Published in final edited form as:

Nat Mach Intell. 2021 November ; 3(11): 936–944. doi:10.1038/s42256-021-00413-z.

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

Milena Pavlović[1,2,3,#], **Lonneke Scheffer**[1,2,#], **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,☯], **Geir Kjetil Sandve**[1,2,3,☯]

[1]Department of Informatics, University of Oslo, Norway

[2]Centre for Bioinformatics, University of 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, USA

[5]Department of Pediatrics, College of Medicine, University of Florida Diabetes Institute, USA

[6]University Center for Information Technology, University of Oslo, Norway

[7]Institute for Systems and Computer Engineering, Technology and Science, Portugal

[8]Biological Sciences, Simon Fraser University, Canada

[9]Department of Immunology, University of Oslo, Norway

[#]Equal contribution

[☯]Joint supervision

Author contributions

MP, VG, GKS conceived the study. MP and GKS designed the overall software architecture. MP, LS, and KM developed the main platform code. MP and LS performed all analyses. MP, LS, CK, FLMB, RA, GSAH, GB, MC, RF, IG, SG, PHH, KR, ER, PAR, AS, DT, CW, and MW created software or documentation content. RK, NV, KW, LS, MP, AAC, and BC designed and developed the Galaxy tools. CK, RA, TB, MC, SC, LGC, IHH, EH, GK, MLK, CLA, AM, TM, JP, KR, PAR, AR, IS, LMS and GY provided critical feedback. MP, LS, VG, GKS drafted the manuscript. VG and GKS supervised the project. All authors read and approved the final manuscript and are personally accountable for its content.

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 DOI: doi.org/10.5281/zenodo.5118741[75]. The immuneML Python package can be downloaded from pypi.org/project/immuneML/.

Competing Interests

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

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

[10]Department of Population and Data Sciences, UT Southwestern Medical Center, USA

[11]Department of Mathematics, University of Oslo, Norway

[12]ELLIS Unit Linz and LIT AI Lab, Institute for Machine Learning, Johannes Kepler University Linz, Austria

[13]Department of Tumor Biology, Institute for Cancer Research, Oslo University Hospital, The Norwegian Radium Hospital, Norway

[14]Centre for Molecular Medicine Norway (NCMM), Nordic EMBL Partnership, University of Oslo, Norway

[15]Department of Pathology, Leiden University Medical Center, the Netherlands

[16]Faculty of Medicine, Institute of Clinical Medicine, University of Oslo, Norway

[17]Department of Biosystems Science and Engineering, ETH Zürich, Switzerland

[18]Faculty of Engineering, Bar Ilan University, Israel

Abstract

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. To date, 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 (i) reproducing a large-scale study on immune state prediction, (ii) developing, integrating, and applying a novel deep learning method for antigen specificity prediction, and (iii) showcasing streamlined interpretability-focused benchmarking of AIRR ML

Editor summary:

The proliferation of molecular biology and bioinformatics tools necessary to generate huge quantities of immune receptor data has not been matched by frameworks that allow for easy data analysis. The authors present immuneML, an open-source collaborative ecosystem for machine learning analysis of adaptive immune receptor repertoires.

Introduction

T-cell receptors (TCRs) and B-cell receptors (BCRs), that 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 (e.g., 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, s*equence*-based prediction of antigen and epitope binding is of fundamental importance for AIR-based therapeutics discovery and engineering[15–27]. In this manuscript, the term *AIRR* signifies both AIRs and AIRRs (a collection of AIRs) if not specified otherwise.

Machine learning (ML) has recently entered center stage in the biological sciences because it allows detection, recovery, and re-creation 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, (i) ~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 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]. (ii) 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 into a very low rate of relevant sequences per repertoire (low witness rate)[11,39,40]. (iii) 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 to define, implement, and train 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

immuneML overview

Here, we present immuneML, an open-source collaborative ecosystem for AIRR ML (Figure 1). immuneML enables the ML study of both experimental and synthetic AIRR-seq data that are labeled on the repertoire-level (e.g., immune state, sex, age, or any other metadata) or sequence-level (e.g., antigen binding), all the way from preprocessing to model training and model interpretation. It natively implements model selection and assessment procedures like 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 to cloud services such as Amazon Web Services (AWS) and Google Cloud, or on a local server for data privacy concerns. Computational reproducibility and transparency are achieved by shareable specification files, which include all analysis details (Supplementary Figure 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 AIRR ML results' downstream analysis. 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. Additionally, immuneML is integrated with the AIRR-specific attention-based multiple-instance learning ML method DeepRC[39], the TCR-specific clustering method TCRdist[17], and is compatible with GLIPH2[59].

