CompAIRR is currently the only available tool for near-identical matching of AIRs at a large scale, by allowing a user-defined Hamming distance and up to 1 insertion or deletion.

To showcase the speed and efficiency of CompAIRR, we benchmarked its exact AIR overlap functionalities against the implementations provided by immunarch, immuneREF and VDJtools. On a large synthetic dataset consisting of 10.000 repertoires of 100.000 sequences each, CompAIRR finished in approximately 17 minutes while the fastest alternative tool took 10 days to complete the task. Moreover, the maximum memory usage of CompAIRR was consistently lower than ⅓ of the memory usage of each of the other tools.

CompAIRR is designed to have a flexible interface for easy integration into other tools or computational pipelines. As a proof of concept, CompAIRR was integrated into immuneML to speed up the computation of two encodings for ML, but it may in the future be used to improve the speed and memory efficiency of a much wider variety of use cases.

Paper 3: Predictability of antigen binding based on short motifs in the antibody CDRH3

While the exact rules determining antibody-antigen binding are unknown, one of the proposed assumptions is that AIR-antigen binding is determined by the presence of a short motif within the antibody heavy chain CDR3. Some studies have created AIR(R) classification or clustering models that are explicitly built on this assumption [36], [37], [102], [115], and the same assumption is reflected in the simulation of AIRR datasets by implanting k-mers into ‘diseased’ repertoires [146], [151], [159]. Paper 3 presents a methodology for identifying short motifs with high precision in determining antigen binding and high generalisability to unseen test data. This methodology is used to analyse to what degree antigen binding is driven by the presence of such short motifs, regardless of the remaining amino acids in the CDR3. We use a previously published combinatorial mutagenesis dataset of HER2 binders and non-binders (Mason dataset) [113], as well as a second dataset (Mehta dataset) that aims to reproduce the sequence distribution of the former dataset.

Within the context of this study, a short motif is defined as a combination of 2–5 amino acids that occur at specific positions in the CDR3 sequence, and may be separated by gaps. Given a user-defined precision cutoff on the training data (e.g., 0.8 or 0.9) for predicting antigen binding by motif presence, the method can be used to learn a recall threshold above which the motifs are likely to retain a relatively high precision on separate validation data. A final set of antigen binding motifs is then learned using the combined training and validation data, by applying the predefined precision and learned recall thresholds, and the performance of these motifs is evaluated on previously unseen test data.

The method is first validated using two simulated datasets. The first simulated dataset contains ground truth motifs which determine the antigen binding status of a sequence: if any of these motifs is present, the sequence is labelled to be a binder. Thus, in such a simulated dataset, the ground truth motifs have a perfect precision. We show that our method is able to recover these ground truth motifs as long as they occur with sufficient abundance in the dataset, and that all of the learned motifs overlap with at least one ground truth motif. In the second simulated dataset no ground truth motifs are implanted and a random antigen binding status is assigned, making it impossible to accurately predict antigen binding status through the presence of motifs. This dataset is used to show that our methodology correctly concludes that no generalisable high-precision motif exists in such simulated data.

Next, the method is applied to the Mason dataset, and it is shown that the learned motifs not only retain high precision on the held out test set (25% of the Mason dataset), but also on the independently generated Mehta dataset. The learned motifs consisted of 3–5 amino acids and spanned across the entirety of the sequence length. They occurred in highly variable sets of sequences, confirming that the presence of the motif itself rather than overall sequence similarity was predictive of antigen binding in these CDR3 sequences. It should be noted that many of the high-precision motifs identified by our method were dispersed, containing multiple gaps, whereas previous methods have often used contiguous k-mers [36], [37], [102], [115].

Finally, we constructed simple motif-based classifiers which predict a sequence to be a binder if it contains any of the set of motifs of interest. Using such a simple classifier, we were able to obtain a classification performance on the Mason test set that was nearly as good as the CNN designed for this dataset by Mason et al. [113], and even slightly outperformed this CNN on the Mehta dataset.

Discussion

This thesis is based on three papers. Paper 1 introduces the AIR(R)-ML platform immuneML, which forms the computational foundation of the work presented in this thesis. CompAIRR, introduced in Paper 2, is an ultra-fast tool for the computation of overlap between sets of AIRs, which has been integrated into several workflows to increase speed and reduce memory usage. Paper 3 presents a new computational methodology for the discovery of short motifs that are highly predictive of antigen binding.

Coherence between the computational tools and analyses introduced in this thesis

While the three papers comprising this thesis stand on their own, immuneML serves as the computational foundation for all the work presented in this thesis. The modularity of the immuneML platform makes it easily extensible, allowing many of the existing components to be reused for different use cases. The basic modular components in immuneML are: dataset types, preprocessing steps, encodings, ML methods, graphical analysis reports and data simulation steps. These components represent the basic building blocks of an analysis and can be used to execute a variety of instructions. The most important instruction is the training and evaluation of ML strategies through nested cross-validation. Other instructions include applying previously trained models to new data, exploratory analysis of (encoded) datasets through reports, or simulation of data with ground-truth disease signals.

CompAIRR was designed to be a dependency-free tool with flexible output format, making it easy to integrate into other tools in order to speed up core components. CompAIRR has been integrated into immuneML in order to speed up two encodings for AIRR classification, one ML method for AIR classification, and several related analysis reports. These AIRR classification components are the encoding for the method introduced by Emerson et al. [72] as well as a pairwise AIRR distance encoding for k-nearest neighbours classification. For these encodings, both the original version and a faster CompAIRR-based version are now available. The CompAIRR-based AIR classifier is a simple baseline method which classifies a sequence as positive if at least one similar sequence has been observed in the training data, where similarity is defined by a given Hamming distance. Such baseline methods are important for the benchmarking of AIR classification models, as it provides an estimate of the a priori difficulty of the classification problem and the true generalisation power of the classifier [161], and has been used for this purpose in Paper 3.

The computational analyses in Paper 3 were included in immuneML in the form of new encodings, ML methods and reports. For the motif encoding, a sequence dataset labelled with antigen binding status should be provided, and an efficient algorithm is used to search for every motif exceeding a given precision and recall threshold. This encoding is accompanied with an analysis report for calibrating these recall thresholds given a user-defined precision threshold. The encoding can also be used in combination with baseline ML methods such as logistic regression, or with the new ML method which classifies a sequence as positive if it contains any of the learned set of motifs. This ML method can furthermore be parameterised to search for a reduced set of motifs which retains high classification accuracy. Additionally, the previously published CNN by Mason et al. [113] was introduced as a new ML method in the immuneML platform as part of this work, as well as several analysis reports providing additional insight into the sequence distribution of the datasets or learned motifs.

Using the immuneML codebase as a foundation for the work presented in Paper 3 enabled the re-use of existing workflows for importing data, constructing a dataset object, performing exploratory data analyses, training ML models, optimising their hyperparameters and evaluating their performance. All figures shown in Paper 3 have been created using immuneML reports. The immuneML report interface is highly flexible and can be used to represent any compartmentalised analysis, ranging from plotting basic dataset statistics to complex multi-step analyses such as determining the optimal motif recall thresholds. Representing each sub-analysis as an immuneML report creates a natural organisation and readability of the code, yet also contributes to computational efficiency because intermediate results that are used by multiple encodings or reports can be retrieved through caching. Lastly, a major advantage of this immuneML integration is that every result can easily be reproduced since all parameters are stored in a YAML file.

Possibilities for extending the immuneML platform

Since the release of immuneML in 2021, the code base has been continuously updated to incorporate bug fixes, implement performance enhancements, and add few new reports and encodings. This section discusses several potential larger extensions that could be made to the immuneML platform. Some of these topics have been preliminarily explored by master students [162]–[167], but no official implementations have resulted from this.

While immuneML supports both AIR and AIRR classifiers, most of its current functionalities are aimed at AIRRs. The platform could be extended in several ways to improve the support for AIR classifiers. Firstly, since the predicted target in immuneML is assumed to be a categorical label, any epitope sequence information is always ignored. Yet many state-of-the-art AIR classifiers incorporate both AIR and epitope sequence information [122]–[127], and these methods currently cannot be directly integrated into the platform. Classification of AIR-epitope sequence pairs would require a new dataset type to be defined inside immuneML. Apart from serving as a common platform for the comparison of AIR-epitope classifiers, immuneML could be equipped with additional features aimed specifically at this new dataset type. For instance, since the definition of negative data examples is particularly challenging for this prediction problem [46], immuneML could provide functionality to supplement a set of binder examples with non-binder examples according to different strategies for generating negative data. This would allow researchers to not only benchmark their ML method against competing methods, but also to automatically map how each method performs using a range of differently defined negative datasets.

Another potential dataset extension in this line would be support for 3D AIR-antigen structural data. The number of experimental AIR-antigen 3D structures is currently still limited, but simulated data such as those generated by Absolut! [131] could kickstart the development of relevant methodology. Since flat AIR sequences can be extracted from 3D structural data, support for both data types within one platform could allow the direct comparison of ML models using either the structural or sequence-based representation of the same dataset. This could help us investigate what the additional value of the 3D structure is, and to what extent correct classifications can already be made using sequence information alone.

immuneML is equipped with a series of reports which assist in the understanding and interpretability of results. Some of these reports can be used to investigate general qualities of datasets, such as showing the distribution of sequence lengths or the per-position amino acid frequency distribution of CDR3 sequences in a dataset. Other reports provide explainability of trained ML models, like logo plots of CNN kernels, or bar plots displaying the coefficients of logistic regression and support vector machine models. However, these interpretability reports are specific to individual ML models or encodings. The more recently published pipeline DECODE [168] presents a model-agnostic approach for interpreting the behaviour of ML models for AIR classification. Based on the positive and negative predictions by a given model, DECODE constructs sets of logical if-then rules aiming to represent the underlying model. Downstream analysis and visualisation of such learned rules can provide insight into the decision process of a model. Furthermore, the model-agnostic nature of these rules creates opportunities for an interpretable comparison between ML models with entirely unrelated architectures. Currently, immuneML and DECODE fulfil complementary roles, and relatively few code changes would be needed to allow trained immuneML models to be imported into the DECODE pipeline. A streamlined integration allowing the user to automatically run DECODE-based interpretability analysis of immuneML models could be an invaluable enhancement to both platforms. DECODE focuses on interpretability of AIR classifiers, and integration with immuneML highlights the need for similar model-agnostic interpretability analyses on the AIRR level.

Other possible extensions include, but are not limited to, implementing the possibility to build regression models for the prediction of numeric values such as antigen binding affinity, adding functionalities for clustering, or generative modelling of data. As the number of different functionalities of the platform grows, it is important to consider its usability. While immuneML is extensively documented both in written and video form, additional guidance may be needed to help the user navigate the provided resources.

Ongoing developments in the simulation of AIRR datasets

Simulated datasets are of crucial importance for the benchmarking of AIR(R)-ML models, as they allow insight into how well the learned discriminatory patterns align with the ground truth immune signals. For this reason, immuneML provides instructions allowing the user to simulate synthetic immune signals into a given baseline AIRR datasets by implanting (gapped) k-mers into a fraction of the AIRRs.

Kanduri et al. [146] used immuneML to profile the baseline performance of classic ML models on such simulated AIRR data. This study provides a reference benchmark outlining the scenarios where such baseline ML models are already able to classify the data with high accuracy, and what the impact of different simulation parameters is on the ML method performance. Here, AIRR datasets are simulated by generating sets of synthetic AIR sequences using OLGA [17], followed by implanting k-mers using immuneML in order to simulate immune states. The simulated immune signals varied in k-mer size and complexity (e.g., gaps), number of different k-mers and abundance (witness rate). Other dataset-defining parameters that were varied include the AIRR size, number of AIRRs, balance between positive and negative AIRRs and negative class noise. It was found that the AIRR size was a more important factor for the predictive performance of methods than the number of AIRRs. Furthermore, ML methods performed better when their parameterisations were compatible with those of the ground truth signal. This exemplifies the importance of explicitly stating what assumptions are made about the immune signal, and tailoring ML methodology to capture such signals.

Despite the usefulness of the k-mer implanting simulation strategy to create transparent benchmarking datasets, the resulting immune signals are not realistic. New insights have uncovered limitations and potential artefacts associated with this simulation strategy, and new simulation strategies are being developed accordingly to address these issues.

By introducing a k-mer into an existing sequence, the natural dependencies within the sequence may be disrupted. This could result in a sequence that would be highly unlikely or even impossible to create through V(D)J recombination. ML models trained on such simulated data may then learn to discriminate between healthy and diseased AIRRs based on the absence of natural sequence patterns, rather than the presence of ground truth k-mers. To avoid disrupting natural sequence dependencies when simulating disease-associated AIRs, a new simulation platform named LIgO has been developed [169]. LIgO is at its core powered by immuneML. It contains a broad extension of the immuneML simulation component, with a comparable YAML interface. Rather than *implanting* immune signals into AIRs, LIgO uses the immune signal as a criterion to *filter* a pool of natural or generated AIRs, and keeps only those containing the immune signal. This process is called rejection sampling. While rejection sampling is more computationally intensive than implanting, the natural properties of AIRs are preserved. In LIgO, the immune signal can be defined as the presence of a k-mer, but far more complex definitions of immune signals are supported as well.

While LIgO concerns the simulation of AIRs with natural-like properties, there also exists a natural relationship between the generation probability of an AIR and how often this AIR is expected to be present in an AIRR dataset. Convergent selection may result in disease-associated AIRs that are more abundantly present in an AIRR dataset than expected based on their generation probabilities alone. Such AIRs appear as outliers when comparing their generation probability to their occurrence rate, which can be used as the basis for identifying AIRs active in an immune response [67]. However, naive simulation strategies where either k-mers or full AIR sequences are inserted into AIRRs leads to a greater number of outliers, as well as more extreme outliers, than observed in experimental AIRR datasets [170]. Thus, implanting k-mers or AIRs without regard for their generation probabilities creates artefacts which may give an unfair advantage to generation probability-aware AIRR classifiers. To overcome this, the AIRR simulation platform simAIRR has been developed to simulate AIRR datasets with a more realistic degree of AIR sharing across AIRRs [170]. simAIRR internally relies on CompAIRR to compute models of AIR sharing across AIRRs.

LIgO and simAIRR address complementary needs for the simulation of realistic AIRR datasets. While LIgO produces more realistic sets of AIRs sharing an immune signal, simAIRR ensures a more realistic distribution of disease-associated AIRs across AIRRs. Furthermore, the methodology and findings presented in Paper 3 (and potential future extensions thereof) can be used to inform the immune signal definition to be used by LIgO. As these projects have proceeded in parallel, future work may be needed to ensure a streamlined integration of these different tools into a single digital workspace.

Using CompAIRR to speed up TCRMatch

CompAIRR has been designed to have a flexible tool output, in order for it to be used in a wide variety of different applications. The integration of CompAIRR can improve the speed and memory usage of any use case relying on matching AIRs across large sets. Besides its integration in immuneML and simAIRR, CompAIRR has been used in several yet-to-be published projects.

In an ongoing collaboration with the Peters lab at the La Jolla Institute for Immunology (LJI) in San Diego, a pipeline has been created integrating CompAIRR and TCRMatch. TCRMatch is a tool for matching TCRs to curated TCR-epitope pairs that are available in the IEDB [121]. The purpose of integrating CompAIRR into this pipeline is to reduce the running time and memory usage of TCRMatch. Previously, the TCRMatch functionalities have been demonstrated using several tens of thousands of TCRs, yet this new pipeline broadens the scope of TCRMatch to being able to scan large-scale TCR repertoire datasets for similar sequences in the IEDB. In this pipeline, CompAIRR is first used to pre-filter the IEDB reference TCRs based on a Hamming distance cutoff (for example 1 or 2) to the query sequences. Then, TCRMatch is used to compute a more refined similarity score between the query sequences and the reduced version of the IEDB. A visual report has been implemented to guide the user in choosing an optimal Hamming distance threshold for their desired TCRMatch similarity score. The vast majority of sequence pairs in the IEDB with a TCRMatch score above 0.97 (the default threshold) are retained with a Hamming distance of 1, and all such sequence pairs fall within a Hamming distance of 2. Thus, the pipeline allows the users to recover (nearly) all high-similarity matches using only a fraction of the time and memory compared to using stand-alone TCRMatch. This pipeline opens up new opportunities for exploratory analyses of large-scale AIRR datasets guided by knowledge of curated AIRs.

Opportunities and challenges for exploring antigen binding motifs

With the methodology presented in Paper 3, we aimed to investigate to what extent antigen binding was determined by the presence of a short motif in the CDR3. To the best of our knowledge, this is the first study aiming to directly quantify the degree to which AIR-antigen binding is determined by the *presence* of a short motif in the CDR3, regardless of the remaining positions. While Glanville et al. explored the clustering of TCRs using local motifs in the CDR3

[36], the used dataset contained at most only 739 TCRs binding to the same target. These TCRs were obtained from blood samples and exhibited high similarity, obscuring the distinction between motifs that are shared due to convergent V(D)J recombination and genuine indicators of antigen binding.

We instead chose to work with mutagenesis-based datasets. This choice allowed for a more precise investigation of the decision boundary between binders and non-binders, as both groups originate from highly similar distributions. The resulting dataset contains a large variety of binders, but highly similar non-binders are represented as well. Such highly similar non-binders may remain unobserved in blood-derived AIR datasets, as the cells carrying these AIRs do not receive the same stimuli for proliferation. The sequences in the Mason and Mehta datasets are intentionally created to follow a highly skewed positional amino acid frequency distribution. Across the 10 heavy chain CDR3 positions that were mutated, some positions permit a lot of variability, whereas others are almost entirely fixed to one or few amino acids. This skewing was necessary to produce a sufficiently large number of different binder examples. However, it also introduces challenges for the interpretation of the results presented in Paper 3.

Even though the binder/non-binder status of a sequence is independent of the distribution the sequence was derived from, the sequence distribution does influence performance metrics such as precision and recall for a given motif. This means the high-precision, high-recall motifs learned on the Mason dataset may not be equally indicative of HER2 binding in an antibody dataset derived from a different distribution (e.g., experimental data generated through a different protocol, or combined datasets from a database). In an attempt to correct for the skewed positional amino acid distribution when computing motif performance metrics, an example weighting approach was initially explored. The goal was to estimate what the precision or recall of a given motif would be on a set of sequences drawn from a different (uniform, rather than skewed) positional amino acid frequency distribution. This was done by relative up-weighting of sequences containing ‘rare’ amino acids and down-weighting sequences with ‘common’ amino acid usage when computing performance metrics. While this approach was successful on simple simulated datasets, the up-weighting of ‘rare’ sequences in experimental datasets led to unstable results, possibly due to an amplification of noise. It should be noted that generalisability of classification criteria across data from different distributions is a universal problem in ML [171], and future investigations are needed to better understand how to combine or translate results between different AIR(R) datasets.

Despite the motifs themselves not necessarily generalising to different data distributions, the investigation and proposed methodology can still be used to investigate the properties of the immune signal (e.g., the shape and size). Naturally, the method can be extended to include a larger variety of motif definitions. Comparing the number of motifs found under different definitions, as well as the predictive performances of these different sets of motifs, could give an indication of which definition best captures the immune signal. Such investigations may help us design better AIR(R) classifiers in the future, both in performance and interpretability.

The two ways in which the motif definition can be modified are the *alphabet* and the *positions*. The alphabet could be updated to allow for synonymous amino acids according to biophysicochemical similarity, or it could include ‘negative’ amino acids, which match anything except a particular amino acid (group). One could also define a new motif type which is position-independent, and thus able to capture the same amino acid subsequence occurring across different positions in the CDR3. The benefit of position-independent motifs is that they can be searched for across CDR3 sequences of different lengths without prior alignment. Another way to introduce flexibility in the motif positions is to allow the length of the gap to vary. The motif alphabet could be extended with relatively few changes from the current implementation, for instance by introducing a set of special characters that represent groups of synonymous amino acids. Updating the motif positions is however not trivial. The total number of possible motifs is enormous, and the large majority of theoretically possible motifs do not occur in any sequence in the dataset. An efficient algorithm was implemented to find all candidate motifs that occur in a sufficient number of true positive (binder) sequences. The space of candidate motifs is explored by recursively extending each motif with only those position-specific amino acids that occur in a sufficient number of binders. By introducing any type of flexibility in the motif form, this trick for efficiently traversing the motif space can no longer be applied, as multiple amino acids in different positions need to be considered.

Conclusion

ML has transformed the analysis of large and highly complex datasets, including the analysis of AIR(R) data. AIRs are vital for the targeted recognition of antigens and maintaining immunological memory, and ML-based analysis of AIR(R) data could significantly accelerate the development of therapeutic drugs and diagnostic tests. To support such AIR(R)-ML analyses, several new computational tools have been introduced in this thesis.

The immuneML platform was created to simplify the development of new AIR(R)-ML methods, by providing a single, extensible platform supporting the comparative evaluation and introspection of such ML models. Researchers can integrate their new methodology into the platform and benefit from the already implemented workflows for ML model training and hyperparameter optimization, and easily benchmark their new ML method against a range of readily available methods. Furthermore, the YAML analysis specification allows for easy sharing of reproducible analysis workflows with collaborators, and can be executed both on the command line and Galaxy web interfaces.

While CompAIRR functions as a stand-alone tool for rapid computation of overlap between AIRRs, its flexibility allows it to be integrated in a wide variety of different workflows. CompAIRR has been used to speed up several AIR(R)-ML classifiers in immuneML, calibrating models of AIR sharing for simAIRR, and pre-filtering the IEDB when computing TCRMatch similarity scores. Any large-scale AIRR analysis which internally relies on computing (near) identical AIR matches can potentially benefit from integrating CompAIRR to improve speed or reduce memory usage.

The computational methodology presented in Paper 3 can be used to directly investigate the degree to which antigen binding can be predicted based on the presence of short motifs in the AIR CDR3. The method employs an efficient algorithm to search for motifs which have high precision in predicting antigen binding, yet also a sufficiently high recall to ensure generalisability to unseen data. We were able to find antibody CDR3 motifs that were highly predictive of HER2 binding, which generalised well to an independently generated antibody dataset. We found that many of the learned motifs were highly dispersed in nature, containing multiple gaps. This indicates that the field of AIR(R)-ML may benefit from developing architectures capable of capturing such gapped motifs. As more large experimental AIR-antigen binding datasets become available, it could serve as a foundation for the analysis of immune signals in the form of antigen binding motifs.

References

[1] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter, “T Cells and MHC Proteins,” in *Molecular Biology of the Cell. 4th edition*, Garland Science, 2002. Accessed: Jun. 12, 2023. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK26926/

[2] C. A. Janeway, P. Campregher, M. Walport, and M. J. Shlomchik, *Immunobiology*, 5th ed. Garland Science, 2001.

[3] J. K. Actor and K. C. Smith, “Inflammation,” in *Encyclopedia of Infection and Immunity*, N. Rezaei, Ed., Oxford: Elsevier, 2022, pp. 230–242. doi: 10.1016/B978-0-12-818731-9.00155-5.

[4] V. H. Odegard and D. G. Schatz, “Targeting of somatic hypermutation,” *Nat. Rev. Immunol.*, vol. 6, no. 8, pp. 573–583, Aug. 2006, doi: 10.1038/nri1896.

[5] Z. Li, C. J. Woo, M. D. Iglesias-Ussel, D. Ronai, and M. D. Scharff, “The generation of antibody diversity through somatic hypermutation and class switch recombination,” *Genes Dev.*, vol. 18, no. 1, pp. 1–11, Jan. 2004, doi: 10.1101/gad.1161904.

[6] N. A. Doria-Rose and M. G. Joyce, “Strategies to guide the antibody affinity maturation process,” *Curr Opin Virol*, vol. 11, pp. 137–147, Apr. 2015, doi: 10.1016/j.coviro.2015.04.002.

[7] S. Crotty and R. Ahmed, “Immunological memory in humans,” *Seminars in Immunology*, vol. 16, no. 3, pp. 197–203, Jun. 2004, doi: 10.1016/j.smim.2004.02.008.

[8] O. Backhaus, “Generation of Antibody Diversity,” *Antibody Engineering*, Feb. 2018, doi: 10.5772/intechopen.72818.

[9] J. J. A. Calis and B. R. Rosenberg, “Characterizing immune repertoires by high throughput sequencing: strategies and applications,” *Trends in Immunology*, vol. 35, no. 12, pp. 581–590, Dec. 2014, doi: 10.1016/j.it.2014.09.004.

[10] L. Scheffer, “Simulation and analysis of immune receptor repertoire frequency distributions,” Master thesis, 2019. Accessed: Jun. 30, 2023. [Online]. Available: https://www.duo.uio.no/handle/10852/69136

[11] F. W. Alt, E. M. Oltz, F. Young, J. Gorman, G. Taccioli, and J. Chen, “VDJ recombination,” *Immunology Today*, vol. 13, no. 8, pp. 306–314, Jan. 1992, doi: 10.1016/0167-5699(92)90043-7.

[12] T. M. Yankee, “B-Cell Antigen Receptor,” in *Encyclopedia of Biological Chemistry (Second Edition)*, W. J. Lennarz and M. D. Lane, Eds., Waltham: Academic Press, 2013, pp. 161–164. doi: 10.1016/B978-0-12-378630-2.00325-X.

[13] V. I. Zarnitsyna, B. D. Evavold, L. N. Schoettle, J. N. Blattman, and R. Antia, “Estimating the Diversity, Completeness, and Cross-Reactivity of the T Cell Repertoire,” *Frontiers in Immunology*, vol. 4, 2013, doi: 10.3389/fimmu.2013.00485.

[14] B. Briney, A. Inderbitzin, C. Joyce, and D. R. Burton, “Commonality despite exceptional diversity in the baseline human antibody repertoire,” *Nature*, vol. 566, no. 7744, pp. 393–397, Feb. 2019, doi: 10.1038/s41586-019-0879-y.

[15] H. L. Spiegelberg, “Biological Role of Different Antibody Classes,” *International Archives of Allergy and Applied Immunology*, vol. 90, no. Suppl. 1, pp. 22–27, Aug. 2009, doi: 10.1159/000235071.