To get started with immuneML, we refer the reader to Focus 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: (i) 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, (ii) 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 (iii) 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 it complies 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 also synthetic data as generated by immuneSIM[62] can be imported. Importing synthetic data as generated by IGoR[63] and OLGA[64] is also supported. 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 to download 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 includes 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 for interpretation of ML models

immuneML supports two major ML platforms to ensure flexibility: scikit-learn[47] and PyTorch[48] and, therefore, is 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 denoting appropriate amino acid or nucleotide, encodings by 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 as part of an analysis, providing details about the encoded data (e.g., feature value distributions), the ML model (e.g., interpretability reports), and the prediction accuracy (a variety of performance metrics across training, validation, and test sets). Additionally, the trained models may be exported and used in future analyses.

immuneML facilitates 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] (Figure 1).

Usability is achieved by a range of installation and usage options, catered 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. 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] that 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 we are here referring to reproducibility mainly 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 well suited to explore the extent to which results are affected by modifications of analysis parameters.

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

Extensibility of immuneML, signifying straightforward inclusion of new ML methods, encodings, reports, and preprocessing, is ensured by its modular design (Supplementary Figure 2). The code is open-source and available on GitHub (Focus 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 (i) a YAML analysis specification in which the assumptions of the AIRR ML analysis are explicitly defined, and default parameter settings are exported, (ii) 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 Figure 3) and (iii) for each analysis report, the availability of underlying data for further user inspection.

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] (Figure 2A). Using the standard interface of immuneML, we set up a repertoire classification analysis using 10-fold 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. 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 (see Methods).

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 further used built-in robustness analysis of immuneML to explore how classification accuracy and the set of immune-status-associated sequences varied when learning classifiers based on smaller subsets of repertoires (Figure 2 A and 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) (Figure 2 B and 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 is to discover motifs in the two receptor chains (sequences) and to exploit the presence of these motifs to predict if the receptor will bind the antigen. As the immuneML platform provides comprehensive functionality for parsing and encoding paired-chain data, for hyperparameter optimization, and for presenting results, the only development step needed was to add the code for the CNN-based method itself (Supplementary Figure 5). Briefly, the added CNN consists 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 if the receptor will 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 neighbor 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] (Figure 2 D).

Use case 3: ML methods benchmarking on ground-truth synthetic data

Given the current rise in AIRR ML applications, the ability for method developers and practitioners to efficiently benchmark the variety of available approaches is becoming crucial[1,15,62]. Due to the limited current availability of high-resolution, labeled 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, using the OLGA framework[64], 2000 human IgH repertoires consisting of 10^5 CDR3 amino acid sequences each, and implanted sequence motifs reflecting five different immune events of varying complexity (Figure 2 G, Supplementary Table 2). We examined the classification accuracy of three assessed ML methods (Figure 2 H) and used a native immuneML report to examine the overlap between ground truth implanted motifs and learned model features (Figure 2 I, Supplementary Figure 6).

Discussion

We have presented immuneML, a collaborative and open-source platform for transparent AIRR ML, 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 be easily 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, so as to improve software extensibility and maintainability. With the field of ML maturing, we see such aspects connected to 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 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* to effectively evaluate 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, pre-processing, hyperparameter optimization strategies, interpretability, results presentation. The fact that the immuneML architecture builds strictly 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, hyper-parameter optimization, and performance assessment can 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 modeling[15,20,73]. We also aim to improve the general support for model introspection, in particular in the direction of supporting causal interpretations for discovering and alleviating technical biases or challenges related to the study design[74].

In conclusion, immuneML enables the transition of AIRR ML method setup representing a bona fide research project to being at the fingertips of immunologists and clinicians. Complementally, AIRR ML method developers can 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 (i) as a web tool through the Galaxy web interface (galaxy.immuneml.uio.no), (ii) from a command-line interface (CLI), (iii) through Docker (hub.docker.com/repository/docker/milenapavlovic/immuneml), (iv) via cloud services such as Google Cloud (cloud.google.com) through Docker integration, or (v) as a Python library (pypi.org/project/immuneML). It is also deposited on Zenodo with DOI: doi.org/10.5281/zenodo.5118741[75].

immuneML analysis specification: immuneML analyses are specified using a YAML specification file (Supplementary Figure 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 CLI 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 instructions: (i) training and assessment of ML models, (ii) applications of trained ML models, (iii) exploratory data analysis, and (iv) generation of synthetic AIRR datasets. Tutorials detailing these instructions are available at docs.immuneml.uio.no/latest/tutorials.html.

immuneML public instance: 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 Figure 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 the users. Such extensibility mechanisms allow the 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 (CV), allowing newly developed components for either of these parts (data processing, encoding, ML method) to be selected in competition against 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 Figure 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 Emerson et al. study, 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 (Figure 2 B and C). While most of the results are consistent within the subsampled dataset size, in Figure 2 B, 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 Figure 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. Subsequently, we 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. Additionally, we 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 compare these recovered motifs with the motifs extracted by GLIPH2[59] on the same datasets. Each dataset includes a set of epitope-specific TCR-β receptors downloaded from VDJdb and a set of naive, randomly paired TCR-β receptors from the peripheral blood samples of 4 healthy donors[80]. Epitope-specific datasets are specific to cytomegalovirus (KLGGALQAK epitope, with 13000 paired TCR-β receptors), Influenza A (GILGFVFTL epitope, with 2000 paired TCR-β receptors), and Epstein-Barr virus (AVFDRKSDAK epitope, with 1700 paired TCR-β 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, 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 immuneML's utility 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, 2000 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, so as to allow for a demonstration of how benchmarking may be performed using basic ML approaches.