[16] Q. Marcou, T. Mora, and A. M. Walczak, “High-throughput immune repertoire analysis with IGoR,” *Nature Communications*, vol. 9, no. 1, Art. no. 1, Feb. 2018, doi: 10.1038/s41467-018-02832-w.

[17] Z. Sethna, Y. Elhanati, C. G. Callan, A. M. Walczak, and T. Mora, “OLGA: fast computation of generation probabilities of B- and T-cell receptor amino acid sequences and motifs,” *Bioinformatics*, vol. 35, no. 17, pp. 2974–2981, Sep. 2019, doi: 10.1093/bioinformatics/btz035.

[18] R. Levi and Y. Louzoun, “Two Step Selection for Bias in β Chain V-J Pairing,” *Frontiers in Immunology*, vol. 13, 2022, Accessed: Jun. 30, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fimmu.2022.906217

[19] M. J. Kidd, K. J. L. Jackson, S. D. Boyd, and A. M. Collins, “DJ Pairing during VDJ Recombination Shows Positional Biases That Vary among Individuals with Differing IGHD Locus Immunogenotypes,” *J Immunol*, vol. 196, no. 3, pp. 1158–1164, Feb. 2016, doi: 10.4049/jimmunol.1501401.

[20] K. Peng *et al.*, “Diversity in immunogenomics: the value and the challenge,” *Nat Methods*, vol. 18, no. 6, Art. no. 6, Jun. 2021, doi: 10.1038/s41592-021-01169-5.

[21] A. Slabodkin *et al.*, “Individualized VDJ recombination predisposes the available Ig sequence space,” *Genome Res*, vol. 31, no. 12, pp. 2209–2224, Dec. 2021, doi: 10.1101/gr.275373.121.

[22] M. L. Russell *et al.*, “Combining genotypes and T cell receptor distributions to infer genetic loci determining V(D)J recombination probabilities,” *eLife*, vol. 11, p. e73475, Mar. 2022, doi: 10.7554/eLife.73475.

[23] J. Charles A Janeway, P. Travers, M. Walport, and M. J. Shlomchik, “The rearrangement of antigen-receptor gene segments controls lymphocyte development,” in *Immunobiology: The Immune System in Health and Disease. 5th edition*, Garland Science, 2001. Accessed: Jun. 12, 2023. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK27113/

[24] M. M. Corcoran *et al.*, “Production of individualized V gene databases reveals high levels of immunoglobulin genetic diversity,” *Nat Commun*, vol. 7, no. 1, pp. 1–14, Dec. 2016, doi: 10.1038/ncomms13642.

[25] N. J. Schuldt and B. A. Binstadt, “Dual TCR T Cells: Identity Crisis or Multitaskers?,” *The Journal of Immunology*, vol. 202, no. 3, pp. 637–644, Feb. 2019, doi: 10.4049/jimmunol.1800904.

[26] M. Wieczorek *et al.*, “Major Histocompatibility Complex (MHC) Class I and MHC Class II Proteins: Conformational Plasticity in Antigen Presentation,” *Frontiers in Immunology*, vol. 8, 2017, Accessed: Jun. 15, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fimmu.2017.00292

[27] I. L. Dzhagalov, K. G. Chen, P. Herzmark, and E. A. Robey, “Elimination of Self-Reactive T Cells in the Thymus: A Timeline for Negative Selection,” *PLoS Biol*, vol. 11, no. 5, p. e1001566, May 2013, doi: 10.1371/journal.pbio.1001566.

[28] K. C. Garcia and E. J. Adams, “How the T Cell Receptor Sees Antigen—A Structural View,” *Cell*, vol. 122, no. 3, pp. 333–336, Aug. 2005, doi: 10.1016/j.cell.2005.07.015.

[29] E. M. Shevach, “MHC Restriction,” in *Encyclopedia of Immunology (Second Edition)*, P. J. Delves, Ed., Oxford: Elsevier, 1998, pp. 1709–1712. doi: 10.1006/rwei.1999.0434.

[30] R.-D. Jores *et al.*, “HLA-DQB1*0201 homozygosis predisposes to severe intestinal damage in celiac disease,” *Scandinavian Journal of Gastroenterology*, vol. 42, no. 1, pp. 48–53, Jan. 2007, doi: 10.1080/00365520600789859.

[31] C. Nguyen, M. D. Varney, L. C. Harrison, and G. Morahan, “Definition of High-Risk Type 1 Diabetes HLA-DR and HLA-DQ Types Using Only Three Single Nucleotide Polymorphisms,” *Diabetes*, vol. 62, no. 6, pp. 2135–2140, May 2013, doi: 10.2337/db12-1398.

[32] C. Szeto, C. A. Lobos, A. T. Nguyen, and S. Gras, “TCR Recognition of Peptide-MHC-I: Rule Makers and Breakers,” *Int J Mol Sci*, vol. 22, no. 1, p. 68, Dec. 2020, doi: 10.3390/ijms22010068.

[33] K. Tsumoto and J. M. Caaveiro, “Antigen–Antibody Binding,” in *Encyclopedia of Life Sciences*, John Wiley & Sons, Ltd, 2016, pp. 1–8. doi: 10.1002/9780470015902.a0001117.pub3.

[34] M.-P. Lefranc, “Epitope,” in *Encyclopedia of Systems Biology*, W. Dubitzky, O.

Wolkenhauer, K.-H. Cho, and H. Yokota, Eds., New York, NY: Springer, 2013, pp. 672–673. doi: 10.1007/978-1-4419-9863-7_663.

[35] M.-P. Lefranc, “Paratope,” in *Encyclopedia of Systems Biology*, W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota, Eds., New York, NY: Springer, 2013, pp. 1632–1633. doi: 10.1007/978-1-4419-9863-7_673.

[36] J. Glanville *et al.*, “Identifying specificity groups in the T cell receptor repertoire,” *Nature*, vol. 547, no. 7661, pp. 94–98, Jul. 2017, doi: 10.1038/nature22976.

[37] J. Ostmeyer, S. Christley, I. T. Toby, and L. G. Cowell, “Biophysicochemical motifs in T cell receptor sequences distinguish repertoires from tumor-infiltrating lymphocytes and adjacent healthy tissue,” *Cancer Res*, p. canres.2292.2018, Jan. 2019, doi: 10.1158/0008-5472.CAN-18-2292.

[38] R. Akbar *et al.*, “A compact vocabulary of paratope-epitope interactions enables predictability of antibody-antigen binding,” *Cell Reports*, vol. 34, no. 11, p. 108856, Mar. 2021, doi: 10.1016/j.celrep.2021.108856.

[39] J. L. Xu and M. M. Davis, “Diversity in the CDR3 Region of VH Is Sufficient for Most Antibody Specificities,” *Immunity*, vol. 13, no. 1, pp. 37–45, Jul. 2000, doi: 10.1016/S1074-7613(00)00006-6.

[40] M. M. Davis and P. J. Bjorkman, “T-cell antigen receptor genes and T-cell recognition,” *Nature*, vol. 334, no. 6181, p. 395, Aug. 1988, doi: 10.1038/334395a0.

[41] D. K. Cole *et al.*, “T-cell receptor (TCR)-peptide specificity overrides affinity-enhancing TCR-major histocompatibility complex interactions,” *J Biol Chem*, vol. 289, no. 2, pp. 628–638, Jan. 2014, doi: 10.1074/jbc.M113.522110.

[42] S. A. Frank, “Specificity and Cross-Reactivity,” in *Immunology and Evolution of Infectious Disease*, Princeton University Press, 2002. Accessed: Jun. 30, 2023. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK2396/

[43] G. Petrova, A. Ferrante, and J. Gorski, “Cross-Reactivity of T Cells and Its Role in the Immune System,” *Crit Rev Immunol*, vol. 32, no. 4, pp. 349–372, 2012, Accessed: Nov. 20, 2019. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3595599/

[44] A. K. Sewell, “Why must T cells be cross-reactive?,” *Nat Rev Immunol*, vol. 12, no. 9, Art. no. 9, Sep. 2012, doi: 10.1038/nri3279.

[45] A. Cossarizza *et al.*, “Guidelines for the use of flow cytometry and cell sorting in immunological studies (third edition),” *European Journal of Immunology*, vol. 51, no. 12, pp. 2708–3145, 2021, doi: 10.1002/eji.202170126.

[46] C. Dens, K. Laukens, W. Bittremieux, and P. Meysman, “The pitfalls of negative data bias for the T-cell epitope specificity challenge.” *Nat Mach Intell*, vol. 5, p. 1060–1062, Oct. 05, 2023. doi: 10.1038/s42256-023-00727-0.

[47] V. Greiff, G. Yaari, and L. G. Cowell, “Mining adaptive immune receptor repertoires for biological and clinical information using machine learning,” *Current Opinion in Systems Biology*, vol. 24, pp. 109–119, Dec. 2020, doi: 10.1016/j.coisb.2020.10.010.

[48] H. Liu *et al.*, “The methods and advances of adaptive immune receptors repertoire sequencing,” *Theranostics*, vol. 11, no. 18, pp. 8945–8963, Aug. 2021, doi: 10.7150/thno.61390.

[49] A. J. Brown *et al.*, “Augmenting adaptive immunity: progress and challenges in the quantitative engineering and analysis of adaptive immune receptor repertoires,” *Mol. Syst. Des. Eng.*, vol. 4, no. 4, pp. 701–736, Aug. 2019, doi: 10.1039/C9ME00071B.

[50] J. K. H. Liu, “The history of monoclonal antibody development – Progress, remaining challenges and future innovations,” *Annals of Medicine and Surgery*, vol. 3, no. 4, p. 113, Dec. 2014, doi: 10.1016/j.amsu.2014.09.001.

[51] R. Akbar *et al.*, “Progress and challenges for the machine learning-based design of fit-for-purpose monoclonal antibodies,” *mAbs*, vol. 14, no. 1, p. 2008790, Dec. 2022, doi: 10.1080/19420862.2021.2008790.

[52] A. R. Rees, “Understanding the human antibody repertoire,” *mAbs*, vol. 12, no. 1, p. 1729683, Jan. 2020, doi: 10.1080/19420862.2020.1729683.
[53] Q. Qi *et al.*, “Diversity and clonal selection in the human T-cell repertoire,” *PNAS*, vol. 111, no. 36, pp. 13139–13144, Sep. 2014, doi: 10.1073/pnas.1409155111.
[54] G. Lythe, R. E. Callard, R. L. Hoare, and C. Molina-París, “How many TCR clonotypes does a body maintain?,” *J Theor Biol*, vol. 389, pp. 214–224, Jan. 2016, doi: 10.1016/j.jtbi.2015.10.016.
[55] E. Miho, A. Yermanos, C. R. Weber, C. T. Berger, S. T. Reddy, and V. Greiff, “Computational Strategies for Dissecting the High-Dimensional Complexity of Adaptive Immune Repertoires,” *Front. Immunol.*, vol. 9, 2018, doi: 10.3389/fimmu.2018.00224.
[56] T. Mora, A. M. Walczak, W. Bialek, and C. G. Callan, “Maximum entropy models for antibody diversity,” *PNAS*, vol. 107, no. 12, pp. 5405–5410, Mar. 2010, doi: 10.1073/pnas.1001705107.
[57] T. Oakes *et al.*, “Quantitative Characterization of the T Cell Receptor Repertoire of Naïve and Memory Subsets Using an Integrated Experimental and Computational Pipeline Which Is Robust, Economical, and Versatile,” *Front Immunol*, vol. 8, Oct. 2017, doi: 10.3389/fimmu.2017.01267.
[58] M. Shugay, D. Bolotin, E. Putintseva, M. Pogorelyy, I. Mamedov, and D. Chudakov, “Huge Overlap of Individual TCR Beta Repertoires,” *Frontiers in Immunology*, vol. 4, 2013, Accessed: Jun. 19, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fimmu.2013.00466
[59] A. M. Rosenfeld *et al.*, “Computational Evaluation of B-Cell Clone Sizes in Bulk Populations,” *Front Immunol*, vol. 9, p. 1472, Jun. 2018, doi: 10.3389/fimmu.2018.01472.
[60] G. A. Rempała and M. Seweryn, “Methods for diversity and overlap analysis in T-cell receptor populations,” *J Math Biol*, vol. 67, no. 0, Dec. 2013, doi: 10.1007/s00285-012-0589-7.
[61] G. A. Rempala, M. Seweryn, and L. Ignatowicz, “Model for comparative analysis of antigen receptor repertoires,” *Journal of Theoretical Biology*, vol. 269, no. 1, pp. 1–15, Jan. 2011, doi: 10.1016/j.jtbi.2010.10.001.
[62] V. Venturi, K. Kedzierska, M. M. Tanaka, S. J. Turner, P. C. Doherty, and M. P. Davenport, “Method for assessing the similarity between subsets of the T cell receptor repertoire,” *J Immunol Methods*, vol. 329, no. 1–2, pp. 67–80, Jan. 2008, doi: 10.1016/j.jim.2007.09.016.
[63] V. Venturi, K. Kedzierska, S. J. Turner, P. C. Doherty, and M. P. Davenport, “Methods for comparing the diversity of samples of the T cell receptor repertoire,” *Journal of Immunological Methods*, vol. 321, no. 1, pp. 182–195, Apr. 2007, doi: 10.1016/j.jim.2007.01.019.
[64] A. Pelissier, S. Luo, M. Stratigopoulou, J. E. Guikema, and M. R. Martinez, “Quantifying B-cell Clonal Diversity In Repertoire Data.” bioRxiv, p. 2022.12.12.520133, Dec. 18, 2022. doi: 10.1101/2022.12.12.520133.
[65] J. D. Galson *et al.*, “In-Depth Assessment of Within-Individual and Inter-Individual Variation in the B Cell Receptor Repertoire,” *Frontiers in Immunology*, vol. 6, 2015, Accessed: May 21, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fimmu.2015.00531
[66] H. S. Robins *et al.*, “Overlap and Effective Size of the Human CD8+ T Cell Receptor Repertoire,” *Science Translational Medicine*, vol. 2, no. 47, pp. 47ra64-47ra64, Sep. 2010, doi: 10.1126/scitranslmed.3001442.
[67] M. V. Pogorelyy *et al.*, “Method for identification of condition-associated public antigen receptor sequences,” *eLife*, vol. 7, p. e33050, Mar. 2018, doi: 10.7554/eLife.33050.
[68] Y. Elhanati, Z. Sethna, C. G. Callan, T. Mora, and A. M. Walczak, “Predicting the spectrum of TCR repertoire sharing with a data‐driven model of recombination,” *Immunol*

Rev, vol. 284, no. 1, pp. 167–179, Jul. 2018, doi: 10.1111/imr.12665.

[69] H. Li *et al.*, “Recombinatorial biases and convergent recombination determine interindividual TCRβ sharing in murine thymocytes,” *J Immunol*, vol. 189, no. 5, pp. 2404–2413, Sep. 2012, doi: 10.4049/jimmunol.1102087.

[70] V. Venturi *et al.*, “Sharing of T cell receptors in antigen-specific responses is driven by convergent recombination,” *Proceedings of the National Academy of Sciences*, vol. 103, no. 49, pp. 18691–18696, Dec. 2006, doi: 10.1073/pnas.0608907103.

[71] M. F. Quigley *et al.*, “Convergent recombination shapes the clonotypic landscape of the naive T-cell repertoire,” *Proc Natl Acad Sci U S A*, vol. 107, no. 45, pp. 19414–19419, Nov. 2010, doi: 10.1073/pnas.1010586107.

[72] R. O. Emerson *et al.*, “Immunosequencing identifies signatures of cytomegalovirus exposure history and HLA-mediated effects on the T cell repertoire,” *Nature Genetics*, vol. 49, no. 5, pp. 659–665, May 2017, doi: 10.1038/ng.3822.

[73] M. V. Pogorelyy *et al.*, “Detecting T cell receptors involved in immune responses from single repertoire snapshots,” *PLOS Biology*, vol. 17, no. 6, p. e3000314, Jun. 2019, doi: 10.1371/journal.pbio.3000314.

[74] A. Six *et al.*, “The Past, Present, and Future of Immune Repertoire Biology – The Rise of Next-Generation Repertoire Analysis,” *Frontiers in Immunology*, vol. 4, 2013, Accessed: May 19, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fimmu.2013.00413

[75] S. Valkiers *et al.*, “Recent advances in T-cell receptor repertoire analysis: Bridging the gap with multimodal single-cell RNA sequencing,” *ImmunoInformatics*, vol. 5, p. 100009, Mar. 2022, doi: 10.1016/j.immuno.2022.100009.

[76] D. Shlesinger *et al.*, “Single-cell immune repertoire sequencing of B and T cells in murine models of infection and autoimmunity,” *Genes Immun*, vol. 23, no. 6, Art. no. 6, Sep. 2022, doi: 10.1038/s41435-022-00180-w.

[77] J. Trück *et al.*, “Biological controls for standardization and interpretation of adaptive immune receptor repertoire profiling,” *eLife*, vol. 10, p. e66274, May 2021, doi: 10.7554/eLife.66274.

[78] N. C. Curtis and J. Lee, “Beyond bulk single-chain sequencing: Getting at the whole receptor,” *Curr Opin Syst Biol*, vol. 24, pp. 93–99, Dec. 2020, doi: 10.1016/j.coisb.2020.10.008.

[79] D. A. Bolotin *et al.*, “MiXCR: software for comprehensive adaptive immunity profiling,” *Nature Methods*, vol. 12, no. 5, pp. 380–381, May 2015, doi: 10.1038/nmeth.3364.

[80] J. A. Vander Heiden *et al.*, “AIRR Community Standardized Representations for Annotated Immune Repertoires,” *Front. Immunol.*, vol. 9, 2018, doi: 10.3389/fimmu.2018.02206.

[81] M. Shugay *et al.*, “VDJdb: a curated database of T-cell receptor sequences with known antigen specificity,” *Nucleic Acids Res*, vol. 46, no. Database issue, pp. D419–D427, Jan. 2018, doi: 10.1093/nar/gkx760.

[82] D. V. Bagaev *et al.*, “VDJdb in 2019: database extension, new analysis infrastructure and a T-cell receptor motif compendium,” *Nucleic Acids Research*, vol. 48, no. D1, pp. D1057–D1062, Jan. 2020, doi: 10.1093/nar/gkz874.

[83] N. Tickotsky, T. Sagiv, J. Prilusky, E. Shifrut, and N. Friedman, “McPAS-TCR: a manually curated catalogue of pathology-associated T cell receptor sequences,” *Bioinformatics*, vol. 33, no. 18, pp. 2924–2929, Sep. 2017, doi: 10.1093/bioinformatics/btx286.

[84] B. Peters *et al.*, “The immune epitope database and analysis resource: from vision to blueprint,” *PLoS Biol*, vol. 3, no. 3, p. e91, Mar. 2005, doi: 10.1371/journal.pbio.0030091.

[85] R. Vita *et al.*, “The immune epitope database 2.0,” *Nucleic Acids Res*, vol. 38, no. Database issue, pp. D854-862, Jan. 2010, doi: 10.1093/nar/gkp1004.

[86] R. Vita *et al.*, “The immune epitope database (IEDB) 3.0,” *Nucleic Acids Res*, vol. 43, no.

Database issue, pp. D405-412, Jan. 2015, doi: 10.1093/nar/gku938.
[87] R. Vita *et al.*, “The Immune Epitope Database (IEDB): 2018 update,” *Nucleic Acids Res*, vol. 47, no. D1, pp. D339–D343, Jan. 2019, doi: 10.1093/nar/gky1006.
[88] B. D. Corrie *et al.*, “iReceptor: a platform for querying and analyzing antibody/B-cell and T-cell receptor repertoire data across federated repositories,” *Immunol Rev*, vol. 284, no. 1, pp. 24–41, Jul. 2018, doi: 10.1111/imr.12666.
[89] A. L. Samuel, “Some Studies in Machine Learning Using the Game of Checkers,” *IBM Journal of Research and Development*, vol. 3, no. 3, pp. 210–229, Jul. 1959, doi: 10.1147/rd.33.0210.
[90] H. Wang and B. Raj, “On the Origin of Deep Learning.” arXiv, Mar. 02, 2017. Accessed: Jul. 02, 2023. [Online]. Available: http://arxiv.org/abs/1702.07800
[91] S. Min, B. Lee, and S. Yoon, “Deep learning in bioinformatics,” *Briefings in Bioinformatics*, vol. 18, no. 5, pp. 851–869, Sep. 2017, doi: 10.1093/bib/bbw068.
[92] B. Tang, Z. Pan, K. Yin, and A. Khateeb, “Recent Advances of Deep Learning in Bioinformatics and Computational Biology,” *Frontiers in Genetics*, vol. 10, 2019, Accessed: Jul. 02, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fgene.2019.00214
[93] M. Wainberg, D. Merico, A. Delong, and B. J. Frey, “Deep learning in biomedicine,” *Nat Biotechnol*, vol. 36, no. 9, Art. no. 9, Oct. 2018, doi: 10.1038/nbt.4233.
[94] X. Shen, C. Jiang, Y. Wen, C. Li, and Q. Lu, “A Brief Review on Deep Learning Applications in Genomic Studies,” *Frontiers in Systems Biology*, vol. 2, 2022, Accessed: Jul. 02, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fsysb.2022.877717
[95] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org
[96] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer, 2009. [Online]. Available: hastie.su.domains/ElemStatLearn
[97] M. G. H. Omran, A. P. Engelbrecht, and A. Salman, “An overview of clustering methods,” *Intelligent Data Analysis*, vol. 11, no. 6, pp. 583–605, Jan. 2007, doi: 10.3233/IDA-2007-11602.
[98] T. Ullmann, C. Hennig, and A.-L. Boulesteix, “Validation of cluster analysis results on validation data: A systematic framework,” *WIREs Data Mining and Knowledge Discovery*, vol. 12, no. 3, p. e1444, 2022, doi: 10.1002/widm.1444.
[99] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez, “Solving the multiple instance problem with axis-parallel rectangles,” *Artificial Intelligence*, vol. 89, no. 1, pp. 31–71, Jan. 1997, doi: 10.1016/S0004-3702(96)00034-3.
[100] M.-A. Carbonneau, V. Cheplygina, E. Granger, and G. Gagnon, “Multiple instance learning: A survey of problem characteristics and applications,” *Pattern Recognition*, vol. 77, pp. 329–353, May 2018, doi: 10.1016/j.patcog.2017.10.009.
[101] M. Widrich *et al.*, “Modern Hopfield Networks and Attention for Immune Repertoire Classification,” in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2020, pp. 18832–18845. Accessed: Apr. 22, 2023. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/da4902cb0bc38210839714ebdcf0efc3-Abstract.html
[102] J. Ostmeyer *et al.*, “Biophysicochemical motifs in T cell receptor sequences as a potential biomarker for high-grade serous ovarian carcinoma,” *PLOS ONE*, vol. 15, no. 3, p. e0229569, Mar. 2020, doi: 10.1371/journal.pone.0229569.
[103] J.-W. Sidhom, H. B. Larman, D. M. Pardoll, and A. S. Baras, “DeepTCR is a deep learning framework for revealing sequence concepts within T-cell repertoires,” *Nat Commun*, vol. 12, no. 1, Art. no. 1, Mar. 2021, doi: 10.1038/s41467-021-21879-w.
[104] J. Foulds and E. Frank, “A review of multi-instance learning assumptions,” *The*

Knowledge Engineering Review, vol. 25, no. 1, pp. 1–25, Mar. 2010, doi: 10.1017/S026988890999035X.
[105] K. C. Kiwiel, “Convergence and efficiency of subgradient methods for quasiconvex minimization,” *Math. Program.*, vol. 90, no. 1, pp. 1–25, Mar. 2001, doi: 10.1007/PL00011414.
[106] U. Ferraro Petrillo, M. Sorella, G. Cattaneo, R. Giancarlo, and S. E. Rombo, “Analyzing big datasets of genomic sequences: fast and scalable collection of k-mer statistics,” *BMC Bioinformatics*, vol. 20, no. 4, p. 138, Apr. 2019, doi: 10.1186/s12859-019-2694-8.
[107] S. Varma and R. Simon, “Bias in error estimation when using cross-validation for model selection,” *BMC Bioinformatics*, vol. 7, no. 1, p. 91, Feb. 2006, doi: 10.1186/1471-2105-7-91.
[108] P. Meysman, N. De Neuter, S. Gielis, D. Bui Thi, B. Ogunjimi, and K. Laukens, “On the viability of unsupervised T-cell receptor sequence clustering for epitope preference,” *Bioinformatics*, vol. 35, no. 9, pp. 1461–1468, May 2019, doi: 10.1093/bioinformatics/bty821.
[109] R. A. Arnaout *et al.*, “The Future of Blood Testing Is the Immunome,” *Front. Immunol.*, vol. 12, 2021, doi: 10.3389/fimmu.2021.626793.
[110] P. Meysman *et al.*, “Benchmarking solutions to the T-cell receptor epitope prediction problem: IMMREP22 workshop report,” *ImmunoInformatics*, vol. 9, p. 100024, Mar. 2023, doi: 10.1016/j.immuno.2023.100024.
[111] N. De Neuter *et al.*, “On the feasibility of mining CD8+ T cell receptor patterns underlying immunogenic peptide recognition,” *Immunogenetics*, vol. 70, no. 3, pp. 159–168, Mar. 2018, doi: 10.1007/s00251-017-1023-5.
[112] D. Hudson, R. A. Fernandes, M. Basham, G. Ogg, and H. Koohy, “Can we predict T cell specificity with digital biology and machine learning?,” *Nat Rev Immunol*, pp. 1–11, Feb. 2023, doi: 10.1038/s41577-023-00835-3.
[113] D. M. Mason *et al.*, “Optimization of therapeutic antibodies by predicting antigen specificity from antibody sequence via deep learning,” *Nat Biomed Eng*, vol. 5, no. 6, Art. no. 6, Jun. 2021, doi: 10.1038/s41551-021-00699-9.
[114] S. Zhang, T. Yang, X. Liu, J. Yang, and X. Zheng, “Antibody repertoire sequencing analysis,” *Acta Biochim Biophys Sin (Shanghai)*, vol. 54, no. 6, pp. 864–873, Jun. 2022, doi: 10.3724/abbs.2022062.
[115] H. Huang, C. Wang, F. Rubelt, T. J. Scriba, and M. M. Davis, “Analyzing the Mycobacterium tuberculosis immune response by T-cell receptor clustering with GLIPH2 and genome-wide antigen screening,” *Nature Biotechnology*, vol. 38, no. 10, Art. no. 10, Oct. 2020, doi: 10.1038/s41587-020-0505-4.
[116] P. Dash *et al.*, “Quantifiable predictive features define epitope-specific T cell receptor repertoires,” *Nature*, vol. 547, no. 7661, pp. 89–93, Jul. 2017, doi: 10.1038/nature22383.
[117] K. Mayer-Blackwell *et al.*, “TCR meta-clonotypes for biomarker discovery with tcrdist3 enabled identification of public, HLA-restricted clusters of SARS-CoV-2 TCRs,” *eLife*, vol. 10, p. e68605, Nov. 2021, doi: 10.7554/eLife.68605.
[118] H. Zhang *et al.*, “Investigation of Antigen-Specific T-Cell Receptor Clusters in Human Cancers,” *Clin Cancer Res*, vol. 26, no. 6, pp. 1359–1371, 15 2020, doi: 10.1158/1078-0432.CCR-19-3249.
[119] S. Valkiers, M. Van Houcke, K. Laukens, and P. Meysman, “ClusTCR: a python interface for rapid clustering of large sets of CDR3 sequences with unknown antigen specificity,” *Bioinformatics*, no. btab446, Jun. 2021, doi: 10.1093/bioinformatics/btab446.
[120] H. Zhang, X. Zhan, and B. Li, “GIANA allows computationally-efficient TCR clustering and multi-disease repertoire classification by isometric transformation,” *Nat Commun*, vol. 12, no. 1, p. 4699, Aug. 2021, doi: 10.1038/s41467-021-25006-7.
[121] W. D. Chronister *et al.*, “TCRMatch: Predicting T-Cell Receptor Specificity Based on

Sequence Similarity to Previously Characterized Receptors," *Frontiers in Immunology*, vol. 12, 2021, Accessed: May 21, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fimmu.2021.640725

[122] V. I. Jurtz *et al.*, "NetTCR: sequence-based prediction of TCR binding to peptide-MHC complexes using convolutional neural networks." bioRxiv, p. 433706, Oct. 03, 2018. doi: 10.1101/433706.

[123] A. Montemurro *et al.*, "NetTCR-2.0 enables accurate prediction of TCR-peptide binding by using paired TCRα and β sequence data," *Commun Biol*, vol. 4, no. 1, Art. no. 1, Sep. 2021, doi: 10.1038/s42003-021-02610-3.

[124] I. Springer, H. Besser, N. Tickotsky-Moskovitz, S. Dvorkin, and Y. Louzoun, "Prediction of Specific TCR-Peptide Binding From Large Dictionaries of TCR-Peptide Pairs," *Frontiers in Immunology*, vol. 11, 2020, Accessed: Jun. 28, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fimmu.2020.01803

[125] 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," *Frontiers in Immunology*, vol. 12, 2021, Accessed: Jun. 28, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fimmu.2021.664514

[126] P. Moris *et al.*, "Current challenges for unseen-epitope TCR interaction prediction and a new perspective derived from image classification," *Briefings in Bioinformatics*, vol. 22, no. 4, p. bbaa318, Jul. 2021, doi: 10.1093/bib/bbaa318.

[127] A. Weber, J. Born, and M. Rodriguez Martínez, "TITAN: T-cell receptor specificity prediction with bimodal attention networks," *Bioinformatics*, vol. 37, no. Supplement_1, pp. i237–i244, Jul. 2021, doi: 10.1093/bioinformatics/btab294.

[128] Y. Zhao *et al.*, "DeepAIR: a deep-learning framework for effective integration of sequence and 3D structure to enable adaptive immune receptor analysis." *Sci Adv*, p. vol. 9, p. eabo5128, 2023. doi: 10.1126/sciadv.abo5128.

[129] N. L. Miller, T. Clark, R. Raman, and R. Sasisekharan, "Learned features of antibody-antigen binding affinity," *Frontiers in Molecular Biosciences*, vol. 10, 2023, Accessed: Apr. 26, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fmolb.2023.1112738

[130] L. Slabinski *et al.*, "The challenge of protein structure determination—lessons from structural genomics," *Protein Sci*, vol. 16, no. 11, pp. 2472–2482, Nov. 2007, doi: 10.1110/ps.073037907.

[131] P. A. Robert *et al.*, "Unconstrained generation of synthetic antibody–antigen structures to guide machine learning methodology for antibody specificity prediction," *Nat Comput Sci*, vol. 2, no. 12, Art. no. 12, Dec. 2022, doi: 10.1038/s43588-022-00372-4.

[132] M. E. Zaslavsky *et al.*, "Disease diagnostics using machine learning of immune receptors," *bioRxiv*, p. 2022.04.26.489314, Feb. 2023, doi: 10.1101/2022.04.26.489314.

[133] Y. Katayama, R. Yokota, T. Akiyama, and T. J. Kobayashi, "Machine Learning Approaches to TCR Repertoire Analysis," *Frontiers in Immunology*, vol. 13, 2022, Accessed: Mar. 22, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fimmu.2022.858057

[134] M. Pavlović *et al.*, "Improving generalization of machine learning-identified biomarkers with causal modeling: an investigation into immune receptor diagnostics." arXiv, Apr. 03, 2023. doi: 10.48550/arXiv.2204.09291.

[135] V. Greiff, P. Bhat, S. C. Cook, U. Menzel, W. Kang, and S. T. Reddy, "A bioinformatic framework for immune repertoire diversity profiling enables detection of immunological status," *Genome Med*, vol. 7, no. 1, May 2015, doi: 10.1186/s13073-015-0169-8.

[136] M. O. Hill, "Diversity and evenness: a unifying notation and its consequences," *Ecology*, vol. 54, no. 2, pp. 427–432, Mar. 1973, doi: 10.2307/1934352.

[137] Y. Katayama and T. J. Kobayashi, "Comparative Study of Repertoire Classification Methods Reveals Data Efficiency of k-mer Feature Extraction," *Frontiers in Immunology*,

vol. 13, 2022, Accessed: Jul. 04, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fimmu.2022.797640

[138] M. Cinelli *et al.*, “Feature selection using a one dimensional naïve Bayes’ classifier increases the accuracy of support vector machine classification of CDR3 repertoires,” *Bioinformatics*, vol. 33, no. 7, pp. 951–955, Apr. 2017, doi: 10.1093/bioinformatics/btw771.

[139] Y. Sun *et al.*, “Specificity, Privacy, and Degeneracy in the CD4 T Cell Receptor Repertoire Following Immunization,” *Frontiers in Immunology*, vol. 8, 2017, Accessed: Jul. 04, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fimmu.2017.00430

[140] D. Xiong, Z. Zhang, T. Wang, and X. Wang, “A comparative study of multiple instance learning methods for cancer detection using T-cell receptor sequences,” *Computational and Structural Biotechnology Journal*, vol. 19, pp. 3255–3268, Jan. 2021, doi: 10.1016/j.csbj.2021.05.038.

[141] Y. Kim, T. Wang, D. Xiong, X. Wang, and S. Park, “Multiple instance neural networks based on sparse attention for cancer detection using T-cell receptor sequences,” *BMC Bioinformatics*, vol. 23, no. 1, p. 469, Nov. 2022, doi: 10.1186/s12859-022-05012-2.

[142] N. De Neuter *et al.*, “Memory CD4+ T cell receptor repertoire data mining as a tool for identifying cytomegalovirus serostatus,” *Genes Immun*, vol. 20, no. 3, Art. no. 3, Mar. 2019, doi: 10.1038/s41435-018-0035-y.

[143] G. Picarda and C. A. Benedict, “Cytomegalovirus: shape-shifting the immune system,” *J Immunol*, vol. 200, no. 12, pp. 3881–3889, Jun. 2018, doi: 10.4049/jimmunol.1800171.

[144] J.-W. Sidhom and A. S. Baras, “Deep learning identifies antigenic determinants of severe SARS-CoV-2 infection within T-cell repertoires,” *Sci Rep*, vol. 11, no. 1, Art. no. 1, Jul. 2021, doi: 10.1038/s41598-021-93608-8.

[145] G. K. Sandve and V. Greiff, “Access to ground truth at unconstrained size makes simulated data as indispensable as experimental data for bioinformatics methods development and benchmarking,” *Bioinformatics*, vol. 38, no. 21, pp. 4994–4996, Nov. 2022, doi: 10.1093/bioinformatics/btac612.

[146] C. Kanduri *et al.*, “Profiling the baseline performance and limits of machine learning models for adaptive immune receptor repertoire classification,” *Gigascience*, vol. 11, p. giac046, May 2022, doi: 10.1093/gigascience/giac046.

[147] Z. Sethna, G. Isacchini, T. Dupic, T. Mora, A. M. Walczak, and Y. Elhanati, “Population variability in the generation and selection of T-cell repertoires,” *PLOS Computational Biology*, vol. 16, no. 12, p. e1008394, Dec. 2020, doi: 10.1371/journal.pcbi.1008394.

[148] G. Isacchini, A. M. Walczak, T. Mora, and A. Nourmohammad, “Deep generative selection models of T and B cell receptor repertoires with soNNia,” *Proc Natl Acad Sci U S A*, vol. 118, no. 14, Apr. 2021, doi: 10.1073/pnas.2023141118.

[149] Y. Safonova, A. Lapidus, and J. Lill, “IgSimulator: a versatile immunosequencing simulator,” *Bioinformatics*, vol. 31, no. 19, pp. 3213–3215, Oct. 2015, doi: 10.1093/bioinformatics/btv326.

[150] A. Yermanos *et al.*, “Comparison of methods for phylogenetic B-cell lineage inference using time-resolved antibody repertoire simulations (AbSim),” *Bioinformatics*, vol. 33, no. 24, pp. 3938–3946, Dec. 2017, doi: 10.1093/bioinformatics/btx533.

[151] C. R. Weber *et al.*, “immuneSIM: tunable multi-feature simulation of B- and T-cell receptor repertoires for immunoinformatics benchmarking,” *Bioinformatics*, vol. 36, no. 11, pp. 3594–3596, Jun. 2020, doi: 10.1093/bioinformatics/btaa158.

[152] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *J. Mach. Learn. Res.*, vol. 12, no. null, pp. 2825–2830, Nov. 2011.

[153] A. Paszke *et al.*, “PyTorch: An Imperative Style, High-Performance Deep Learning Library,” in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2019. Accessed: Aug. 18, 2023. [Online]. Available:

https://papers.nips.cc/paper_files/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html
[154] M. Abadi *et al.*, "TensorFlow: a system for large-scale machine learning," in *Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation*, in OSDI'16. USA: USENIX Association, Nov. 2016, pp. 265–283.
[155] K. M. Chen, E. M. Cofer, J. Zhou, and O. G. Troyanskaya, "Selene: a PyTorch-based deep learning library for sequence data," *Nat Methods*, vol. 16, no. 4, pp. 315–318, Apr. 2019, doi: 10.1038/s41592-019-0360-8.
[156] A. Tomic, I. Tomic, Y. Rosenberg-Hasson, C. L. Dekker, H. T. Maecker, and M. M. Davis, "SIMON, an Automated Machine Learning System, Reveals Immune Signatures of Influenza Vaccine Responses," *J Immunol*, vol. 203, no. 3, pp. 749–759, Aug. 2019, doi: 10.4049/jimmunol.1900033.
[157] Z. Wu *et al.*, "MoleculeNet: a benchmark for molecular machine learning," *Chem. Sci.*, vol. 9, no. 2, pp. 513–530, Jan. 2018, doi: 10.1039/C7SC02664A.
[158] V. I. Nazarov, V. O. Tsvetkov, and E. Rumynskiy, "immunarch: An R Package for Painless Bioinformatics Analysis of T-Cell and B-Cell Immune Repertoires." ImmunoMind, Aug. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3367200
[159] C. R. Weber *et al.*, "Reference-based comparison of adaptive immune receptor repertoires," *Cell Rep Methods*, vol. 2, no. 8, p. 100269, Aug. 2022, doi: 10.1016/j.crmeth.2022.100269.
[160] M. Shugay *et al.*, "VDJtools: Unifying Post-analysis of T Cell Receptor Repertoires," *PLoS Comput Biol*, vol. 11, no. 11, Nov. 2015, doi: 10.1371/journal.pcbi.1004503.
[161] A. Montemurro, L. E. Jessen, and M. Nielsen, "NetTCR-2.1: Lessons and guidance on how to develop models for TCR specificity predictions," *Frontiers in Immunology*, vol. 13, 2022, Accessed: Apr. 25, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fimmu.2022.1055151
[162] W. S. Ye, "Adding 3D structure support to immuneML," Master thesis, 2023. Accessed: Aug. 28, 2023. [Online]. Available: https://www.duo.uio.no/handle/10852/103884
[163] O. Nybø, "Generation probability of immune receptors and full sequence implanting in adaptive immune receptor repertoires," Master thesis, 2023. Accessed: Aug. 28, 2023. [Online]. Available: https://www.duo.uio.no/handle/10852/103899
[164] M. B. Kristensen, "Incorporation of Dimensionality Reduction Methods into immuneML," Master thesis, 2023. Accessed: Aug. 28, 2023. [Online]. Available: https://www.duo.uio.no/handle/10852/103883
[165] M. T. S. Bergersen, "Introducing a Generative Model Pipeline in immuneML," Master thesis, 2023. Accessed: Aug. 28, 2023. [Online]. Available: https://www.duo.uio.no/handle/10852/103917
[166] I. M. Volan, "Integrating Clustering for Adaptive Immune Receptors in immuneML," Master thesis, 2023. Accessed: Aug. 28, 2023. [Online]. Available: https://www.duo.uio.no/handle/10852/103860
[167] M. Fjeld, "Streamlining the use of parallelization in a software platform for domain-tailored machine learning analyses," Master thesis, 2023. Accessed: Aug. 28, 2023. [Online]. Available: https://www.duo.uio.no/handle/10852/103846
[168] I. Papadopoulou, A.-P. Nguyen, A. Weber, and M. R. Martínez, "DECODE: a computational pipeline to discover T cell receptor binding rules," *Bioinformatics*, vol. 38, no. Suppl 1, pp. i246–i254, Jun. 2022, doi: 10.1093/bioinformatics/btac257.
[169] M. Chernigovskaya et al., "Simulation of adaptive immune receptors and repertoires with complex immune information to guide the development and benchmarking of AIRR machine learning." bioRxiv, p. 2023.10.20.562936, Nov. 01, 2023. doi: 10.1101/2023.10.20.562936.
[170] C. Kanduri *et al.*, "simulation of adaptive immune repertoires with realistic receptor

sequence sharing for benchmarking of immune state prediction methods” *GigaScience*, vol.12, p. giad074, Oct. 2023, doi: 10.1093/gigascience/giad074.

[171] J. Liu *et al.*, “Towards Out-Of-Distribution Generalization: A Survey.” arXiv, Jul. 27, 2023. doi: 10.48550/arXiv.2108.13624.

Papers

I

ARTICLES
https://doi.org/10.1038/s42256-021-00413-z

nature machine intelligence

Check for updates

The immuneML ecosystem for machine learning analysis of adaptive immune receptor repertoires

Milena Pavlović[1,2,3,19], Lonneke Scheffer[1,2,19], Keshav Motwani[4], Chakravarthi Kanduri[2], Radmila Kompova[2], Nikolay Vazov[6], Knut Waagan[6], Fabian L. M. Bernal[6], Alexandre Almeida Costa[7], Brian Corrie[8], Rahmad Akbar[9], Ghadi S. Al Hajj[1], Gabriel Balaban[1,2], Todd M. Brusko[4,5], Maria Chernigovskaya[9], Scott Christley[10], Lindsay G. Cowell[11], Robert Frank[9], Ivar Grytten[1,2], Sveinung Gundersen[2], Ingrid Hobæk Haff[11], Eivind Hovig[1,2,13], Ping-Han Hsieh[14], Günter Klambauer[12], Marieke L. Kuijjer[14,15], Christin Lund-Andersen[13,16], Antonio Martini[1], Thomas Minotto[11], Johan Pensar[11], Knut Rand[1,2], Enrico Riccardi[1,2], Philippe A. Robert[9], Artur Rocha[7], Andrei Slabodkin[9], Igor Snapkov[9], Ludvig M. Sollid[3,9], Dmytro Titov[2], Cédric R. Weber[17], Michael Widrich[12], Gur Yaari[18], Victor Greiff[9,20] and Geir Kjetil Sandve[1,2,3,20] ✉

Adaptive immune receptor repertoires (AIRR) are key targets for biomedical research as they record past and ongoing adaptive immune responses. The capacity of machine learning (ML) to identify complex discriminative sequence patterns renders it an ideal approach for AIRR-based diagnostic and therapeutic discovery. So far, widespread adoption of AIRR ML has been inhibited by a lack of reproducibility, transparency and interoperability. immuneML (immuneml.uio.no) addresses these concerns by implementing each step of the AIRR ML process in an extensible, open-source software ecosystem that is based on fully specified and shareable workflows. To facilitate widespread user adoption, immuneML is available as a command-line tool and through an intuitive Galaxy web interface, and extensive documentation of workflows is provided. We demonstrate the broad applicability of immuneML by (1) reproducing a large-scale study on immune state prediction, (2) developing, integrating and applying a novel deep learning method for antigen specificity prediction and (3) showcasing streamlined interpretability-focused benchmarking of AIRR ML.

T-cell receptors (TCRs) and B-cell receptors (BCRs), which are collectively known as adaptive immune receptor (AIR) repertoires (AIRRs), recognize antigens and record information on past and ongoing immune responses[1–4]. AIRR-encoded information is particularly useful for the repertoire-based prediction and analysis of immune states (for example, health, disease, infection, vaccination) in relation to other metadata such as major histocompatibility complex (MHC)[5–7], age[7,8] and sex[9]. Together, this information shapes the foundation for AIRR-based diagnostics[6,10–14]. Similarly, sequence-based prediction of antigen and epitope binding is of fundamental importance for AIR-based therapeutics discovery and engineering[15–27]. In this Article, the term AIRR signifies both AIRs and AIRRs (a collection of AIRs) unless specified otherwise.

Machine learning (ML) has recently taken centre stage in the biological sciences because it allows the detection, recovery and recreation of high-complexity biological information from large-scale biological data[28–31]. AIRRs have complex biology with specialized research questions, such as immune state and receptor-specificity prediction, that warrant domain-specific ML analysis[15]. Briefly, (1) ~10^8–10^{10} distinct AIRs exist in a given individual at any one time[32–34], with little overlap among individuals, necessitating encodings that allow the detection of predictive patterns. These shared patterns may correspond to full-length AIRs[6], subsequences or[16] alternative AIR representations[11,12,17,18,22,35–37]. (2) In repertoire-based ML, the patterns relevant to any immune state may be as rare as one antigen-binding AIR per million lymphocytes in a repertoire[38], translating to a very low rate of relevant sequences per repertoire

[1]Department of Informatics, University of Oslo, Oslo, Norway. [2]Centre for Bioinformatics, University of Oslo, Oslo, Norway. [3]K.G. Jebsen Centre for Coeliac Disease Research, Institute of Clinical Medicine, University of Oslo, Oslo, Norway. [4]Department of Pathology, Immunology and Laboratory Medicine, College of Medicine, University of Florida Diabetes Institute, Gainesville, FL, USA. [5]Department of Pediatrics, College of Medicine, University of Florida Diabetes Institute, Gainesville, FL, USA. [6]University Center for Information Technology, University of Oslo, Oslo, Norway. [7]Institute for Systems and Computer Engineering, Technology and Science, Porto, Portugal. [8]Biological Sciences, Simon Fraser University, Burnaby, British Columbia, Canada. [9]Department of Immunology, University of Oslo and Oslo University Hospital, Oslo, Norway. [10]Department of Population and Data Sciences, UT Southwestern Medical Center, Lawton, OK, USA. [11]Department of Mathematics, University of Oslo, Oslo, Norway. [12]ELLIS Unit Linz and LIT AI Lab, Institute for Machine Learning, Johannes Kepler University Linz, Linz, Austria. [13]Department of Tumor Biology, Institute for Cancer Research, Oslo University Hospital, The Norwegian Radium Hospital, Oslo, Norway. [14]Centre for Molecular Medicine Norway (NCMM), Nordic EMBL Partnership, University of Oslo, Oslo, Norway. [15]Department of Pathology, Leiden University Medical Center, Leiden, the Netherlands. [16]Faculty of Medicine, Institute of Clinical Medicine, University of Oslo, Oslo, Norway. [17]Department of Biosystems Science and Engineering, ETH Zürich, Zurich, Switzerland. [18]Faculty of Engineering, Bar Ilan University, Ramat Gan, Israel. [19]These authors contributed equally: Milena Pavlović, Lonneke Scheffer. [20]These authors jointly supervised this work: Victor Greiff, Geir Kjetil Sandve. ✉e-mail: geirksa@ifi.uio.no

(low witness rate)[11,39,40]. (3) In sequence-based ML, the enormous diversity of antigen recognition combined with polyreactivity points to complex high-order statistical dependencies in the short sequence known to be the main determinant of antigen recognition (complementarity-determining region 3, CDR3)[1,16].

Tailored ML frameworks and platforms that account for the idiosyncrasies of the underlying data have been published for applications in genomics[41,42], proteomics[43,44], biomedicine[45] and chemistry[46]. Their creation recognizes the infeasibility of defining, implementing and training appropriate ML models by relying solely on generic ML frameworks such as scikit-learn[47] or PyTorch[48]. The lack of a standardized framework for AIRR ML has led to heterogeneity in terms of technical solutions, domain assumptions and user interaction options, hampering transparent comparative evaluation and the ability to explore and select the ML methodology most appropriate for a given study[15].

Results

Overview of immuneML. Here, we present immuneML, an open-source collaborative ecosystem for AIRR ML (Fig. 1). immuneML enables the ML study of both experimental and synthetic AIRR-seq data that are labelled on the repertoire level (for example, immune state, sex, age or any other metadata) or sequence level (for example, antigen binding), all the way from preprocessing to model training and model interpretation. It natively implements model selection and assessment procedures such as nested cross-validation to ensure robustness in selecting the ML model. immuneML may be operated either via the command line or the Galaxy web interface[49], which offers an intuitive user interface that promotes collaboration and reusability through shareable analysis histories. To expedite analyses, immuneML may also be deployed via cloud services such as Amazon Web Services (AWS) and Google Cloud, or on a local server when there are data privacy concerns. Computational reproducibility and transparency are achieved by shareable specification files, which include all analysis details (Supplementary Fig. 1). immuneML's compliance with AIRR community software and sequence annotation standards[50,51] ensures straightforward integration with third-party tools for AIRR data preprocessing and downstream analysis of AIRR ML results. For example, immuneML is fully compatible with the sequencing read processing and annotation suite MiXCR[52] and the Immcantation[53,54] and immunarch[55] frameworks for AIRR data analysis. AIRR data from the AIRR Data Commons[56] through the iReceptor Gateway[57], as well as the epitope-specific TCR database VDJdb[58] may be directly downloaded into the immuneML Galaxy environment. immuneML is also integrated with the AIRR-specific attention-based multiple-instance learning ML method DeepRC[39] and the TCR-specific clustering method TCRdist[17], and is compatible with GLIPH2[59].

To get started with immuneML, we refer the reader to Box 1. To demonstrate immuneML's capabilities for performing AIRR ML, we provide an overview of the main features of the platform and then highlight three orthogonal use cases: (1) we reproduce the cytomegalovirus (CMV) serostatus prediction study of Emerson et al.[6] inside immuneML and examine the robustness of the approach, showing one way of using immuneML for repertoire-based immune state prediction, (2) we apply a new custom convolutional neural network (CNN) for the sequence-based task of antigen-binding prediction based on paired-chain TCR data and (3) we show the use of immuneML for benchmarking AIRR ML methods.

immuneML allows read-in of experimental single- and paired-chain data from offline and online sources, as well as the generation of synthetic data for ML benchmarking. Experimental data may be read-in directly if they comply with the major formats used for AIRR-seq data V(D)J annotation: AIRR-C standard-conforming[50], MIXCR[52], 10x Genomics[60], Adaptive Biotechnologies ImmunoSEQ[6,61] or VDJdb formats[58]. The AIRR-C format compatibility ensures that synthetic data generated by immuneSIM[62] can also be imported, as can synthetic data generated by IGoR[63] and OLGA[64]. Moreover, immuneML can be configured to read in data from any custom tabular format. To facilitate access to large-scale AIRR-seq data repositories, we provide Galaxy[49] tools for downloading data from the AIRR Data Commons[56] via the iReceptor Gateway[57] and from VDJdb[58] into the Galaxy environment for subsequent ML analysis. Furthermore, immuneML has built-in capacities for complex synthetic AIRR data generation to satisfy the need for ground-truth data in the context of ML method benchmarking. Finally, read-in data may be filtered by clone count, metadata and chain.

immuneML supports multiple ML frameworks and allows the interpretation of ML models. immuneML supports two major ML platforms to ensure flexibility—scikit-learn[47] and PyTorch[48]— and is therefore compliant with all ML methods inside these platforms. immuneML features scikit-learn implementations such as logistic regression, support vector machine and random forest. In addition, we provide AIRR-adapted ML methods. Specifically, for repertoire classification, immuneML includes a custom implementation of the method published by Emerson et al.[6], as well as the attention-based deep learning method DeepRC[39]. For paired-chain sequence-based prediction, immuneML includes a custom-implemented CNN-based deep learning method, integrates with TCRdist[17] and is compatible with GLIPH2[59]. immuneML also includes several encodings that are commonly used for AIRR data, such as *k*-mer frequency decomposition, one-hot encoding (where each position in the sequence is represented by a vector of zeros except one entry containing 1, the denoting appropriate amino acid or nucleotide), encoding via the presence of disease-associated sequences, and repertoire distances. For the full overview of analysis components, see Supplementary Table 1.

A variety of tabular and graphical analysis reports may be automatically generated during an analysis, providing details about the encoded data (for example, feature value distributions), the ML model (for example, interpretability reports) and the prediction accuracy (a variety of performance metrics across training, validation and test sets). The trained models can also be exported and used in future analyses.

immuneML facilitates the reproducibility, interoperability and transparency of ML models. immuneML draws on a broad range of techniques and design choices to ensure that it meets the latest expectations with regard to usability, reproducibility, interoperability, extensibility and transparency[65–68] (Fig. 1).

Usability is achieved by a range of installation and usage options, catering to novices and experts, and to small and large-scale analyses. A Galaxy web interface[49] allows users to run analyses without the need for any installation and without requiring any skills in programming or command-line operations. Its availability through GitHub, pip and Docker streamlines usage at scales ranging from laptops to high-performance infrastructures such as Google Cloud and AWS (docs.immuneml.uio.no/latest/installation/cloud.html).

Reproducibility is ensured by leveraging the Galaxy framework[49], which enables sharing of users' analysis histories, including the data and parameters, so that they can be independently reproduced. If working outside Galaxy, reproducibility is ensured by shareable analysis specification (YAML) files. YAML specification files produced in the Galaxy web interface can also be downloaded to seamlessly switch between Galaxy and command-line operation. Note that here we are referring to reproducibility in the sense of repeating a computational analysis in its exact form, also referred to as methods reproducibility[69], although the YAML files are also

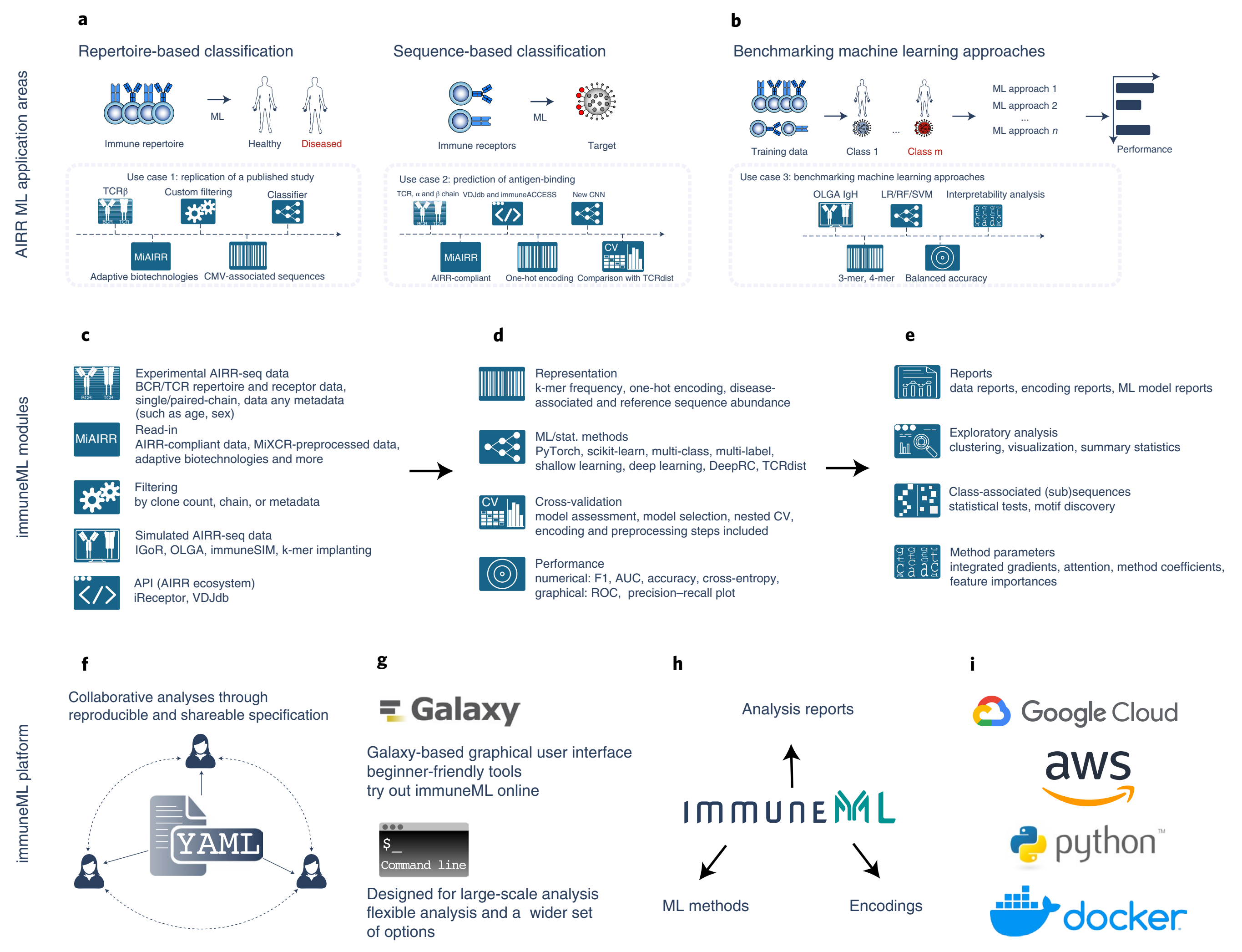

AIRR ML application areas
a
Repertoire-based classification
Immune repertoire
ML
Healthy
Diseased
Sequence-based classification
Immune receptors
ML
Target
Use case 1: replication of a published study
TCRβ
Custom filtering
Classifier
MiAIRR
Adaptive biotechnologies
CMV-associated sequences
Use case 2: prediction of antigen-binding
TCR, α and β chain
VDJdb and immuneACCESS
New CNN
MiAIRR
AIRR-compliant
One-hot encoding
Comparison with TCRdist
b
Benchmarking machine learning approaches
Training data
Class 1
Class m
ML approach 1
ML approach 2
ML approach n
Performance
Use case 3: benchmarking machine learning approaches
OLGA IgH
LR/RF/SVM
Interpretability analysis
3-mer, 4-mer
Balanced accuracy
immuneML modules
c
Experimental AIRR-seq data
BCR/TCR repertoire and receptor data, single/paired-chain, data any metadata (such as age, sex)
Read-in
AIRR-compliant data, MiXCR-preprocessed data, adaptive biotechnologies and more
Filtering
by clone count, chain, or metadata
Simulated AIRR-seq data
IGoR, OLGA, immuneSIM, k-mer implanting
API (AIRR ecosystem)
iReceptor, VDJdb
d
Representation
k-mer frequency, one-hot encoding, disease-associated and reference sequence abundance
ML/stat. methods
PyTorch, scikit-learn, multi-class, multi-label, shallow learning, deep learning, DeepRC, TCRdist
Cross-validation
model assessment, model selection, nested CV, encoding and preprocessing steps included
Performance
numerical: F1, AUC, accuracy, cross-entropy, graphical: ROC, precision–recall plot
e
Reports
data reports, encoding reports, ML model reports
Exploratory analysis
clustering, visualization, summary statistics
Class-associated (sub)sequences
statistical tests, motif discovery
Method parameters
integrated gradients, attention, method coefficients, feature importances
immuneML platform
f
Collaborative analyses through reproducible and shareable specification
YAML
g
Galaxy
Galaxy-based graphical user interface
beginner-friendly tools
try out immuneML online
$_
Command line
Designed for large-scale analysis
flexible analysis and a wider set of options
h
Analysis reports
immuneML
ML methods
Encodings
i
Google Cloud
aws
python
docker

Fig. 1 | Overview of immuneML. a,**b**, The main immuneML application areas are sequence- and repertoire-based classification of AIRRs with application to immunodiagnostics and therapeutics research (**a**) and the development of AIRR-based methods (**b**). We present three use cases belonging to these application areas (use cases 1–3). **c**–**e**, The immuneML core is composed of three pillars: AIRR-seq data input and filtering (**c**), ML (**d**) and interpretability analysis (**e**). Each of these pillars has different modules that may be interconnected to build an immuneML workflow. **f**, immuneML uses a specification file (YAML) that is customizable and allows full reproducibility and shareability with collaborators or the broader research community. An overview of immuneML analyses is given in Supplementary Fig. 1. **g**, immuneML may be operated via the Galaxy web interface or the command line. **h**, All immuneML modules are extendable. Documentation for developers is available online. **i**, immuneML is available as a Python package, a Docker image and may be deployed to cloud frameworks (for example, AWS, Google Cloud). 3-mer, amino acid sequence of length three; LR, logistic regression; RF, random forest; SVM, support vector machine; AUC, area under the curve; ROC, receiver-operating characteristic curve; IgH, immunoglobulin heavy chain; F1, F1-score; iReceptor[57]; immuneSIM[62].

well suited to exploring the extent to which results are affected by modifications of analysis parameters.

Interoperability is ensured by supporting data import from multiple sources and export to AIRR-C format (MiAIRR standard) for post-analysis by third-party tools that are AIRR-compliant[50].

The extensibility of immuneML, signifying straightforward inclusion of new ML methods, encodings, reports and preprocessing, is ensured by its modular design (Supplementary Fig. 2). The code is open source and available on GitHub (Box 2). The documentation details step-by-step developer tutorials for immuneML extension (docs.immuneml.uio.no/latest/developer_docs.html).

Transparency is established by (1) a YAML analysis specification in which the assumptions of the AIRR ML analysis are explicitly defined, and default parameter settings are exported; (2) separate immunologist-centric Galaxy user interfaces that translate parameters and assumptions of the ML process to aspects of immune receptors that immunologists may better relate to (Supplementary Fig. 3) and (3) making the underlying data for each analysis report available for further inspection by the user.

Use case 1—reproduction of a published study inside immuneML. To show how a typical AIRR ML analysis may be performed within immuneML, we reproduced a previously published study by Emerson et al. on the TCRβ-repertoire-based classification of individuals into CMV seropositive and seronegative[6] (Fig. 2a). Using the standard interface of immuneML, we set up a repertoire classification analysis using tenfold cross-validation on cohort 1 of 563 patients to choose optimal hyperparameters for immuneML's native implementation of the statistical classifier introduced by Emerson and colleagues[6]. We then retrained the classifier on the complete cohort 1 and tested it on a second cohort (cohort 2) of 120 patients, as described in the original publication (Methods).

Box 1 | Getting started with immuneML

Visit the project website at immuneml.uio.no. immuneML may be used (1) online via the Galaxy web interface (galaxy.immuneml.uio.no), (2) through a Docker container or (3) from the command line by installing and running immuneML as a Python package. Detailed instructions for each of these options are available in the immuneML documentation: docs.immuneml.uio.no/latest/installation.html.

Getting started—web interface

- For immunologists, we recommend the Quickstart guide based on simplified interfaces for training ML models: docs.immuneml.uio.no/latest/quickstart/galaxy_simple.html. Explanations of the relevant ML concepts can be found in the documentation (sequence classification docs.immuneml.uio.no/latest/galaxy/galaxy_simple_receptors.html and repertoire classification docs.immuneml.uio.no/latest/galaxy/galaxy_simple_repertoires.html)
- Alternatively, to have full control over all details of the analysis, see the YAML-based Galaxy Quickstart guide: docs.immuneml.uio.no/latest/quickstart/galaxy_yaml.html.
- For guidance on how to use each immuneML Galaxy tool, see the immuneML and Galaxy documentation (docs.immuneml.uio.no/latest/galaxy.html) and the list of published example Galaxy histories (galaxy.immuneml.uio.no/histories/list_published).

Getting started—command-line interface

- For the command-line Quickstart guide, see docs.immuneml.uio.no/latest/quickstart/cli_yaml.html.
- For detailed examples of analyses that can be performed with immuneML, see the tutorials (docs.immuneml.uio.no/latest/tutorials.html), use case examples (docs.immuneml.uio.no/latest/usecases.html) and all of the supported analysis options in the YAML specification documentation (docs.immuneml.uio.no/latest/specification.html).

For any questions, contact us at contact@immuneml.uio.no, visit the troubleshooting page in the documentation (docs.immuneml.uio.no/latest/troubleshooting.html) or open an issue on our GitHub repository (github.com/uio-bmi/immuneML/issues).

Box 2 | How to contribute to immuneML

There are multiple avenues for contributing to and extending immuneML:

- ML workflows for specific research questions can be shared on galaxy.immuneml.uio.no, which allows other researchers to use them directly in their own data analysis.
- Questions, enhancements or encountered bugs can be reported as issues via the immuneML GitHub (github.com/uio-bmi/immuneML/issues).
- To improve or extend the immuneML platform, the source code can be obtained from GitHub at github.com/uio-bmi/immuneML. The immuneML codebase is described in the immuneML developer documentation docs.immuneml.uio.no/latest/developer_docs.html, along with tutorials on how to add new ML methods, encodings and report components to the platform. A new ML method could initially be developed as a separate component and subsequently integrated into immuneML to benefit from available immuneML functionalities related to importing datasets from different formats, using various data representations, benchmarking against existing methods and robustly assessing the performance—all through a convenient user interface.
- We encourage developers to contribute their improvements and extensions to the community, either by making their own versions public or by submitting their contributions as GitHub 'pull requests' to the main immuneML codebase.

immuneML exports classifier details, such as a list of immune-status-associated sequences for each classifier created during cross-validation, as well as a performance overview using the metrics of choice. We replicated the predictive performance achieved by Emerson et al.[6], finding 143 of the same CMV-associated TCRs (out of 164) reported in the original study.

We then used the built-in robustness analysis of immuneML to explore how the classification accuracy and the set of immune-status-associated sequences varied when learning classifiers were based on smaller subsets of repertoires (Fig. 2a,b). While the exact set of learned immune-status-associated sequences varied across subsampled data of sizes close to the full dataset, the classification accuracy was nonetheless consistently high (>0.85) as long as the number of training repertoires was 400 or higher (below this, classification accuracy on the separate test sets deteriorated sharply) (Fig. 2b,c).

Use case 2—extending immuneML with a deep learning component for antigen-specificity prediction based on paired-chain (single-immune-cell) data. To illustrate the extensibility of the immuneML platform, we added a new CNN component for predicting antigen specificity based on paired-chain AIR data. The ML task was to discover motifs in the two receptor chains (sequences) and to exploit the presence of these motifs to predict whether the receptor would bind the antigen. As the immuneML platform provides comprehensive functionality for parsing and encoding paired-chain data, hyperparameter optimization and presenting results, the only development step needed was to add the code for the CNN-based method itself (Supplementary Fig. 5). Briefly, the added CNN consisted of a set of kernels for each chain that act as motif detectors, a vector representation of the receptor obtained by combining all kernel activations and a fully connected layer that predicts whether the receptor would bind the antigen or not. Furthermore, we show how to run analyses with the added component and compare its results with those of alternative models, such as a logistic regression model based on 3-mer frequencies and a *k*-nearest neighbour classifier relying on TCRdist[17] as the distance metric (available directly from immuneML through the tcrdist3 package[70]). We also show that the motifs can be recovered from the CNN model, the logistic regression, TCRdist and GLIPH2[59] (Fig. 2d).

Use case 3—ML methods benchmarking on ground-truth synthetic data. Given the current rise in AIRR ML applications, the ability of method developers and practitioners to efficiently benchmark the variety of available approaches is becoming crucial[1,15,62]. Owing to the limited current availability of high-resolution labelled experimental data, rigorous benchmarking relies on a combination of experimental and simulated ground-truth data. The immuneML platform natively supports both the generation of synthetic data for benchmarking purposes and the efficient comparative benchmarking of multiple methodologies based on synthetic (as well as experimental) data. To exhibit the efficiency with which such benchmarking can be performed within the immuneML framework, we simulated 2,000 human IgH repertoires consisting of 10^5 CDR3 amino acid sequences each using the OLGA framework[64], and implanted sequence motifs reflecting five different immune

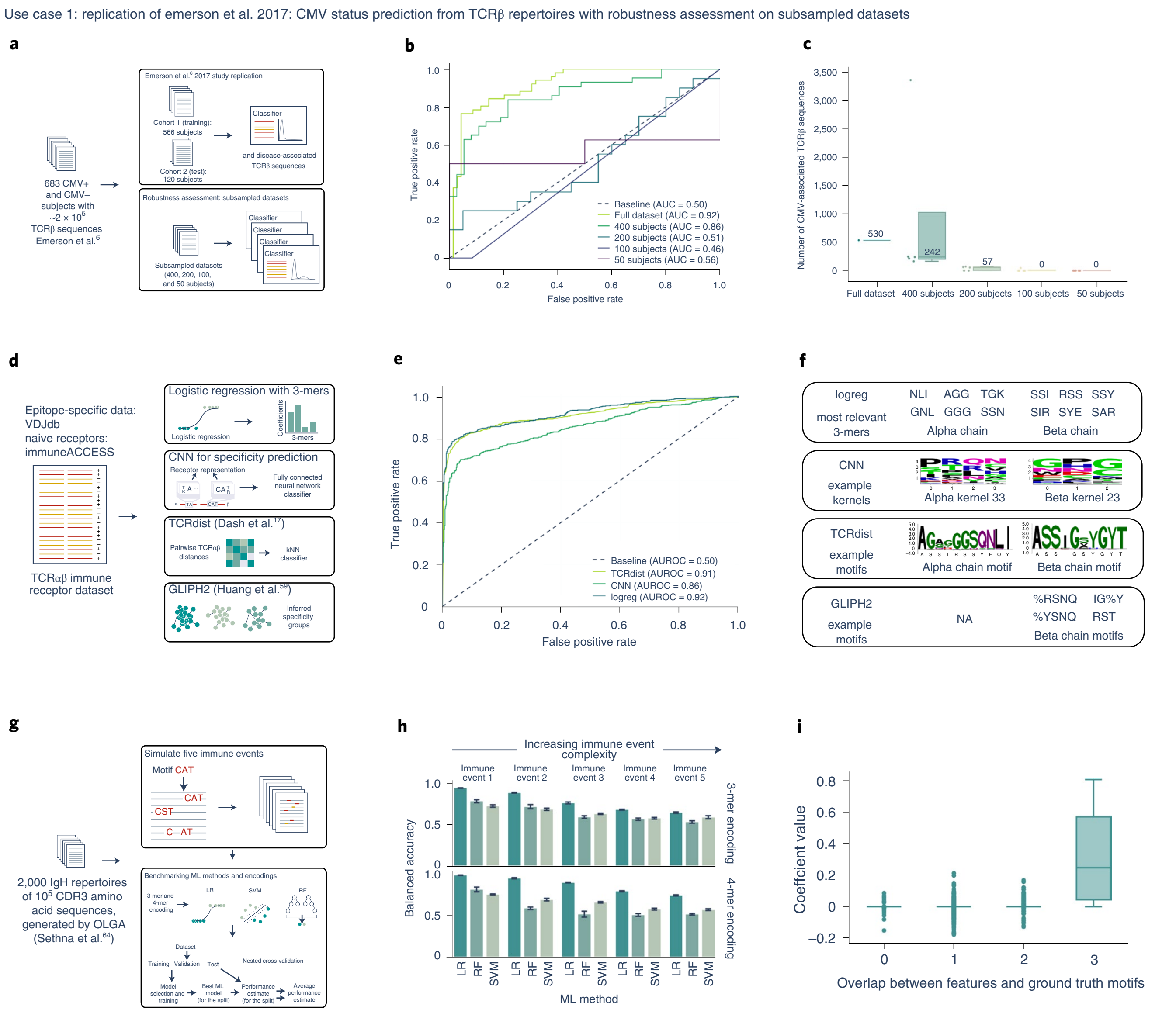


Fig. 2 | Use cases demonstrating ML model training, benchmarking and platform extension. Use cases 1–3 exemplify immuneML usage. **a**–**c**, Use case 1 is the reproduction of a published study[6] . **a**, The task consisted of distinguishing between TCRβ repertoires from CMV seropositive and seronegative individuals, as well as identifying TCRβ sequences that are associated with CMV status. **b**, We assessed the robustness of the statistical approach, measured by the predictive performance, as a function of decreasing dataset size: fewer repertoires (400, 200, 100 and 50) led to decreased prediction accuracy (AUROC: 0.86–0.46). **c**, Fewer CMV-associated TCRβ sequences were found with fewer subject (with almost none found in datasets of 100 and 50 subjects). **d**–**f**, For use case 2, we developed a new ML method for antigen-specificity prediction on paired-chain T-cell receptor data using a CNN architecture. **d**, The task here was antigen-binding prediction. The method separately detected motifs in paired chains and combined the motif scores (corresponding to kernel activations) to obtain the receptor representation, which was then used as input to a classifier. **e**, We compared the CNN method with the TCRdist-based *k*-nearest neighbour classifier and logistic regression on a dataset consisting of epitope-specific and naive TCRαβ sequences (assumed to be non-epitope-specific). For epitope-specific sequences, we used Epstein–Barr-virus-specific TCRαβ sequences binding to the GILGFVFTL epitope. **f**, The motifs recovered by CNN, TCRdist and GLIPH2 among the epitope-specific sequences. **g**–**i**, In use case 3, we show how ground-truth synthetic data can be used to benchmark AIRR ML methods. The dataset consisted of 2,000 immune repertoires generated by OLGA[64]. Using immuneML, five immune events of increasing complexity were simulated by implanting synthetic signals into the OLGA-generated repertoires. **g**, This dataset was used to benchmark three different ML methods in combination with two encodings (3-mer and 4-mer encoding) inside immuneML, showing the classification performance with a standard deviation that dropped as the immune event complexity increased. **h**, The quality of the ML models was assessed by comparing the feature coefficient sizes with how well these features represented the ground-truth signals. **i**, Models with a good classification performance were in fact able to recover the ground-truth signals. The overlap was calculated from a logistic regression model for immune event 3. Error bars in **h** represent the standard deviation.

events of varying complexity (Fig. 2g and Supplementary Table 2). We examined the classification accuracy of three assessed ML methods (Fig. 2h) and used a native immuneML report to examine the overlap between ground-truth implanted motifs and learned model features (Fig. 2i and Supplementary Fig. 6).

Discussion

We have presented immuneML, a collaborative and open-source platform for transparent AIRR ML that is accessible both via the command line and via an intuitive Galaxy web interface[49]. immuneML supports the analysis of both BCR and TCR repertoires, with single or paired chains, at the sequence (receptor) and repertoire level. It accepts experimental data in a variety of formats and includes native support for generating synthetic AIRR data to benchmark the performance of AIRR ML approaches. As a flexible platform for tailoring AIRR ML analyses, immuneML features a broad selection of modular software components for data import, feature encoding, ML and performance assessment (Supplementary Table 1). The platform can easily be extended with new encodings, ML methods and analytical reports by the research community. immuneML supports all major standards in the AIRR field, uses YAML analysis specification files for transparency and scales from local machines to the cloud. Throughout the platform development phase, we have tried to adhere to best practices of software engineering to improve software extensibility and maintainability. With the field of ML maturing, we see such aspects connected to the longevity and interoperability of ML functionality as increasingly deserving of attention. Extensive documentation for both users and contributors is available (docs.immuneml.uio.no).

immuneML caters to a variety of user groups and usage contexts. The Galaxy web tools make sophisticated ML-based receptor-specificity and repertoire immune-state prediction accessible to immunologists and clinicians through intuitive, graphical interfaces. The diversity of custom preprocessing and encoding used in published AIRR ML studies hinders their comparison and reproducibility. In contrast, the YAML-based specification of analyses on the command line or through Galaxy improves the potential for collaboration, transparency and reproducibility of AIRR ML for experienced bioinformaticians and data scientists. The integrated support for AIRR data simulation and systematic ML method benchmarking helps method users to select those approaches most appropriate to their analytical setting, and to assists method developers in effectively evaluating ML-related methodological ideas.

From a developer perspective, the impressive sophistication of generic ML frameworks such as TensorFlow[71] and PyTorch[48] may suggest that these frameworks would suffice as a starting point for AIRR ML method development. These frameworks are, however, limited to the specification of ML methods on generic data representations—meaning that it is up to every AIRR ML developer to implement (reinvent) all remaining parts of a full AIRR workflow, including data read-in, preprocessing, hyperparameter optimization strategies, interpretability and presentation of results. The fact that the immuneML architecture builds on top of frameworks such as PyTorch underlines the breadth of additional functionality needed for robust ML development and execution in the AIRR domain. For ML researchers, the rich support for integrating novel ML components within existing code for data processing, hyperparameter optimization and performance assessment could greatly accelerate method development.

The current version of immuneML includes a set of components mainly focused on supervised ML, but the platform is also suitable for the community to extend it with components for settings such as unsupervised learning[72] or generative receptor modelling[15,20,73]. We also aim to improve the general support for model introspection, particularly with regards to supporting causal interpretations for discovering and alleviating technical biases or challenges related to the study design[74].

In conclusion, immuneML enables the transition from AIRR ML method set-up representing a bona fide research project to being at the fingertips of immunologists and clinicians. AIRR ML method developers can also focus on the implementation of components reflecting their unique research contribution, relying on existing immuneML functionality for the entire remaining computational process. immuneML facilitates the increased adoption of AIRR-based diagnostics and therapeutics discovery by supporting the accelerated development of AIRR ML methods.

Methods

immuneML availability. immuneML can be used (1) as a web tool through the Galaxy web interface (galaxy.immuneml.uio.no), (2) from a command-line interface, (3) through Docker (registry.hub.docker.com/r/milenapavlovic/immuneml), (4) via cloud services such as Google Cloud (cloud.google.com) through Docker integration or (5) as a Python library (pypi.org/project/immuneML). It is also deposited on Zenodo at https://doi.org/10.5281/zenodo.5118741 (ref. [75]).

immuneML analysis specification. immuneML analyses are specified using a YAML specification file (Supplementary Fig. 1), which allows streamlined specification of full analyses based on an external domain-specific language for AIRR ML[76]. When using Galaxy, the user may choose to provide a specification file directly or use a graphical interface that compiles the specification for the user. When used as a command-line interface tool, locally or in the cloud, with or without Docker, the specification file is provided by the user. Examples of specification files and detailed documentation on how to create them are available at docs.immuneml.uio.no/latest/tutorials/how_to_specify_an_analysis_with_yaml.html.

immuneML supports different types of instruction: (1) training and assessment of ML models, (2) applications of trained ML models, (3) exploratory data analysis and (4) generation of synthetic AIRR datasets. Tutorials detailing these instructions are available at docs.immuneml.uio.no/latest/tutorials.html.

immuneML public server. the immuneML Galaxy web interface is available at galaxy.immuneml.uio.no. In addition to core immuneML components, the Galaxy instance includes interfaces towards the VDJdb[58] database and the iReceptor Gateway[57]. The documentation for the Galaxy immuneML tools is available at docs.immuneml.uio.no/latest/galaxy.html.

immuneML architecture. immuneML has a modular architecture that can easily be extended (Supplementary Fig. 2). In particular, we have implemented glass-box extensibility mechanisms[77], which enable the creation of customized code to implement new functionalities (encodings, ML methods, reports) that might be needed by users. Such extensibility mechanisms allow users to adapt immuneML to their specific cases without the need to understand the complexity of the immuneML code. As an example, immuneML orchestrates the exploration (grid search) of alternative components for data processing, encodings and ML method hyperparameters on data subsets for the inner splits of a nested cross-validation, allowing newly developed components for either of these parts (data processing, encoding, ML method) to be selected in competition with existing components as part of an unbiased hyperparameter selection and prediction performance estimation. For tutorials on how to add a new ML method, encoding, or an analysis report, see the developer documentation: docs.immuneml.uio.no/latest/developer_docs.html.

Use cases. *Use case 1—reproduction of a published study inside immuneML.* We reproduced the study by Emerson and colleagues using a custom implementation of the encoding and classifier described in the original publication[6]. Out of the 786 subjects listed in the original study, we removed 103 subjects (1 with missing repertoire data, 25 with unknown CMV status, 3 with negative template counts for some of the sequences, and the rest with no template count information, all of which occurred in cohort 1), and performed the analysis on the remaining 683 subjects. We achieved comparable results to the original publication, as shown in Supplementary Fig. 4. Supplementary Table 3 shows TCRβ receptor sequences inferred to be CMV-associated, comparing them to those published by Emerson et al.

In addition to reproducing the study by Emerson et al., we retrained the classifier on datasets consisting of 400, 200, 100 and 50 TCRβ repertoires randomly subsampled from cohort 1 and cohort 2. We show how the performance and the overlap of CMV-associated sequences changes with such reductions of dataset size (Fig. 2b,c). While most of the results are consistent within the subsampled dataset size, in Fig. 2b, a less stringent P value threshold was selected during the hyperparameter optimization for one of the cross-validation splits for the dataset of 400 subjects, resulting in a higher number of CMV-associated sequences.

The YAML specification files for this use case are available in the immuneML documentation under use case examples: docs.immuneml.uio.no/latest/usecases/emerson_reproduction.html. The complete collection of results produced by immuneML, as well as the subsampled datasets, can be found in the NIRD research data archive[78].

Use case 2—extending immuneML with a deep learning component for antigen specificity prediction based on paired-chain (single-immune-cell) data. To demonstrate the ease of extensibility for the platform, we added a CNN-based receptor-specificity prediction ML method to the platform (Supplementary Fig. 5). Detailed instructions for adding such a new component to immuneML can be found in the developer documentation: docs.immuneml.uio.no/latest/developer_docs/how_to_add_new_ML_method.html. We subsequently ran the added component through the standard immuneML model training interface, comparing its predictive performance with TCRdist[17,70] and logistic regression across three datasets. We also recovered motifs from the kernels of the neural network by limiting the values of the kernels similar to Ploenzke and Irizarry[79], and from the hierarchical clustering based on TCRdist distance, and compared these recovered motifs with the motifs extracted by GLIPH2[59] on the same datasets. Each dataset included a set of epitope-specific TCRαβ receptors downloaded from VDJdb and a set of naive, randomly paired TCRαβ receptors from the peripheral blood samples of four healthy donors[80]. Epitope-specific datasets were specific to cytomegalovirus (KLGGALQAK epitope, with 13,000 paired TCRαβ receptors), Influenza A (GILGFVFTL epitope, with 2,000 paired TCRαβ receptors) and Epstein–Barr virus (AVFDRKSDAK epitope, with 1,700 paired TCRαβ receptors). Dataset details are summarized in Supplementary Table 4. The code for creating the datasets and YAML specifications describing the analysis can be found in the immuneML documentation: docs.immuneml.uio.no/latest/usecases/extendability_use_case.html. The three datasets of epitope-specific receptors and the complete collection of kernel visualizations produced by immuneML, as well as the results produced by GLIPH2, have been stored in the NIRD research data archive[81].

Use case 3—ML methods benchmarking on ground-truth synthetic data. To show the utility of immuneML for benchmarking AIRR ML methods, we constructed a synthetic AIR dataset with known implanted ground-truth signals and performed a benchmarking of ML methods and encodings inside immuneML. To create the dataset for this use case, 2,000 human IgH repertoires of 10^5 CDR3 amino acid sequences were generated using OLGA[64]. Subsequently, immuneML was used to simulate five different immune events of varying complexity by implanting signals containing probabilistic 3-mer motifs (Supplementary Table 2). The signals of each immune event were implanted in 50% of the repertoires, without correlating the occurrence of different immune events. Signals were implanted in 0.1% of the CDRH3 sequences of the repertoires selected for immune event simulation. While signal rates down to one antigen-binding AIR per million lymphocytes have been reported for certain disease states[38], we here chose a signal rate substantially higher than these most challenging cases to allow a demonstration of how benchmarking may be performed using basic ML approaches.

Using immuneML, three different ML methods (logistic regression, random forest and support vector machine) combined with two encodings (3-mer and 4-mer frequency encoding) were benchmarked. Hyperparameter optimization was done through nested cross-validation. For the model assessment (outer) cross-validation loop, the 2,000 repertoires were randomly split into 70% training and 30% testing data, and this was repeated three times. In the model selection (inner) cross-validation loop, threefold cross-validation was used. The test set classification performances of the trained classifiers for each immune event are shown in Fig. 2h.

The immune signals implanted in this dataset were used to examine the ability of the ML methods to recover ground-truth motifs by comparing the coefficient value (logistic regression, support vector machine) or feature importance (random forest) of a given feature with the overlap between that feature and an implanted signal (Fig. 2i, Supplementary Fig. 6).

The bash script for generating the OLGA sequences, as well as the YAML specification files describing the simulation of immune events and benchmarking of ML methods, are available in the immuneML documentation under use case examples: docs.immuneml.uio.no/latest/usecases/benchmarking_use_case.html. The benchmarking dataset with simulated immune events as well as the complete collection of figures (for all cross-validation splits, immune events, ML methods and encodings) can be downloaded from the NIRD research data archive[82].

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

Data availability
All data for the analyses presented in the manuscript are openly available. The detailed result files for use cases 1–3 are available as zip files at https://doi.org/10.11582/2021.00008 (ref. [78]; use case 1), https://doi.org/10.11582/2021.00009 (ref. [81]; use case 2) and https://doi.org/10.11582/2021.00005 (ref. [82]; use case 3). Input data for use case 1 was downloaded from https://doi.org/10.21417/B7001Z.

Code availability
The immuneML source code is openly available at Github (github.com/uio-bmi/immuneML) under a free software license (AGPL-3.0). immuneML version 2.0.2 has been deposited on Zenodo with https://doi.org/10.5281/zenodo.5118741 (ref. [75]). The immuneML Python package can be downloaded from pypi.org/project/immuneML.

Received: 7 April 2021; Accepted: 6 October 2021;
Published online: 16 November 2021

References
1. Brown, A. J. et al. Augmenting adaptive immunity: progress and challenges in the quantitative engineering and analysis of adaptive immune receptor repertoires. *Mol. Syst. Des. Eng.* **4**, 701–736 (2019).
2. Georgiou, G. et al. The promise and challenge of high-throughput sequencing of the antibody repertoire. *Nat. Biotechnol.* **32**, 158–168 (2014).
3. Yaari, G. & Kleinstein, S. H. Practical guidelines for B-cell receptor repertoire sequencing analysis. *Genome Med.* **7**, 121 (2015).
4. Csepregi, L., Ehling, R. A., Wagner, B. & Reddy, S. T. Immune literacy: reading, writing, and editing adaptive immunity. *iScience* **23**, 101519 (2020).
5. DeWitt, W. S. III et al. Human T cell receptor occurrence patterns encode immune history, genetic background, and receptor specificity. *eLife* **7**, e38358 (2018).
6. Emerson, R. O. et al. Immunosequencing identifies signatures of cytomegalovirus exposure history and HLA-mediated effects on the T cell repertoire. *Nat. Genet.* **49**, 659–665 (2017).
7. Krishna, C., Chowell, D., Gönen, M., Elhanati, Y. & Chan, T. A. Genetic and environmental determinants of human TCR repertoire diversity. *Immun. Ageing* **17**, 26 (2020).
8. Britanova, O. V. et al. Age-related decrease in TCR repertoire diversity measured with deep and normalized sequence profiling. *J. Immunol.* **192**, 2689–2698 (2014).
9. Schneider-Hohendorf, T. et al. Sex bias in MHC I-associated shaping of the adaptive immune system. *Proc. Natl Acad. Sci. USA* **115**, 2168–2173 (2018).
10. Shemesh, O., Polak, P., Lundin, K. E. A., Sollid, L. M. & Yaari, G. Machine learning analysis of naïve B-cell receptor repertoires stratifies celiac disease patients and controls. *Front. Immunol.* **12**, https://doi.org/10.3389/fimmu.2021.627813 (2021).
11. Ostmeyer, J., Christley, S., Toby, I. T. & Cowell, L. G. Biophysicochemical motifs in T cell receptor sequences distinguish repertoires from tumor-infiltrating lymphocytes and adjacent healthy tissue. *Cancer Res.* https://doi.org/10.1158/0008-5472.CAN-18-2292 (2019).
12. Beshnova, D. et al. De novo prediction of cancer-associated T cell receptors for noninvasive cancer detection. *Sci. Transl. Med.* **12**, eaaz3738 (2020).
13. Liu, X. et al. T cell receptor β repertoires as novel diagnostic markers for systemic lupus erythematosus and rheumatoid arthritis. *Ann. Rheum. Dis.* **78**, 1070–1078 (2019).
14. Arnaout, R. A. et al. The future of blood testing is the immunome. *Front. Immunol.* **12**, 626793 (2021).
15. Greiff, V., Yaari, G. & Cowell, L. Mining adaptive immune receptor repertoires for biological and clinical information using machine learning. *Curr. Opin. Syst. Biol.* https://doi.org/10.1016/j.coisb.2020.10.010 (2020).
16. Akbar, R. et al. A compact vocabulary of paratope-epitope interactions enables predictability of antibody-antigen binding. *Cell Rep.* **34**, 108856 (2021).
17. Dash, P. et al. Quantifiable predictive features define epitope-specific T cell receptor repertoires. *Nature* **547**, 89–93 (2017).
18. Glanville, J. et al. Identifying specificity groups in the T cell receptor repertoire. *Nature* **547**, 94–98 (2017).
19. 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).
20. Friedensohn, S. et al. Convergent selection in antibody repertoires is revealed by deep learning. Preprint at *bioRxiv* https://doi.org/10.1101/2020.02.25.965673 (2020).
21. Mason, D. M. et al. Optimization of therapeutic antibodies by predicting antigen specificity from antibody sequence via deep learning. *Nat. Biomed. Eng.* **5**, 600–612 (2021).
22. Moris, P. et al. Current challenges for unseen-epitope TCR interaction prediction and a new perspective derived from image classification. *Brief. Bioinform.* https://doi.org/10.1093/bib/bbaa318 (2020).
23. Graves, J. et al. A review of deep learning methods for antibodies. *Antibodies* **9**, 12 (2020).
24. Narayanan, H. et al. Machine learning for biologics: opportunities for protein engineering, developability, and formulation. *Trends Pharmacol. Sci.* **42**, 151–165 (2021).
25. 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).
26. Laustsen, A. H., Greiff, V., Karatt-Vellatt, A., Muyldermans, S. & Jenkins, T. P. Animal immunization, in vitro display technologies, and machine learning for antibody discovery. *Trends Biotechnol.* https://doi.org/10.1016/j.tibtech.2021.03.003 (2021).
27. Jokinen, E., Huuhtanen, J., Mustjoki, S., Heinonen, M. & Lähdesmäki, H. Predicting recognition between T cell receptors and epitopes with TCRGP. *PLoS Comput. Biol.* **17**, e1008814 (2021).

28. Eraslan, G., Avsec, Ž., Gagneur, J. & Theis, F. J. Deep learning: new computational modelling techniques for genomics. *Nat. Rev. Genet.* **20**, 389–403 (2019).
29. Esteva, A. et al. A guide to deep learning in healthcare. *Nat. Med.* **25**, 24–29 (2019).
30. Vamathevan, J. et al. Applications of machine learning in drug discovery and development. *Nat. Rev. Drug Discov*. https://doi.org/10.1038/s41573-019-0024-5 (2019).
31. Wainberg, M., Merico, D., Delong, A. & Frey, B. J. Deep learning in biomedicine. *Nat. Biotechnol.* **36**, 829–838 (2018).
32. Lythe, G., Callard, R. E., Hoare, R. L. & Molina-París, C. How many TCR clonotypes does a body maintain? *J. Theor. Biol.* **389**, 214–224 (2016).
33. Mora, T. & Walczak, A. M. How many different clonotypes do immune repertoires contain? *Curr. Opin. Syst. Biol.* **18**, 104–110 (2019).
34. Briney, B., Inderbitzin, A., Joyce, C. & Burton, D. R. Commonality despite exceptional diversity in the baseline human antibody repertoire. *Nature* **566**, 393–397 (2019).
35. Greiff, V. et al. Learning the high-dimensional immunogenomic features that predict public and private antibody repertoires. *J. Immunol.* https://doi.org/10.4049/jimmunol.1700594 (2017).
36. Parameswaran, P. et al. Convergent antibody signatures in human dengue. *Cell Host Microbe* **13**, 691–700 (2013).
37. Thomas, N. et al. Tracking global changes induced in the CD4 T-cell receptor repertoire by immunization with a complex antigen using short stretches of CDR3 protein sequence. *Bioinformatics* **30**, 3181–3188 (2014).
38. Christophersen, A. et al. Tetramer-visualized gluten-specific CD4$^+$ T cells in blood as a potential diagnostic marker for coeliac disease without oral gluten challenge. *United Eur. Gastroenterol. J.* **2**, 268–278 (2014).
39. Widrich, M. et al. Modern Hopfield networks and attention for immune repertoire classification. *Adv. Neural Inf. Process. Syst.* **33**, 18832–18845 (2020).
40. Sidhom, J.-W., Larman, H. B., Pardoll, D. M. & Baras, A. S. DeepTCR is a deep learning framework for revealing sequence concepts within T-cell repertoires. *Nat. Commun.* **12**, 1605 (2021).
41. Chen, K. M., Cofer, E. M., Zhou, J. & Troyanskaya, O. G. Selene: a PyTorch-based deep learning library for sequence data. *Nat. Methods* **16**, 315–318 (2019).
42. Kopp, W., Monti, R., Tamburrini, A., Ohler, U. & Akalin, A. Deep learning for genomics using Janggu. *Nat. Commun.* **11**, 3488 (2020).
43. Feng, J. et al. Firmiana: towards a one-stop proteomic cloud platform for data processing and analysis. *Nat. Biotechnol.* **35**, 409–412 (2017).
44. Gessulat, S. et al. Prosit: proteome-wide prediction of peptide tandem mass spectra by deep learning. *Nat. Methods* **16**, 509–518 (2019).
45. Tomic, A. et al. SIMON: Open-source knowledge discovery platform. *Patterns* **2**, 100178 (2021).
46. Wu, Z. et al. MoleculeNet: a benchmark for molecular machine learning. *Chem. Sci.* **9**, 513–530 (2018).
47. Pedregosa, F. et al. Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011).
48. Paszke, A. et al. in *Advances in Neural Information Processing Systems 32* (eds Wallach, H. et al.) 8026–8037 (Curran Associates, Inc., 2019).
49. Afgan, E. et al. The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update. *Nucleic Acids Res.* **46**, W537–W544 (2018).
50. Rubelt, F. et al. Adaptive immune receptor repertoire community recommendations for sharing immune-repertoire sequencing data. *Nat. Immunol.* **18**, 1274–1278 (2017).
51. Vander Heiden, J. A. et al. AIRR community standardized representations for annotated immune repertoires. *Front. Immunol.* **9**, 2206 (2018).
52. Bolotin, D. A. et al. MiXCR: software for comprehensive adaptive immunity profiling. *Nat. Methods* **12**, 380–381 (2015).
53. Gupta, N. T. et al. Change-O: a toolkit for analyzing large-scale B cell immunoglobulin repertoire sequencing data. *Bioinformatics* **31**, 3356–3358 (2015).
54. Vander Heiden, J. A. et al. pRESTO: a toolkit for processing high-throughput sequencing raw reads of lymphocyte receptor repertoires. *Bioinformatics* **30**, 1930–1932 (2014).
55. Nazarov, V., immunarch.bot & Rumynskiy, E. immunomind/immunarch: 0.6.5: basic single-cell support. *Zenodo* https://doi.org/10.5281/zenodo.3893991 (2020).
56. Christley, S. et al. The ADC API: a web API for the programmatic query of the AIRR data commons. *Front. Big Data* **3**, 22 (2020).
57. Corrie, B. D. et al. iReceptor: a platform for querying and analyzing antibody/B-cell and T-cell receptor repertoire data across federated repositories. *Immunol. Rev.* **284**, 24–41 (2018).
58. 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).
59. Huang, H., Wang, C., Rubelt, F., Scriba, T. J. & Davis, M. M. Analyzing the *Mycobacterium tuberculosis* immune response by T-cell receptor clustering with GLIPH2 and genome-wide antigen screening. *Nat. Biotechnol.* https://doi.org/10.1038/s41587-020-0505-4 (2020).
60. Zheng, G. X. Y. et al. Massively parallel digital transcriptional profiling of single cells. *Nat. Commun.* **8**, 14049 (2017).
61. Nolan, S. et al. A large-scale database of T-cell receptor beta (TCRβ) sequences and binding associations from natural and synthetic exposure to SARS-CoV-2. Preprint at *Research Square* https://doi.org/10.21203/rs.3.rs-51964/v1 (2020).
62. Weber, C. R. et al. immuneSIM: tunable multi-feature simulation of B- and T-cell receptor repertoires for immunoinformatics benchmarking. *Bioinformatics* **36**, 3594–3596 (2020).
63. Marcou, Q., Mora, T. & Walczak, A.M. High-throughput immune repertoire analysis with IGoR. Nat Commun 9, 561 (2018). https://doi.org/10.1038/s41467-018-02832-w
64. Sethna, Z., Elhanati, Y., Callan, C. G., Walczak, A. M. & Mora, T. OLGA: fast computation of generation probabilities of B- and T-cell receptor amino acid sequences and motifs. *Bioinformatics* **35**, 2974–2981 (2019).
65. FAIR principles for data stewardship. *Nat. Genet.* **48**, 343–343 (2016).
66. Scott, J. K. & Breden, F. The adaptive immune receptor repertoire community as a model for FAIR stewardship of big immunology data. *Curr. Opin. Syst. Biol.* **24**, 71–77 (2020).
67. Breden, F. et al. Reproducibility and reuse of adaptive immune receptor repertoire data. *Front. Immunol.* **8**, 1418 (2017).
68. Software with impact. *Nat. Methods* **11**, 211 (2014).
69. Goodman, S. N., Fanelli, D. & Ioannidis, J. P. A. What does research reproducibility mean? *Sci. Transl. Med.* **8**, 341ps12 (2016).
70. Mayer-Blackwell, K. et al. TCR meta-clonotypes for biomarker discovery with tcrdist3: quantification of public, HLA-restricted TCR biomarkers of SARS-CoV-2 infection. Preprint at *bioRxiv* https://doi.org/10.1101/2020.12.24.424260 (2020).
71. Abadi, M. et al. TensorFlow: a system for large-scale machine learning. In *Proc. 12th USENIX Conference on Operating Systems Design and Implementation* 265–283 (USENIX Association, 2016).
72. Vujovic, M. et al. T cell receptor sequence clustering and antigen specificity. *Comput. Struct. Biotechnol. J.* **18**, 2166–2173 (2020).
73. Davidsen, K. et al. Deep generative models for T cell receptor protein sequences. *eLife* **8**, e46935 (2019).
74. Bareinboim, E. & Pearl, J. Causal inference and the data-fusion problem. *Proc. Natl Acad. Sci. USA* **113**, 7345–7352 (2016).
75. Pavlovic, M. et al. immuneML: v2.0.2. *Zenodo* https://doi.org/10.5281/zenodo.5118741 (2021)
76. Fowler, M. *Domain-Specific Languages* (Addison-Wesley Professional, 2010).
77. Zenger, M. *Programming Language Abstractions for Extensible Software Components* Ch. 1.3 (Swiss Federal Institute of Technology, 2004).
78. Pavlović, M. immuneML use case 1: replication of a published study inside immuneML. *NIRD Research Data Archive* https://doi.org/10.11582/2021.00008 (2021).
79. Ploenzke, M. S. & Irizarry, R. A. Interpretable convolution methods for learning genomic sequence motifs. Preprint at *bioRxiv* https://doi.org/10.1101/411934 (2018).
80. Heikkilä, N. et al. Human thymic T cell repertoire is imprinted with strong convergence to shared sequences. *Mol. Immunol.* **127**, 112–123 (2020).
81. Pavlović, M. immuneML use case 2: extending immuneML with a deep learning component for predicting antigen specificity of paired receptor data. *NIRD Research Data Archive* https://doi.org/10.11582/2021.00009 (2021).
82. Scheffer, L. immuneML use case 3: benchmarking ML methods for AIRR classification on ground-truth synthetic data. *NIRD Research Data Archive* https://doi.org/10.11582/2021.00005 (2021).

Acknowledgements

We acknowledge generous support by The Leona M. and Harry B. Helmsley Charitable Trust (grant number 2019PG-T1D011, to V.G. and T.M.B.), the UiO World-Leading Research Community (to V.G. and L.M.S.), the UiO:LifeScience Convergence Environment Immunolingo (to V.G. and G.K.S.), EU Horizon 2020 iReceptorplus (grant number 825821, to V.G. and L.M.S.), a Research Council of Norway FRIPRO project (grant number 300740, to V.G.), a Research Council of Norway IKTPLUSS project (grant number 311341, to V.G. and G.K.S.), the National Institutes of Health (grant numbers P01 AI042288 and HIRN UG3 DK122638 to T.M.B.) and Stiftelsen Kristian Gerhard Jebsen (K.G. Jebsen Coeliac Disease Research Centre, to L.M.S. and G.K.S.). We acknowledge support from ELIXIR Norway in recognizing immuneML as a national node service.

Author contributions

M.P., V.G. and G.K.S. conceived the study. M.P. and G.K.S. designed the overall software architecture. M.P., L.S. and K.M. developed the main platform code. M.P. and L.S.

performed all analyses. M.P., L.S., C.K., F.L.M.B., R.A., G.S.A.H., G.B., M.C., R.F., I.G., S.G., P.-H.H., K.R., E.R., P.A.R., A.S., D.T., C.R.W. and M.W. created software or documentation content. R.K., N.V., K.W., L.S., M.P., A.A.C. and B.C. designed and developed the Galaxy tools. C.K., R.A., T.M.B., M.C., S.C., L.G.C., I.H.H., E.H., G.K., M.L.K., C.L.-A., A.M., T.M., J.P., K.R., P.A.R., A.R., I.S., L.M.S. and G.Y. provided critical feedback. M.P., L.S., V.G. and G.K.S. drafted the manuscript. V.G. and G.K.S. supervised the project. All authors read and approved the final manuscript and are personally accountable for its content.

Competing interests
V.G. declares advisory board positions in aiNET GmbH and Enpicom B.V., and is a consultant for Roche/Genentech.

Additional information
Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s42256-021-00413-z.

Correspondence and requests for materials should be addressed to Geir Kjetil Sandve.

Peer review information *Nature Machine Intelligence* thanks Pieter Meysman, Ryan Emerson and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

Reprints and permissions information is available at www.nature.com/reprints.

nature portfolio

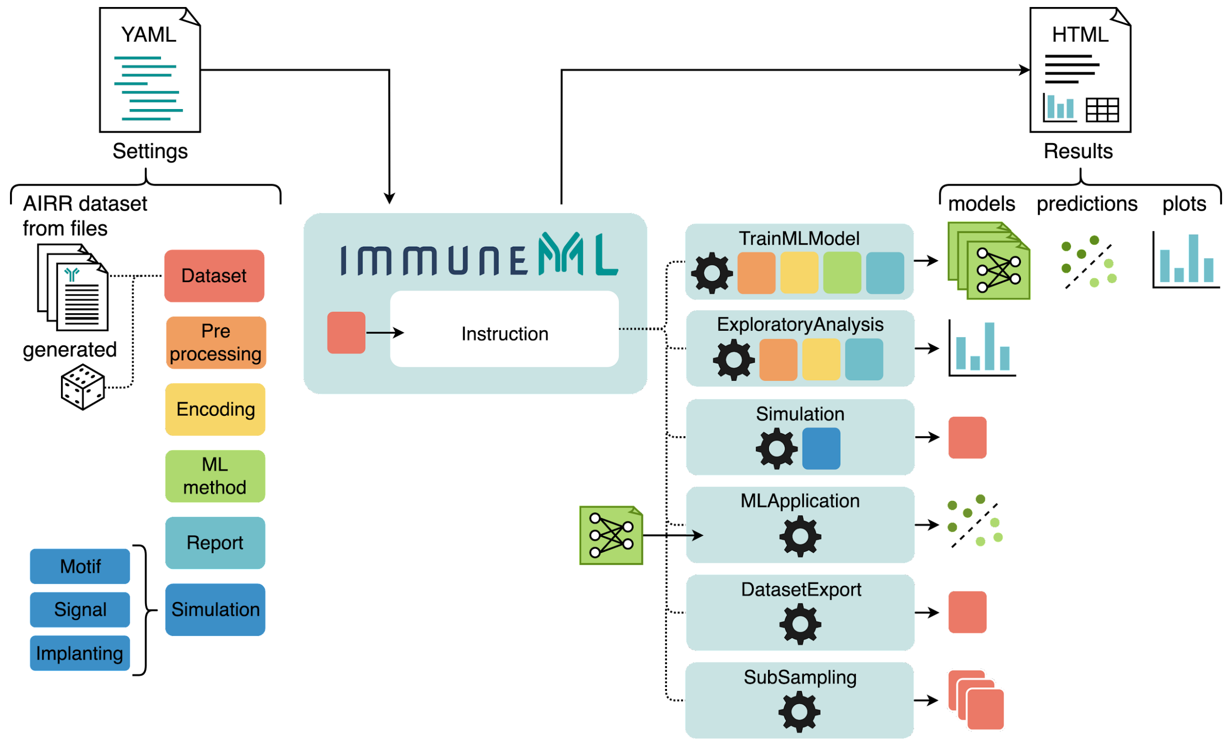
https://doi.org/10.1038/s42256-021-00413-z

Supplementary information

The immuneML ecosystem for machine learning analysis of adaptive immune receptor repertoires

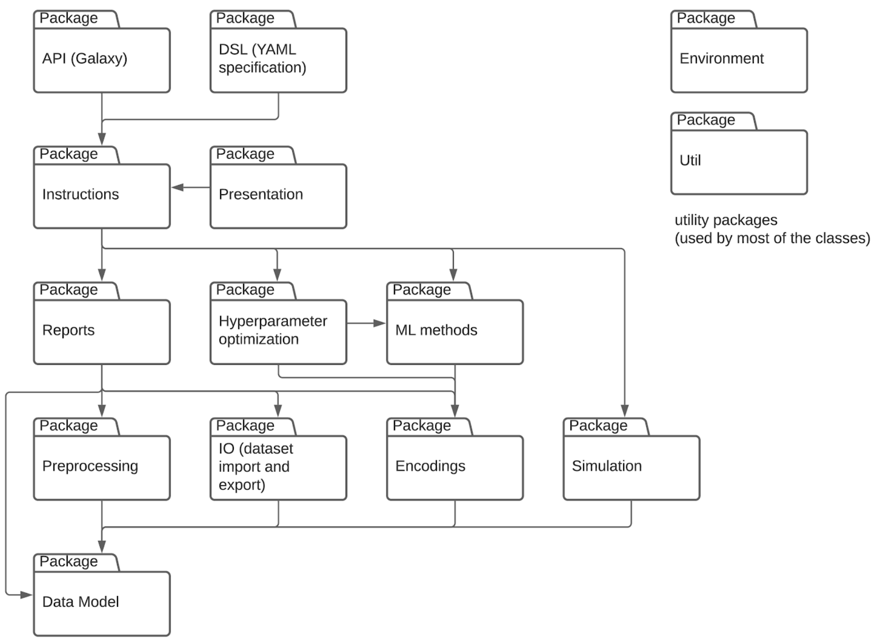
In the format provided by the authors and unedited

Supplementary Figures

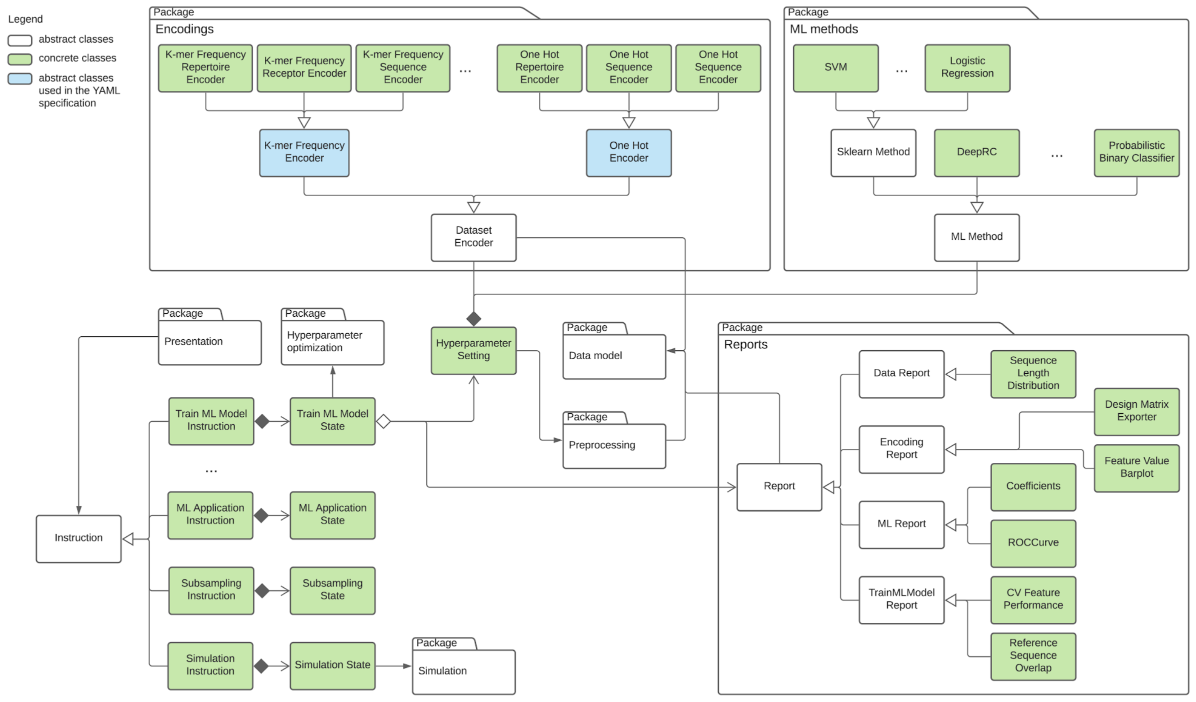


Supplementary Figure 1 | Overview of how immuneML analyses are specified. The YAML specification file describes the analysis components, and what instructions should be performed using these components. The analysis components are datasets, preprocessing, encoding, ML methods, analysis reports, and simulation-related components. Supplementary Table 1 contains a complete list of all components that can be specified. Instructions include training and applying ML models, exploratory analysis, and simulation of synthetic datasets. The results produced by the instructions can be navigated through an HTML summary page generated by immuneML.

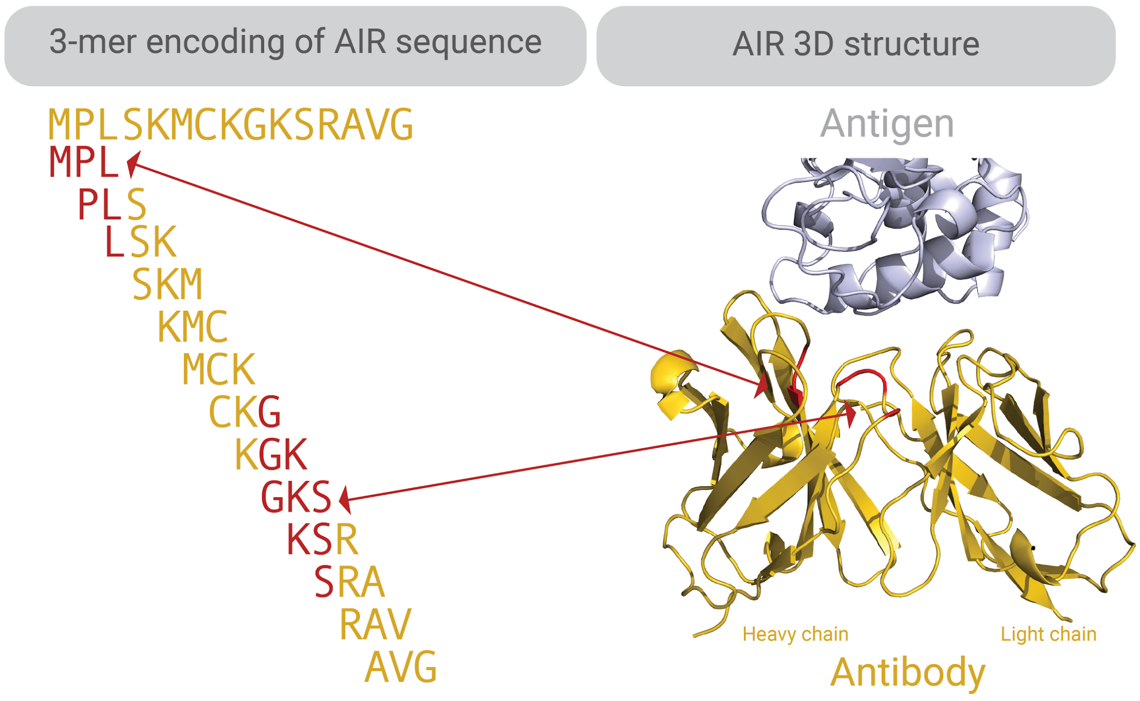
A immuneML architecture: package overview with dependencies between packages



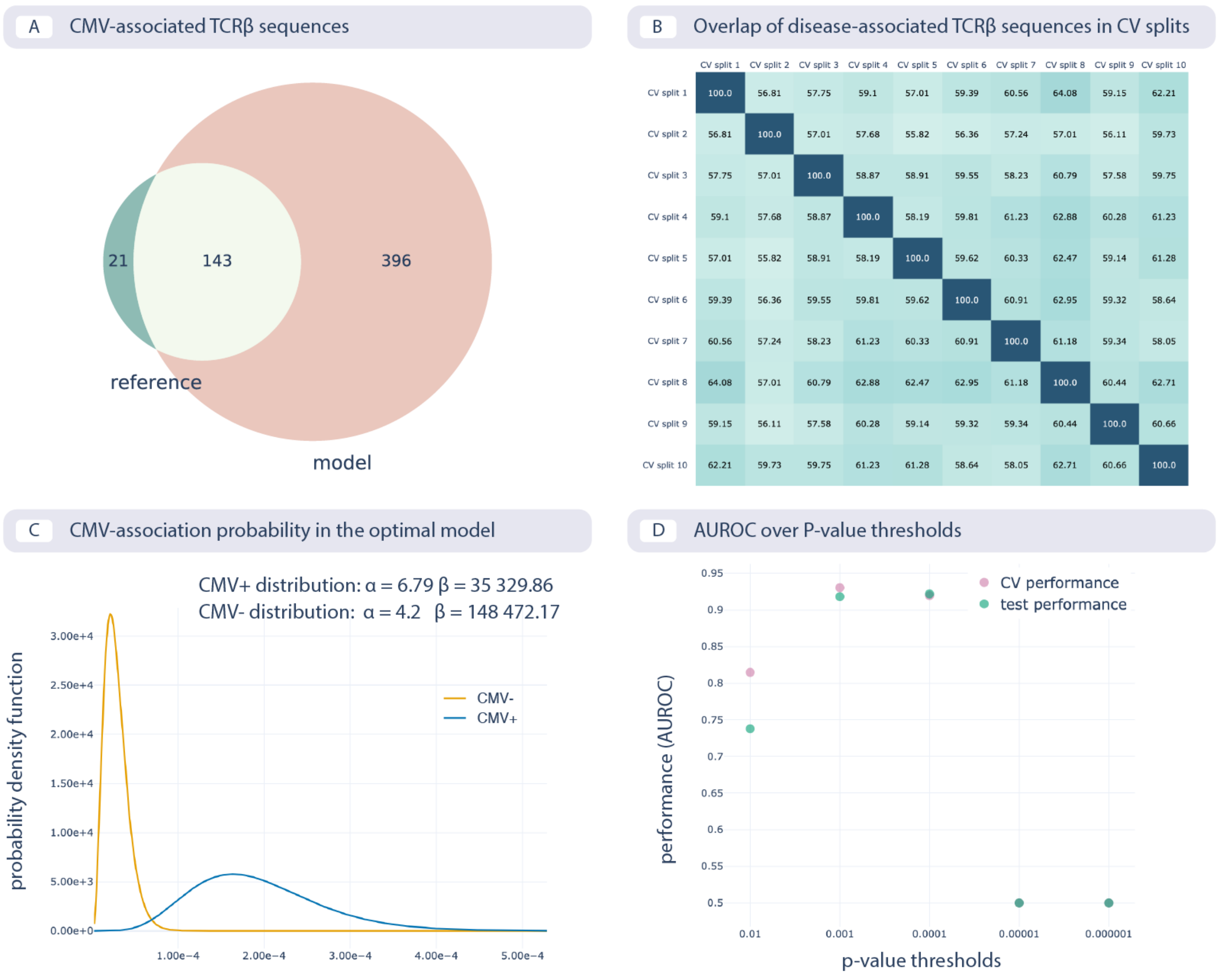
B Extending the immuneML platform with new encodings, ML methods and analysis reports



Supplementary Figure 2 | immuneML architecture overview as UML (Unified Modeling Language) diagrams. (a) High-level overview of the immuneML architecture showing the most important packages and their dependencies. The application programming interface (API) and domain-specific language (DSL) packages represent how the user can interact with immuneML, either through the Galaxy web interface (API package) or when constructing YAML specification files (DSL package). These packages invoke instructions, which map to different analyses that can be performed with immuneML, such as training an ML model or simulating an AIRR dataset. In turn, instructions depend on specific components to perform the analysis. **(b)** To extend the platform with new encodings, ML methods, or reports, users may investigate the corresponding package and implement the functionalities as described by the appropriate abstract class. The added components could then be used in different instructions according to their purpose. Developer tutorials are available at docs.immuneml.uio.no/latest/developer_docs.html.

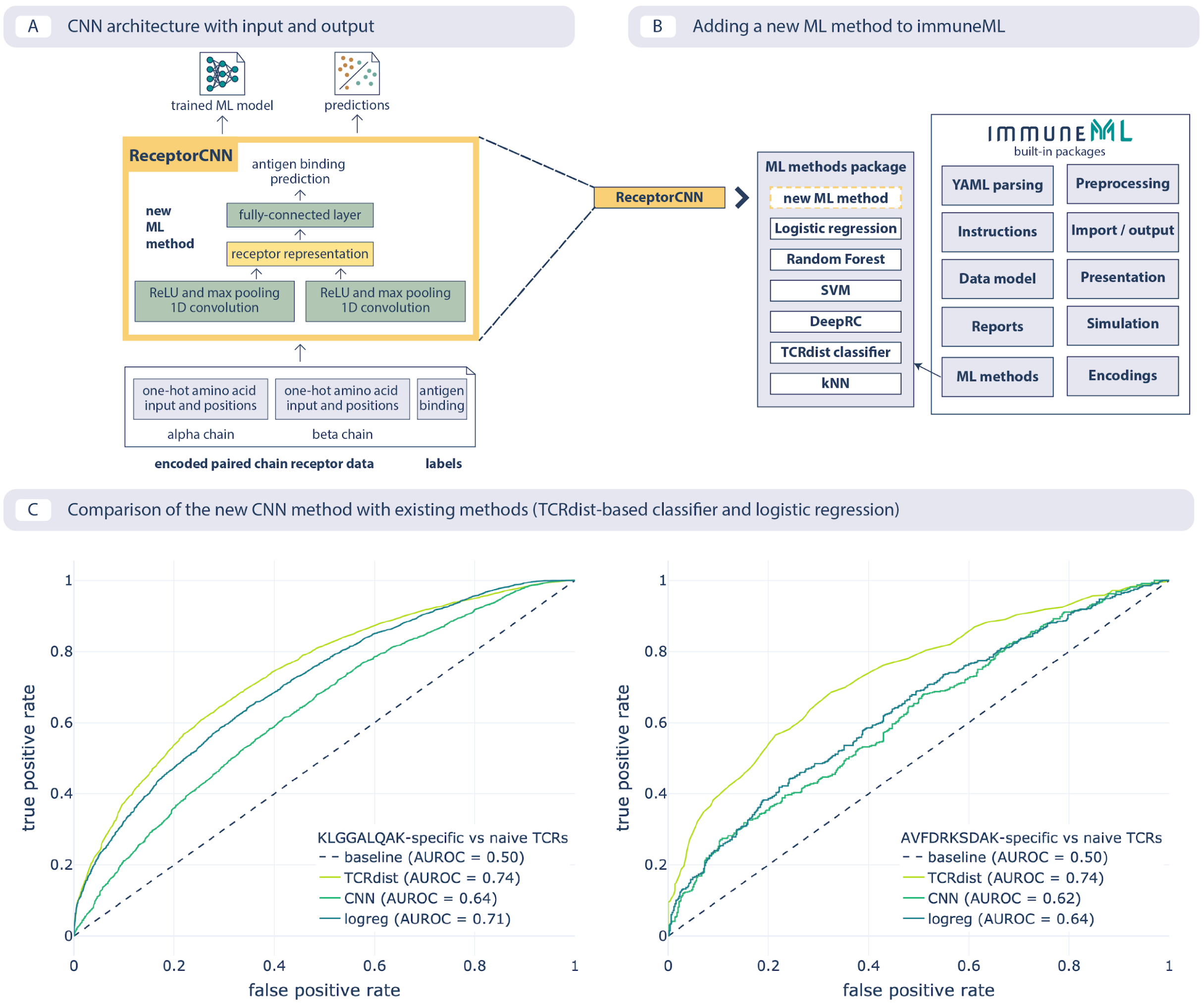


Supplementary Figure 3 | An example of how a k-mer encoded AIR sequence may map the antibody 3D-structure. K-mers are subsequences of length *k*. Through ML, we can learn, for example, which k-mers are important for determining antigen specificity (color-coded in red). These k-mers may map to regions of the (CDR3) sequence that are in contact with the antigen.[16] Protein Data Bank (PDB) ID of the 3D structure: 2DQC[83]. 3D visualization of the antibody-antigen structure was carried out in Pymol[84].

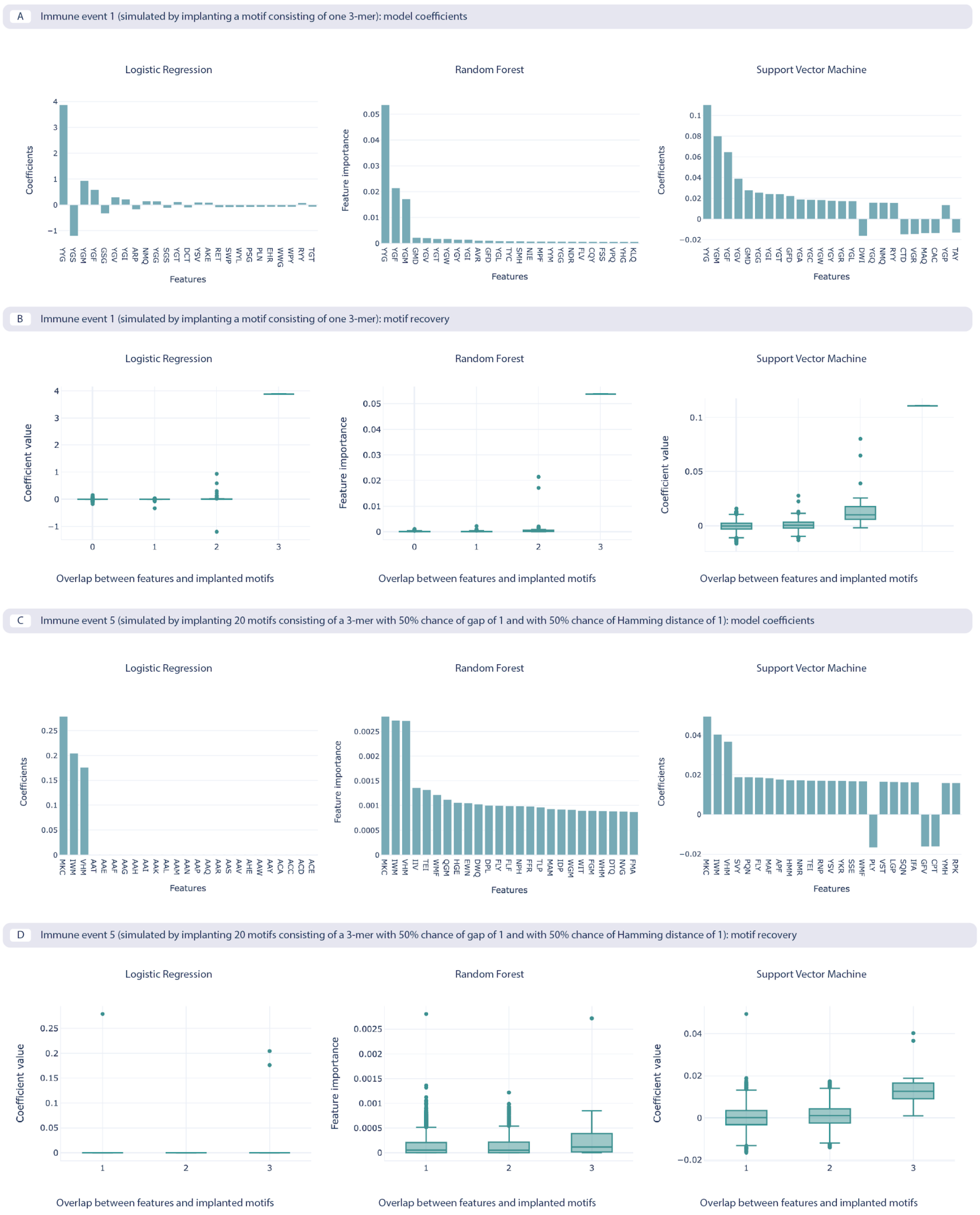


	CV split 1	CV split 2	CV split 3	CV split 4	CV split 5	CV split 6	CV split 7	CV split 8	CV split 9	CV split 10
CV split 1	100.0	56.81	57.75	59.1	57.01	59.39	60.56	64.08	59.15	62.21
CV split 2	56.81	100.0	57.01	57.68	55.82	56.36	57.24	57.01	56.11	59.73
CV split 3	57.75	57.01	100.0	58.87	58.91	59.55	58.23	60.79	57.58	59.75
CV split 4	59.1	57.68	58.87	100.0	58.19	59.81	61.23	62.88	60.28	61.23
CV split 5	57.01	55.82	58.91	58.19	100.0	59.62	60.33	62.47	59.14	61.28
CV split 6	59.39	56.36	59.55	59.81	59.62	100.0	60.91	62.95	59.32	58.64
CV split 7	60.56	57.24	58.23	61.23	60.33	60.91	100.0	61.18	59.34	58.05
CV split 8	64.08	57.01	60.79	62.88	62.47	62.95	61.18	100.0	60.44	62.71
CV split 9	59.15	56.11	57.58	60.28	59.14	59.32	59.34	60.44	100.0	60.66
CV split 10	62.21	59.73	59.75	61.23	61.28	58.64	58.05	62.71	60.66	100.0

Supplementary Figure 4 | Reproducing the CMV status prediction study by Emerson et al.[6] **(a)** The overlap of the 164 disease-associated TCRβ sequences (V-TCRβaa-J) determined in the original study by Emerson et al., labeled "reference", with those determined by the optimal model as reproduced here with a p-value threshold of 0.001 (labeled "model"). **(b)** The overlap percentage of disease-associated TCRβ sequences for the optimal model with the p-value threshold of 0.001 between different data splits in 10-fold cross-validation (between 50% and 65% overlap). **(c)** The probability that a TCRβ sequence is CMV-associated follows a beta distribution estimated separately for CMV positive and negative subjects, which is then used for CMV status prediction of new subjects. **(d)** Area under the ROC curve (AUROC) over p-value thresholds in training data (average AUROC over 10 crossvalidation splits) and test data (AUROC in cohort 2).



Supplementary Figure 5 | Extending immuneML with a new ML method (a) The architecture of the new convolutional neural network (CNN) is shown together with expected input in the format of encoded paired-chain receptors and predictions and trained model as output. **(b)** Adding a new method requires implementing an interface for the method and then it can reuse the infrastructure (data model, encodings, nested cross-validation, visualizations) without any additional changes. **(c)** An example usage where the method can be readily compared with other methods already available within the platform. Here, the area under the ROC curve (AUROC) is shown on two datasets: CMV-specific (epitope: KLGGALQAK, left), and EBV-specific (epitope: AVFDRKSDAK, right), for the CNN that was added (cnn), TCRdist-based k-nearest neighbors classifier (tcrdist) and logistic regression on 3-mer frequencies (logreg).



Supplementary Figure 6 | The model coefficients and motif recovery reports for the benchmarking use case. Repertoire datasets are represented by 3-mer amino acid frequencies. Two immune events are shown: immune event 1 (a, b) is the simplest event simulated by implanting a single 3-mer, while the immune event 5 (c, d) is the most complex one simulated by implanting 20 motifs consisting of a 3-mer with a 50% chance of having a gap and 50% chance of having a Hamming distance of 1. **(a)** The 25 largest coefficients of the logistic regression model, feature importances on random forest model, and coefficients of the support vector machine (SVM) model with a linear kernel. The highest value of the coefficients corresponds to the implanted motif. **(b)** Coefficient values for the features depending on the overlap between the recovered features that overlap with the implanted motif, measuring how well the recovered motifs correspond to the implanted motif, shown across the three ML models. **(c)** The 25 largest coefficients and feature importances for the ML models trained on immune event 5. **(d)** Overlap of recovered and implanted motifs for the ML models trained on immune event 5. Motif recovery for immune event 5 is less effective than for immune event 1.

Supplementary Tables

Supplementary Table 1 | Components for data import, encoding, ML methods, and reports included in immuneML.

Component type	Name	Description
Data import	AIRR	Imports data in AIRR format.
Data import	Generic	Fully configurable import class which can import data from any tabular file.
Data import	IGoR	Imports CDR3 sequences exported by IGoR[63] when running IGoR with the –CDR3 option specified
Data import	IReceptor	Imports datasets from the AIRR Data Commons[56] retrieved through the iReceptor Gateway[57]. This importer automatically parses repertoire metadata as exported by the iReceptor Gateway.
Data import	ImmunoSEQRearrangement	Imports data from Adaptive Biotechnologies immunoSEQ Analyzer[85] rearrangement-level .tsv files.
Data import	ImmunoSEQSample	Imports data from Adaptive Biotechnologies immunoSEQ Analyzer[85] sample-level .tsv files.
Data import	MiXCR	Imports data exported by MiXCR[52].
Data import	OLGA	Imports data generated by OLGA[64].
Data import	Pickle	Imports data from pickle files previously exported by immuneML.
Data import	RandomReceptorDataset	Automatically generate a paired receptor dataset consisting of random amino acids during runtime. This can be used for testing or benchmarking.
Data import	RandomRepertoireDataset	Automatically generate a repertoire dataset consisting of random amino acids during runtime. This can be used for testing or benchmarking.
Data import	RandomSequenceDataset	Automatically generate a sequence dataset consisting of random amino acids during runtime. This can be used for testing or benchmarking.
Data import	TenxGenomics	Imports files produced by 10x Genomics Cell Ranger analysis pipeline[60] named ‘Clonotype consensus annotations (CSV)’.
Data import	VDJdb	Imports data from the VDJdb[58]
Preprocessing	ChainRepertoireFilter	Removes repertoires from the dataset containing sequences with illegal chain types, based on user-specified chain types (e.g., TRB, IGH).

Preprocessing	ClonesPerRepertoireFilter	Removes all repertoires from the dataset which contain too many or too few clonotypes based on user-specified thresholds.
Preprocessing	CountPerSequenceFilter	Removes all sequences from a repertoire which have a sequence count value below the user-specified threshold, or missing count values.
Preprocessing	DuplicateSequenceFilter	Collapses all sequences in a repertoire which have the same amino acid or nucleotide sequence and V and J genes.
Preprocessing	MetadataRepertoireFilter	Removes all repertoires from the dataset based on their specified metadata (e.g., remove repertoires when age is greater than a given threshold).
Preprocessing	SubjectRepertoireCollector	Merges all repertoires in the dataset that have the same subject identifier.
Encoding	AtchleyKmer	Represents a repertoire through Atchley factors and relative abundance of k-mers, as done by Ostmeyer and colleagues [11].
Encoding	Distance	Encodes a given repertoire dataset as a distance matrix, where the pairwise distance between each of the repertoires is calculated using a user-configurable distance metric.
Encoding	DeepRC	Prepare a repertoire dataset to be used with the DeepRC classifier[39].
Encoding	EvennessProfile	Encodes a repertoire as its evenness profile described by Greiff and colleagues[86].
Encoding	KmerFrequency	Encodes repertoires, sequences, or paired receptors by frequencies of k-mers it contains. A k-mer is a sequence of letters of length k into which an immune receptor sequence can be decomposed.
Encoding	MatchedReceptors	Takes a collection of paired reference receptors and counts how often they occur in a repertoire dataset.
Encoding	MatchedRegex	Takes a collection of regular expressions and counts how often they occur in a repertoire dataset.
Encoding	MatchedSequences	Takes a collection of reference sequences and counts how often they occur in a repertoire dataset.
Encoding	OneHot	One-hot encoding for repertoires, sequences or paired receptors. Positional information of the characters in the sequences may be included.
Encoding	SequenceAbundance	Represents the repertoires as vectors where the first element corresponds to the number of label-associated clonotypes, and the second element is the total number of unique clonotypes, as done by Emerson and colleagues[6].

Encoding	TCRdist	Encodes a paired receptor dataset as a distance matrix between all receptors, where the distance is computed using tcrdist3[70].
Encoding	Word2Vec	Encodes a repertoire dataset based on the vector representations of k-mers in the sequences in a repertoire from the context the k-mers appear in, using the Word2Vec implementation from the gensim package[87].
ML method	AtchleyKmerMILClassifier	A multiple instance learning classifier using AtchleyKmer encoding, replicating Ostmeyer et al.[11].
ML method	DeepRC	Internally uses the DeepRC method for repertoire classification[39].
ML method	KNN	A wrapper for the scikit-learn KNeighborsClassifier class[47].
ML method	LogisticRegression	A wrapper for the scikit-learn LogisticRegression class[47].
ML method	ProbabilisticBinaryClassifier	A classifier that predicts the class based on examples encoded by the number of successful trials and the total number of trials (SequenceAbundance encoding). It models this ratio by one beta distribution per class and predicts the class of the new examples using log-posterior odds ratio with the threshold at 0, replicating Emerson et al[6].
ML method	RandomForestClassifier	A wrapper for the scikit-learn RandomForestClassifier class[47].
ML method	ReceptorCNN	A CNN that detects motifs using CNN kernels in each chain of one-hot encoded paired receptors, combines the kernel activations into a unique representation of the receptor and uses this representation to predict the antigen-binding.
ML method	SVM	A wrapper for scikit-learn SVC[47].
ML method	TCRdistClassifier	Implementation of a nearest neighbors classifier based on TCR distances as presented in [17]. This method is implemented using the scikit-learn KNeighborsClassifier with k determined at runtime from the training dataset size and weights linearly scaled to decrease with the distance of examples
Data report	CytoscapeNetworkExporter	Exports paired receptors to .sif format such that they can directly be imported as a network in Cytoscape[88], to visualize chain sharing between the different receptors in a dataset.
Data report	GLIPH2Exporter	Exports paired receptor data to GLIPH2 format so that it can be directly used in GLIPH2[59].

Data report	ReceptorDatasetOverview	Plots a histogram of the lengths of AIRs per chain in a Receptor Dataset.
Data report	SequenceLengthDistribution	Plots a histogram of the lengths of the AIR sequences in an AIRR dataset.
Encoding report	DesignMatrixExporter	Exports the design matrix and related information of an encoded dataset to csv files.
Encoding report	FeatureDistribution	Plots the distribution of feature values for an encoded dataset. Distributions may be grouped based on metadata labels.
Encoding report	FeatureValueBarplot	Plots a barplot of feature values for an encoded dataset. Bars may be grouped based on metadata labels.
Encoding report	Matches	Reports the number of matches found when encoding the data using MatchedSequences, MatchedReceptors or MatchedRegex.
Encoding report	RelevantSequenceExporter	Exports the sequences that are extracted as label-associated by the SequenceAbundance encoder.
ML model report	Coefficients	Plot the coefficients or feature importance for a given ML model.
ML model report	ConfounderAnalysis	Plots the number of false positive and false negative predictions made for the examples (repertoires or receptors), grouped by the metadata.
ML model report	DeepRCMotifDiscovery	Plots the contributions of input sequences and kernels to trained DeepRC model[39] with respect to the test dataset. Contributions are computed using integrated gradients.
ML model report	KernelSequenceLogo	Plots the kernels of the ReceptorCNN model as sequence logos, as well as the weights in the final fully-connected layer of the network associated with kernel outputs.
ML model report	MotifSeedRecovery	Shows how well implanted motifs (simulated immune events) are recovered by ML methods using KmerFrequency encoding. This report plots the number of amino acids that overlap between the feature and implanted motif against the coefficient size of the respective feature.
ML model report	ROCCurve	Plot the ROC curve of a trained ML model.
ML model report	SequenceAssociationLikelihood	Plots the beta distribution used as a prior for a class assignment in ProbabilisticBinaryClassifier. The distribution plotted shows the probability that a sequence is associated with a given class for a label.
ML model report	TCRdistMotifDiscovery	Discovers motifs in the clusters created by the TCRdist classifier and creates logo plots for these motifs. This report internally uses the tcrdist3[70] library.

Train ML model report	CVFeaturePerformance	Plots the performance on the training and test datasets as a function of a specific parameter (feature), where there are multiple training or test datasets, it plots the average performance for the given feature value.
Train ML model report	DiseaseAssociatedSequenceCVOverlap	For disease-associated sequences discovered by SequenceAbundanceEncoder and ProbabilisticBinaryClassifier (implementing the CMV study[6]), shows the overlap of the sequences across cross-validation folds as a heatmap.
Train ML model report	MLSettingsPerformance	Plots the performance of each ML setting (a combination of ML method, encoding, and optionally preprocessing).
Train ML model report	ReferenceSequenceOverlap	Compares a list of disease-associated sequences produced by the SequenceAbundance encoder to a list of reference sequences, and creates a Venn diagram representing the overlap.
Multi dataset report	DiseaseAssociatedSequenceOverlap	Creates a heatmap showing the overlap of disease-associated sequences produced by SequenceAbundance encoder between multiple datasets of different sizes (different number of repertoires per dataset).
Multi dataset report	PerformanceOverview	Creates a ROC plot and precision-recall plot for optimal trained ML models on multiple datasets.

Supplementary Table 2 | An overview of the settings for the five different simulated immune events. The signal of one immune event can be composed of one or more k-mers (here: 3-mers). For the gapped k-mers, a gap of length 1 was introduced with a 50% chance immediately before or after the middle amino acid of the 3-mer. In the two most complex signals, there was a 50% chance that one of the amino acids in the 3-mer was exchanged for a random different amino acid when implanting it in a sequence. Each signal (corresponding to one immune event) is implanted in a different subset of the repertoires. Corresponding results are shown in Figure 2G.

Immune event	**3-mers count**	**Gap length**	**Hamming distance**	**Repertoires with immune events**
Immune event 1	1	0	0	1–1000
Immune event 2	20	0	0	1–500, 1001–1500
Immune event 3	20	0 or 1 (50% chance)	0	1–250, 501–750, 1001–1250, 1501–1750
Immune event 4	20	0	0 or 1 (50% chance)	501–1500
Immune event 5	20	0 or 1 (50% chance)	0 or 1 (50% chance)	251–750, 1251–1750

Supplementary Table 3 | CMV-associated CDR3β sequences with V and J genes. These sequences were determined by training the predictive model on cohort 1 in use case 1 when reproducing the study by Emerson and colleagues[6] that were also found in the original study. The full list of CMV-associated sequences (including the non-overlapping ones) is available in the NIRD research data archive[78].

CDR3β amino acid sequence	TRBV gene	TRBJ gene
ASSYPGETQY	TRBV6-6	TRBJ2-5
ASSQVPGQGDNEQF	TRBV14	TRBJ2-1
ASSSPGRSGANVLT	TRBV28	TRBJ2-6
ASSLEGQQPQH	TRBV28	TRBJ1-5
AWRGTGNSPLH	TRBV30	TRBJ1-6
ASSGDRLYEQY	TRBV2	TRBJ2-7
ASSPDRVGQETQY	TRBV5-1	TRBJ2-5
ASRRGSSYEQY	TRBV28	TRBJ2-7
ASSLIGVSSYNEQF	TRBV7-9	TRBJ2-1
ASSISAGEAF	TRBV19	TRBJ1-1
ASSTGTSGSYEQY	TRBV6-1	TRBJ2-7
ASSPGDEQF	TRBV25-1	TRBJ2-1
ASSLQGADTQY	TRBV7-8	TRBJ2-3
ASSPAGLNTEAF	TRBV19	TRBJ1-1
ASSPLSDTQY	TRBV7-9	TRBJ2-3
ASSLVGDGYT	TRBV7-8	TRBJ1-2
ASSSDRVGQETQY	TRBV5-1	TRBJ2-5
ASSPNQETQY	TRBV5-4	TRBJ2-5
ASSRGTGATDTQY	TRBV19	TRBJ2-3
ASSALGGAGTGELF	TRBV9	TRBJ2-2
ASMGGASYEQY	TRBV27	TRBJ2-7
ASSAQGAYEQY	TRBV9	TRBJ2-7
ASSPPSGLTDTQY	TRBV28	TRBJ2-3
ASSQNRGQETQY	TRBV14	TRBJ2-5
SVRDNYNQPQH	TRBV29-1	TRBJ1-5
ASSLPSGLTDTQY	TRBV28	TRBJ2-3
ASSVTGGTDTQY	TRBV9	TRBJ2-3
ASSDRGNTGELF	TRBV4-1	TRBJ2-2
ASSLTDTGELF	TRBV11-2	TRBJ2-2
ASSGQGAYEQY	TRBV9	TRBJ2-7
ASSLQGINQPQH	TRBV5-6	TRBJ1-5
ASSLAGVDYEQY	TRBV7-9	TRBJ2-7

ASSLVAGGRETQY	TRBV5-6	TRBJ2-5
ASSPSTGTEAF	TRBV5-6	TRBJ1-1
ASSLRREKLF	TRBV5-6	TRBJ1-4
ASSLQGYSNQPQH	TRBV5-8	TRBJ1-5
ASSRNRGQETQY	TRBV14	TRBJ2-5
ASSLGHRDSSYEQY	TRBV5-1	TRBJ2-7
ASSTSGNTIY	TRBV6-5	TRBJ1-3
ASSYGGEGYT	TRBV6-5	TRBJ1-2
ASSPGSGANVLT	TRBV19	TRBJ2-6
ASSLGAGNQPQH	TRBV28	TRBJ1-5
ATSRGTVSYEQY	TRBV15	TRBJ2-7
ASSRLAGGTDTQY	TRBV7-3	TRBJ2-3
ASSIGPLEHNEQF	TRBV19	TRBJ2-1
ASSAGQGVTYEQY	TRBV9	TRBJ2-7
ASSLGDRAYNEQF	TRBV5-6	TRBJ2-1
ASSPLGGTTEAF	TRBV18	TRBJ1-1
ASRPTGYEQY	TRBV6-1	TRBJ2-7
ASSLLWDQPQH	TRBV5-5	TRBJ1-5
ASSTTGGDGYT	TRBV19	TRBJ1-2
ASSLAPGATNEKLF	TRBV7-6	TRBJ1-4
ASSSGQVQETQY	TRBV11-2	TRBJ2-5
ASSFPGGETQY	TRBV11-1	TRBJ2-5
AWSVSDLAKNIQY	TRBV30	TRBJ2-4
ASSTGGAQPQH	TRBV19	TRBJ1-5
ASSLGQGLAEAF	TRBV5-1	TRBJ1-1
ASSVDGGRGTEAF	TRBV9	TRBJ1-1
ASSPQRNTEAF	TRBV4-3	TRBJ1-1
ASSFPTSGQETQY	TRBV7-9	TRBJ2-5
ASSYNPYSNQPQH	TRBV6-6	TRBJ1-5
ASSLNRGQETQY	TRBV14	TRBJ2-5
ASTPGDEQF	TRBV25-1	TRBJ2-1
ASSLGVGPYNEQF	TRBV7-2	TRBJ2-1
ASSQNRAQETQY	TRBV14	TRBJ2-5
ASSIEGNQPQH	TRBV28	TRBJ1-5
SASDHEQY	TRBV20-1	TRBJ2-7
ASSRLAASTDTQY	TRBV7-3	TRBJ2-3
ASSPGDEQY	TRBV25-1	TRBJ2-7

ASGRDTYEQY	TRBV2	TRBJ2-7
ASSEEGIQPQH	TRBV2	TRBJ1-5
ASSRDRNYGYT	TRBV6-4	TRBJ1-2
ASSRGRQETQY	TRBV7-6	TRBJ2-5
ASSEARGGVEKLF	TRBV6-1	TRBJ1-4
ASSEIPNTEAF	TRBV6-4	TRBJ1-1
ASRGQGAGELF	TRBV2	TRBJ2-2
ATSREGSGYEQY	TRBV15	TRBJ2-7
ASSLEAEYEQY	TRBV7-2	TRBJ2-7
ASSLRGSSYNEQF	TRBV5-8	TRBJ2-1
ASSRNRAQETQY	TRBV14	TRBJ2-5
ASSLGWTEAF	TRBV5-1	TRBJ1-1
ASSYVRTGGNYGYT	TRBV6-5	TRBJ1-2
ATSRDTQGSYGYT	TRBV15	TRBJ1-2
SVRDNHNQPQH	TRBV29-1	TRBJ1-5
ATSRDSQGSYGYT	TRBV15	TRBJ1-2
ASSIRTNYYGYT	TRBV19	TRBJ1-2
ASSLETYGYT	TRBV5-6	TRBJ1-2
ATSRVAGETQY	TRBV15	TRBJ2-5
ASRPQGNYGYT	TRBV28	TRBJ1-2
ASSIWGLDTEAF	TRBV19	TRBJ1-1
ASSSDSGGTDTQY	TRBV7-3	TRBJ2-3
SVRDNFNQPQH	TRBV29-1	TRBJ1-5
ASSLTGGNSGNTIY	TRBV7-2	TRBJ1-3
SVEVRGTDTQY	TRBV29-1	TRBJ2-3
ASSSGTGDEQY	TRBV5-1	TRBJ2-7
ASSPRWQETQY	TRBV27	TRBJ2-5
ASSEARTRAF	TRBV6-1	TRBJ1-1
ASSVLAGPTDTQY	TRBV9	TRBJ2-3
ASSEEAGGSGYT	TRBV6-1	TRBJ1-2
ASRTDSGANVLT	TRBV6-4	TRBJ2-6
ASSEAPSTSTDTQY	TRBV2	TRBJ2-3
ASSRNRESNQPQH	TRBV6-5	TRBJ1-5
ASNRDRGRYEQY	TRBV6-1	TRBJ2-7
ASSLGASGSRTDTQY	TRBV7-9	TRBJ2-3
ASSESGDPSSYEQY	TRBV10-1	TRBJ2-7
ASSLGDRPDTQY	TRBV11-2	TRBJ2-3

ASSLQAGANEQF	TRBV7-2	TRBJ2-1
ASRTGESGYT	TRBV6-5	TRBJ1-2
ASRGQGWDEKLF	TRBV6-5	TRBJ1-4
ASSWDRGTEAF	TRBV6-5	TRBJ1-1
ASSHRDRNYEQY	TRBV7-9	TRBJ2-7
ASSPSRNTEAF	TRBV4-3	TRBJ1-1
ASSIQGYSNQPQH	TRBV5-8	TRBJ1-5
ASSPGQEAGANVLT	TRBV5-1	TRBJ2-6
ASSLVIGGDTEAF	TRBV5-1	TRBJ1-1
ASSYGGLGSYEQY	TRBV6-5	TRBJ2-7
ASSPPGQGSDTQY	TRBV18	TRBJ2-3
SVEEDEGIYGYT	TRBV29-1	TRBJ1-2
ASRSDSGANVLT	TRBV6-4	TRBJ2-6
AISESQDRGHEQY	TRBV10-3	TRBJ2-7
ASSLVASGRETQY	TRBV5-6	TRBJ2-5
ASSSGQVYGYT	TRBV5-6	TRBJ1-2
ASSQGRHTDTQY	TRBV14	TRBJ2-3
ASSGLNEQF	TRBV6-1	TRBJ2-1
ASRDWDYTDTQY	TRBV2	TRBJ2-3
ASSSRGTGELF	TRBV28	TRBJ2-2
ASSPISNEQF	TRBV28	TRBJ2-1
ASSLGHRDPNTGELF	TRBV5-1	TRBJ2-2
ASSLGIDTQY	TRBV5-4	TRBJ2-3
ASSLEGQGFGYT	TRBV5-1	TRBJ1-2
ASSPHRNTEAF	TRBV4-3	TRBJ1-1
ASSESGHRNQPQH	TRBV10-2	TRBJ1-5
ASSFHGFNQPQH	TRBV5-6	TRBJ1-5
ASSEGARQPQH	TRBV10-2	TRBJ1-5
ASSSRTGEETQY	TRBV11-3	TRBJ2-5
ASSLEAENEQF	TRBV7-2	TRBJ2-1
ASSLVAAGRETQY	TRBV5-6	TRBJ2-5
ASSLAVLPTDTQY	TRBV7-9	TRBJ2-3
ASSLGRGYEKLF	TRBV5-6	TRBJ1-4
ASSWDRDNSPLH	TRBV25-1	TRBJ1-6
ASRDRDRVNTEAF	TRBV6-1	TRBJ1-1
ASSPTGGELF	TRBV18	TRBJ2-2
ASSVETGGTEAF	TRBV2	TRBJ1-1

Supplementary Table 4 | Three epitope-specific datasets in the use case 2.

Dataset	**Number of TCRαβ receptors**	**Epitope**	**Epitope species**	**Data source: epitope-specific TCRs**	**Data source: naive receptors**	**Organism**
AVFDRKSDAK	3460	AVFDRKSDAK	EBV	VDJdb[58]: 10x Genomics[89] and EBV study[90]	immuneACCESS: randomly paired TCRαβ CDR3 from peripheral blood of 4 healthy donors from Heikkilä et al.[80]	Human
GILGFVFTL	4090	GILGFVFTL	Influenza A	VDJdb[58]: multiple studies[17,18,89,91–95]		
KLGGALQAK	27386	KLGGALQAK	CMV	VDJdb[58]: 10x Genomics[89]		

References

83. Berman, H. M. *et al.* The Protein Data Bank. *Nucleic Acids Res.* **28**, 235–242 (2000).
84. Schrödinger, LLC. *The PyMOL Molecular Graphics System*. (2015).
85. *immunoSEQ Analyzer: From Sequencing Data to Insights* (immunoSEQ, 2021); https://www.immunoseq.com/analyzer/
86. Greiff, V. *et al.* A bioinformatic framework for immune repertoire diversity profiling enables detection of immunological status. *Genome Med.* **7**, 49 (2015). https://doi.org/10.1186/s13073-015-0169-8
87. Rehurek, R. & Sojka, P. Software framework for topic modelling with large corpora. in *Proc. Language Resource and Evaluation Conference (LREC) 2010 Workshop on New Challenges for NLP Frameworks* 45–50 (ELRA, 2010).
88. Shannon, P. *et al.* Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res.* **13**, 2498–2504 (2003).
89. *A New Way of Exploring Immunity: Linking Highly Multiplexed Antigen Recognition to Immune Repertoire and Phenotype* (10x Genomics, 2021). https://pages.10xgenomics.com/3p-immunology-app-note-new-way-exploring-immunity.html
90. Campos-Lima, P. O., Levitsky, V., Imreh, M. P., Gavioli, R. & Masucci, M. G. Epitope-dependent selection of highly restricted or diverse T cell receptor repertoires in response to persistent infection by Epstein-Barr virus. *J. Exp. Med.* **186**, 83–89 (1997).
91. Chen, G. *et al.* Sequence and Structural Analyses Reveal Distinct and Highly Diverse Human CD8+ TCR Repertoires to Immunodominant Viral Antigens. *Cell Rep.* **19**, 569–583 (2017).
92. Grant, E. J. *et al.* Lack of Heterologous Cross-reactivity toward HLA-A*02:01 Restricted Viral Epitopes Is Underpinned by Distinct αβT Cell Receptor Signatures. *J. Biol. Chem.* **291**, 24335–24351 (2016).
93. Wang, Z. *et al.* Clonally diverse CD38+HLA-DR+CD8+ T cells persist during fatal H7N9 disease. *Nat. Commun.* **9**, 824 (2018).
94. Sant, S. *et al.* Single-Cell Approach to Influenza-Specific CD8+ T Cell Receptor Repertoires Across Different Age Groups, Tissues, and Following Influenza Virus Infection. *Front. Immunol.* **9**, 1453 (2018).
95. Lehner, P. J. *et al.* Human HLA-A0201-restricted cytotoxic T lymphocyte recognition of influenza A is dominated by T cells bearing the V beta 17 gene segment. *J. Exp. Med.* **181**, 79–91 (1995).

II

Bioinformatics, 38(17), 2022, 4230–4232
https://doi.org/10.1093/bioinformatics/btac505
Advance Access Publication Date: 19 July 2022
Applications Note

Sequence analysis

CompAIRR: ultra-fast comparison of adaptive immune receptor repertoires by exact and approximate sequence matching

Torbjørn Rognes [1,2,3,†], Lonneke Scheffer [1,3,†], Victor Greiff [4] and Geir Kjetil Sandve [1,3,*]

[1]Department of Informatics, University of Oslo, 0316 Oslo, Norway, [2]Department of Microbiology, Oslo University Hospital, 0424 Oslo, Norway, [3]Centre of Bioinformatics, University of Oslo, 0316 Oslo, Norway and [4]Department of Immunology, University of Oslo and Oslo University Hospital, 0424 Oslo, Norway

*To whom correspondence should be addressed.
[†]The authors wish it to be known that, in their opinion, the first two authors should be regarded as Joint First Authors.
Associate Editor: Alfonso Valencia

Received on December 9, 2021; revised on March 20, 2022; editorial decision on July 2, 2022; accepted on July 18, 2022

Abstract

Motivation: Adaptive immune receptor (AIR) repertoires (AIRRs) record past immune encounters with exquisite specificity. Therefore, identifying identical or similar AIR sequences across individuals is a key step in AIRR analysis for revealing convergent immune response patterns that may be exploited for diagnostics and therapy. Existing methods for quantifying AIRR overlap scale poorly with increasing dataset numbers and sizes. To address this limitation, we developed CompAIRR, which enables ultra-fast computation of AIRR overlap, based on either exact or approximate sequence matching.
Results: CompAIRR improves computational speed 1000-fold relative to the state of the art and uses only one-third of the memory: on the same machine, the exact pairwise AIRR overlap of 10^4 AIRRs with 10^5 sequences is found in $\sim$17 min, while the fastest alternative tool requires 10 days. CompAIRR has been integrated with the machine learning ecosystem immuneML to speed up commonly used AIRR-based machine learning applications.
Availability and implementation: CompAIRR code and documentation are available at https://github.com/uio-bmi/compairr. Docker images are available at https://hub.docker.com/r/torognes/compairr. The code to replicate the synthetic datasets, scripts for benchmarking and creating figures, and all raw data underlying the figures are available at https://github.com/uio-bmi/compairr-benchmarking.
Contact: geirksa@ifi.uio.no
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Adaptive immune receptor (AIR) repertoires (AIRRs) record past immune encounters. High-throughput sequencing now enables millions of AIR sequences to be determined at a cost that facilitates adaptive immunity-based association studies on large patient cohorts (Emerson *et al.*, 2017; Liu *et al.*, 2019). It has been previously shown that shared immune states give rise to identical or similar AIR sequences across individuals, enabling the use of AIRR-seq for diagnostics and therapeutic research (Arnaout *et al.*, 2021; Greiff *et al.*, 2020). Computation of cross-individual AIRR intersections, i.e. the number of matching AIR sequences across AIRRs, is thus a foundational computational task performed in nearly all AIRR analyses. However, since the number of pairwise AIRR comparisons grows asymptotically quadratically with the number of AIRRs considered, where each pairwise AIRR comparison typically involves millions of individual AIRs, computational efficiency is crucial for performing AIR sequence matching at scale.

We here present CompAIRR, a tool that allows to compute AIRR intersections up to 1000-fold faster than current implementations (Nazarov *et al.*, 2019; Shugay *et al.*, 2015; Weber *et al.*, 2022). In contrast to existing tools, CompAIRR supports both exact and approximate sequence matching between AIRs when determining AIRR overlap. The CompAIRR implementation is available both as a stand-alone command-line tool, and as a component integrated with the machine learning ecosystem immuneML (Pavlović *et al.*, 2021) (from immuneML version 2.1.0 onward) to accelerate the computation of AIRR similarity matrices, and to accelerate an AIRR-based immune state classifier (Emerson *et al.*, 2017) that is implemented in the immuneML system (Supplementary Fig. S1).

2 CompAIRR description

CompAIRR is based on a sequence comparison strategy developed for the nucleotide sequence clustering tool Swarm (Mahé *et al.*, 2022). A Bloom filter (Putze *et al.*, 2010) and a hash table are used to quickly look up similar AIR sequences across AIRR sets. For each AIR sequence (nucleotide or amino acid), a 64-bit hash value is generated using a Zobrist hash function (Zobrist, 1970), a form of tabulation hashing that can be computed very efficiently and updated incrementally. When approximate matching is enabled, the hashes of all possible variants of a query sequence (with 1–2 substitutions or indels) are also generated. This search strategy identifies all matching sequences without compromising on accuracy. CompAIRR version 1.7.0 or later also supports a larger number of substitutions by using a simpler all-versus-all algorithm. Matches are optionally restricted by V and J gene. Multi-threading may be enabled to further speed up comparisons (see Fig. 1d). For the comparison of n AIRRs, CompAIRR produces an $n \times n$ matrix where each cell contains the sum of matching AIR frequencies with flexible summary statistics (product, min, max, mean or ratio of the two compared AIR frequencies), or the Morisita-Horn or Jaccard index between AIRRs. Alternatively, CompAIRR can query n AIRRs against m reference AIRs and produce an $n \times m$ sequence presence table. While AIR matching is only supported at the single chain level, two $n \times m$ sequence presence tables for complementary (paired) AIR chains (single-cell data) can easily be merged. For the analysis of a single AIRR, CompAIRR can perform single-linkage clustering of AIRs. CompAIRR can optionally output the list of (approximately) matching AIRs as an AIRR-compliant TSV file, and adheres to the AIRR standard for software tools (Vander Heiden *et al.*, 2018).

3 CompAIRR performance benchmarking

CompAIRR (1.3.1) was benchmarked against VDJtools (1.2.1) (Shugay *et al.*, 2015), immunarch (0.6.5) (Nazarov *et al.*, 2019) and immuneREF (0.5.0) (Weber *et al.*, 2022) by calculating the pairwise AIRR overlap of datasets ranging from 10 to 10^4 AIRRs. Each AIRR consisted of 10^5 amino acid AIR sequences generated using OLGA (1.2.2) (Sethna *et al.*, 2019) with the default human IgH CDR3 model. Figure 1b and c, respectively, shows the running time and maximum RAM usage of each tool. CompAIRR is consistently faster, particularly for large datasets: with 10^4 AIRRs of 10^5 sequences, CompAIRR ran in 17 min while immunarch took 10 days, immuneREF took 23 days and VDJtools failed to complete due to memory constraints. The computational complexity appears to have been reduced from approximately quadratic to almost linear. Furthermore, the maximum RAM usage of CompAIRR is below one-third of that of competing tools. The running time and memory usage as a function of the AIRR size (10^4–10^6 sequences) is shown in Supplementary Figure S2.

In addition, Figure 1d shows how the CompAIRR running time is affected by approximate sequence matching, which is not at all supported by the existing tools. The benefit of multi-threading becomes more apparent when the degree of sequence mismatching is increased, since with exact matching the running time is dominated by disk access (Supplementary Fig. S3).

4 Conclusion

The identification of shared AIRs across AIRRs from different individuals is a core computational task in AIRR analysis. We have here presented CompAIRR, which calculates AIRR overlap up to 1000-fold faster while its peak memory usage is below one third compared to currently available tools. We validated that CompAIRR easily scales to datasets of 10^4 AIRRs of 10^5 sequences each, which surpass the largest available experimental datasets (Liu *et al.*, 2019; Nolan *et al.*, 2020). Furthermore, a novel feature of CompAIRR is efficient

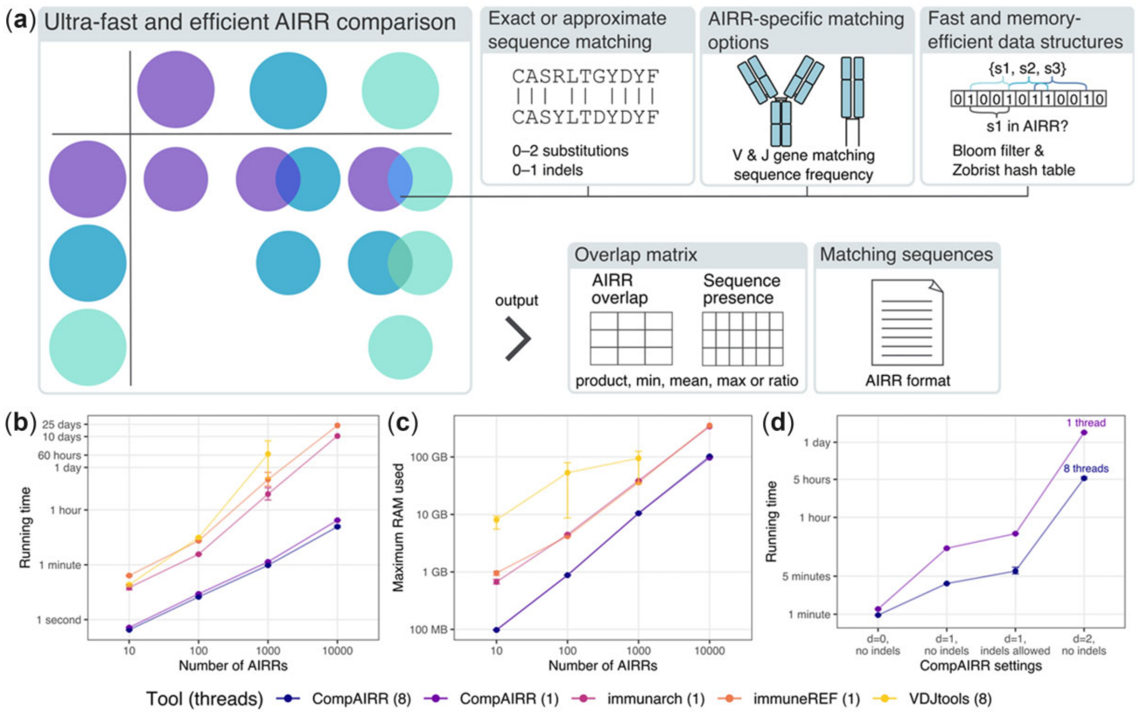


Fig. 1. Overview of CompAIRR features and performance. (**a**) CompAIRR has configurable AIR matching criteria and output formats. (**b**) CompAIRR calculates pairwise AIRR overlap up to 1000-fold faster than currently available tools. (**c**) The maximum RAM usage of CompAIRR is below one-third of the most memory-efficient alternative. (**d**) The CompAIRR running time increases when allowing more AIR sequence mismatches, but multithreading helps reduce this running time. (**b–d**) Data shown are mean with error bars showing min/max values across three replicate runs. For the largest dataset, only CompAIRR was run three times, and VDJtools failed to run due to memory limitations. Unless otherwise specified, datasets consist of 1000 AIRRs containing 10^5 OLGA-generated sequences (Sethna *et al.*, 2019) (default human IgH CDR3 model)

identification of *approximately* matching AIR sequences across AIRRs or to reference databases, which may be a biologically meaningful way to increase the number of matches between AIRRs when the exact overlap is low (Supplementary Fig. S4).

Complementary to sequence-level clustering tools ClusTCR (Valkiers *et al.*, 2021) and GIANA (Zhang *et al.*, 2021), or comparison of AIRR subsets (Yohannes *et al.*, 2021), CompAIRR can be used for ultrafast similarity-based comparison of *complete* AIRRs. Due to flexible specification of summary statistics and output, CompAIRR is easily integrated with any tool capable of reading in either (i) a pairwise distance matrix containing cross-AIRR matches, (ii) a matrix showing individual AIR presence in one or more AIRRs or (iii) an AIRR-compliant TSV file containing (approximately) matching AIRs between AIRRs. This allows accelerating a variety of analyses where AIRR comparison is a core computational component, including AIRR similarity (Weber *et al.*, 2022) and clustering (Rempała and Seweryn, 2013; Shugay *et al.*, 2015), phylogenetic clustering (Hoehn *et al.*, 2022), graph analysis (Madi *et al.*, 2017; Miho *et al.*, 2019; Pogorelyy *et al.*, 2019) and immune state classification (Emerson *et al.*, 2017).

Acknowledgements

The authors are grateful to Milena Pavlović for her assistance. They are also grateful for access to computational resources provided by the Norwegian Research and Education Cloud (NREC) and UNINETT Sigma2 (project NN9383K).

Funding

The authors acknowledge generous support by the Leona M. and Harry B. Helmsley Charitable Trust [#2019PG-T1D011 to V.G.], UiO World-Leading Research Community [to V.G.], UiO: LifeScience Convergence Environment Immunolingo [to V.G. and G.K.S.], EU Horizon 2020 iReceptorplus [#825821 to V.G.], a Research Council of Norway FRIPRO project [#300740 to V.G.], a Research Council of Norway IKTPLUSS project [#311341 to V.G. and G.K.S.] and Stiftelsen Kristian Gerhard Jebsen (K.G. Jebsen Coeliac Disease Research Centre) [G.K.S.].

Conflict of Interest: V.G. declares advisory board positions in aiNET GmbH and Enpicom B.V. V.G. is a consultant for Roche/Genentech.

References

Arnaout,R.A. *et al.*; Adaptive Immune Receptor Repertoire Community. (2021) The future of blood testing is the immunome. *Front. Immunol.*, **12**, 626793.

Emerson,R.O. *et al.* (2017) Immunosequencing identifies signatures of cytomegalovirus exposure history and HLA-mediated effects on the T cell repertoire. *Nat. Genet.*, **49**, 659–665.

Greiff,V. *et al.* (2020) Mining adaptive immune receptor repertoires for biological and clinical information using machine learning. *Curr. Opin. Syst. Biol.*, **24**, 109–119.

Hoehn,K.B. *et al.* (2022) Phylogenetic analysis of migration, differentiation, and class switching in B cells. *PLoS Comput. Biol.*, **18**, e1009885.

Liu,X. *et al.* (2019) T cell receptor β repertoires as novel diagnostic markers for systemic lupus erythematosus and rheumatoid arthritis. *Ann. Rheum. Dis.*, **78**, 1070–1078.

Madi,A. *et al.* (2017) T cell receptor repertoires of mice and humans are clustered in similarity networks around conserved public CDR3 sequences. *eLife*, **6**, e22057.

Mahé,F. *et al.* (2022) Swarm v3: towards tera-scale amplicon clustering. *Bioinformatics*, **38**, 267–269.

Miho,E. *et al.* (2019) Large-scale network analysis reveals the sequence space architecture of antibody repertoires. *Nat. Commun.*, **10**, 1321.

Nazarov,V.I. *et al.* (2019) Immunarch: an R package for painless bioinformatics analysis of T-cell and B-cell immune repertoires. *Zenodo*, https://doi.org/10.5281/zenodo.3367200.

Nolan,S. *et al.* (2020) A large-scale database of T-cell receptor beta (TCRβ) sequences and binding associations from natural and synthetic exposure to SARS-CoV-2. *Res Sq*, rs.3.rs-51964.

Pavlović,M. *et al.* (2021) The immuneML ecosystem for machine learning analysis of adaptive immune receptor repertoires. *Nat. Mach. Intell.*, **3**, 936–944.

Pogorelyy,M.V. *et al.* (2019) Detecting T cell receptors involved in immune responses from single repertoire snapshots. *PLoS Biol.*, **17**, e3000314.

Putze,F. *et al.* (2010) Cache-, hash-, and space-efficient bloom filters. *ACM J. Exp. Algorithmics*, **14**, 4:4.4–4:4.18.

Rempała,G.A. and Seweryn,M. (2013) Methods for diversity and overlap analysis in T-cell receptor populations. *J Math Biol*, **67**, 1339–1368.

Sethna,Z. *et al.* (2019) OLGA: fast computation of generation probabilities of B- and T-cell receptor amino acid sequences and motifs. *Bioinformatics*, **35**, 2974–2981.

Shugay,M. *et al.* (2015) VDJtools: unifying post-analysis of T cell receptor repertoires. *PLoS Comput. Biol.*, **11**, e1004503.

Valkiers,S. *et al.* (2021) ClusTCR: a python interface for rapid clustering of large sets of CDR3 sequences with unknown antigen specificity. *Bioinformatics*, **37**, 4865–4867.

Vander Heiden,J.A. AIRR Community. *et al.* (2018) AIRR community standardized representations for annotated immune repertoires. *Front. Immunol.*, **9**, 2206.

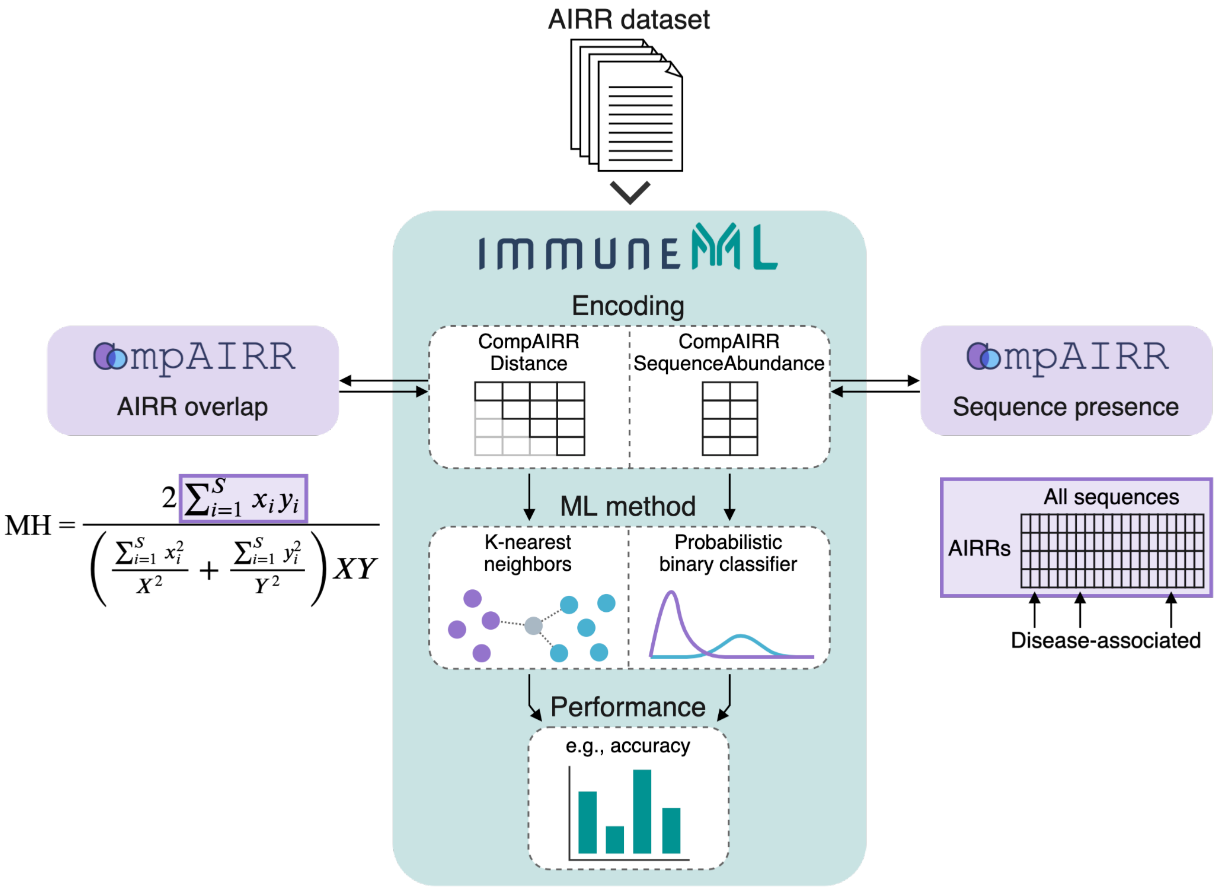
Weber,C.R. *et al.* (2022) Reference-based comparison of adaptive immune receptor repertoires. *bioRxiv*, 2022.01.23.476436.

Yohannes,D.A. *et al.* (2021) Clustering based approach for population level identification of condition-associated T-cell receptor β-chain CDR3 sequences. *BMC Bioinformatics*, **22**, 159.

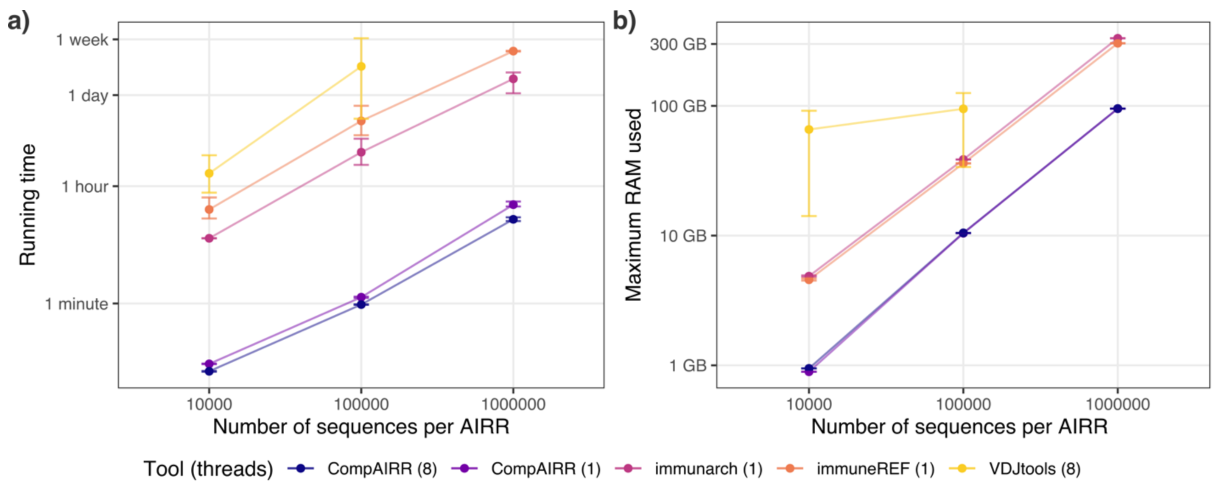
Zhang,H. *et al.* (2021) GIANA allows computationally-efficient TCR clustering and multi-disease repertoire classification by isometric transformation. *Nat. Commun.*, **12**, 4699.

Zobrist,A.L. (1970) A new hashing method with application for game playing. *Technical Report 88*. University of Wisconsin-Madison Department of Computer Sciences.

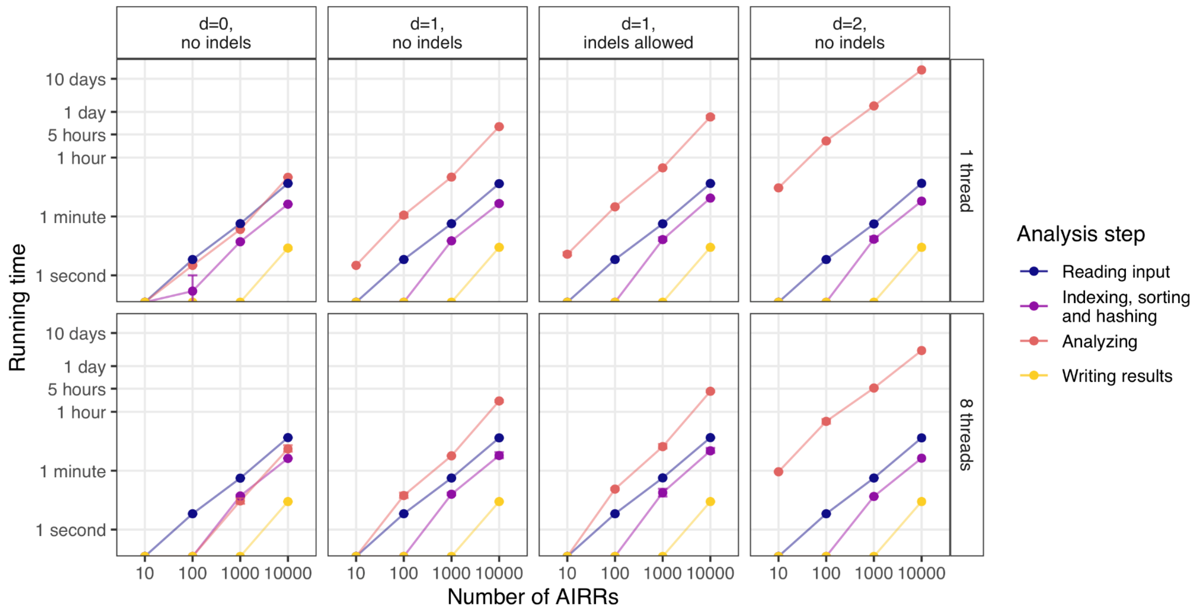
Supplementary Material


AIRR dataset
immuneML
Encoding
CompAIRR Distance
CompAIRR SequenceAbundance
CompAIRR
AIRR overlap
CompAIRR
Sequence presence
$\mathrm{MH} = \frac{2\boxed{\sum_{i=1}^{S} x_i y_i}}{\left(\frac{\sum_{i=1}^{S} x_i^2}{X^2} + \frac{\sum_{i=1}^{S} y_i^2}{Y^2}\right)XY}$
ML method
K-nearest neighbors
Probabilistic binary classifier
All sequences
AIRRs
Disease-associated
Performance
e.g., accuracy

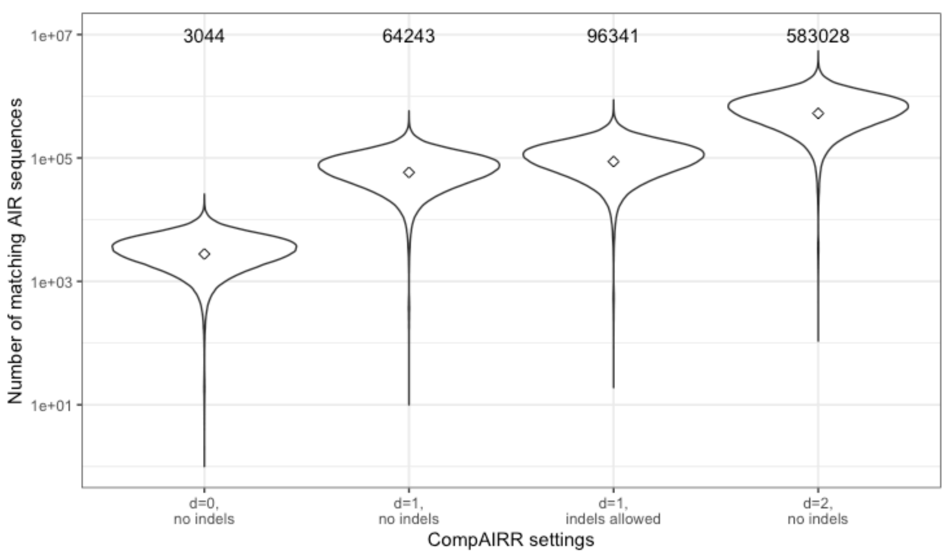
Supplementary Figure 1 | CompAIRR is integrated with immuneML to speed up core calculations of two different encodings. Firstly, CompAIRR accelerates the calculation of a Morisita-Horn (MH) distance matrix by calculating the sum of the products of overlapping AIR frequencies between AIRRs (left, highlighted in purple). This distance matrix can subsequently be used in combination with a k-nearest neighbors classifier. Secondly, calculating a sequence presence matrix with CompAIRR (right, highlighted in purple) speeds up an AIRR classification method (Emerson *et al.*, 2017), which was replicated in the immuneML ecosystem (Pavlović *et al.*, 2021) by a sequence abundance encoding (AIRR size and number of disease-associated AIRs per AIRR) and a probabilistic binary classifier.



Supplementary Figure 2 | Benchmarking of (a) running time and (b) maximum RAM usage as a function of the number of sequences per AIRR. Data shown are mean with error bars showing min/max values across three replicate runs. Datasets consist of 1000 OLGA-generated AIRRs (Sethna *et al.*, 2019) (default human IgH CDR3 model). VDJtools failed to run on the dataset of 10^6 AIRRs due to memory limitations.



Supplementary Figure 3 | Running time of CompAIRR analysis steps. Reading input and analyzing the AIRR overlap are the most computationally intensive steps. When exact sequence matching is used, reading input data takes a relatively large amount of the total running time. As a result, the benefit of multi-threading is more apparent when non-exact matching is used.



Supplementary Figure 4 | Number of matches with approximate sequence matching settings. The number of matching TCRβ AIR sequences between each combination of 710 AIRRs (Emerson *et al.*, 2017) increases when less stringent sequence matching criteria are used. When no or few exact overlapping sequences are found across AIRRs, highly similar sequences may still be present. Medians are represented by rhombuses and their values are displayed above the violin plots.

III