Using immuneML, three different ML methods (logistic regression, random forest, 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 2000 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, 3-fold cross-validation was used. The test set classification performances of the trained classifiers for each immune event are shown in Figure 2 H.

The immune signals implanted in this dataset can be 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 (Figure 2 I, Supplementary Figure 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].

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgements

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

Data availability

All data for the analyses presented in the manuscript are openly available. The detailed result files for the use cases presented in the manuscript are available as zip files with separate DOIs per use case: doi.org/10.11582/2021.00008[78] (use case 1), doi.org/10.11582/2021.00009[81] (use case 2), doi.org/10.11582/2021.00005[82] (use case 3).

Input data for use case 1 was downloaded from doi.org/10.21417/B7001Z.

References

1. Brown AJ 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). [PubMed: 24441474]
3. Yaari G. & Kleinstein SH Practical guidelines for B-cell receptor repertoire sequencing analysis. Genome Med. 7, 121 (2015). [PubMed: 26589402]
4. Csepregi L, Ehling RA, Wagner B. & Reddy ST Immune Literacy: Reading, Writing, and Editing Adaptive Immunity. iScience 23, 101519 (2020).
5. DeWitt WS III et al. Human T cell receptor occurrence patterns encode immune history, genetic background, and receptor specificity. eLife 7, e38358 (2018). [PubMed: 30152754]
6. Emerson RO et al. Immunosequencing identifies signatures of cytomegalovirus exposure history and HLA-mediated effects on the T cell repertoire. Nat. Genet 49, 659–665 (2017). [PubMed: 28369038]
7. Krishna C, Chowell D, Gönen M, Elhanati Y. & Chan TA Genetic and environmental determinants of human TCR repertoire diversity. Immun. Ageing 17, 26 (2020). [PubMed: 32944053]
8. Britanova OV et al. Age-Related Decrease in TCR Repertoire Diversity Measured with Deep and Normalized Sequence Profiling. J. Immunol 192, 2689–2698 (2014). [PubMed: 24510963]
9. Schneider-Hohendorf T. et al. Sex bias in MHC I-associated shaping of the adaptive immune system. Proc. Natl. Acad. Sci 115, 2168–2173 (2018). [PubMed: 29440397]
10. Shemesh O, Polak P, Lundin KEA, Sollid LM & Yaari G. Machine Learning Analysis of Naïve B-Cell Receptor Repertoires Stratifies Celiac Disease Patients and Controls. Front. Immunol 12, (2021).
11. Ostmeyer J, Christley S, Toby IT & Cowell LG Biophysicochemical motifs in T cell receptor sequences distinguish repertoires from tumor-infiltrating lymphocytes and adjacent healthy tissue. Cancer Res. canres.2292.2018 (2019) doi:10.1158/0008-5472.CAN-18-2292.
12. Beshnova D. et al. De novo prediction of cancer-associated T cell receptors for noninvasive cancer detection. Sci. Transl. Med 12, (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). [PubMed: 31101603]
14. Arnaout RA et al. The Future of Blood Testing Is the Immunome. Front. Immunol 12, (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 (2020) doi:10.1016/j.coisb.2020.10.010.
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). [PubMed: 28636592]
18. Glanville J. et al. Identifying specificity groups in the T cell receptor repertoire. Nature 547, 94–98 (2017). [PubMed: 28636589]
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, (2020).
20. Friedensohn S. et al. Convergent selection in antibody repertoires is revealed by deep learning. bioRxiv 2020.02.25.965673 (2020) doi:10.1101/2020.02.25.965673.
21. Mason DM et al. Deep learning enables therapeutic antibody optimization in mammalian cells by deciphering high-dimensional protein sequence space. bioRxiv 617860 (2019) doi:10.1101/617860.
22. Moris P. et al. Current challenges for unseen-epitope TCR interaction prediction and a new perspective derived from image classification. Brief. Bioinform (2020) doi:10.1093/bib/bbaa318.
23. Graves J. et al. A Review of Deep Learning Methods for Antibodies. Antibodies 9, (2020).

24. Narayanan H. et al. Machine Learning for Biologics: Opportunities for Protein Engineering, Developability, and Formulation. Trends Pharmacol. Sci 42, 151–165 (2021). [PubMed: 33500170]
25. Fischer DS, Wu Y, Schubert B. & Theis FJ Predicting antigen specificity of single T cells based on TCR CDR3 regions. Mol. Syst. Biol 16, e9416 (2020). [PubMed: 32779888]
26. Laustsen AH, Greiff V, Karatt-Vellatt A, Muyldermans S. & Jenkins TP Animal Immunization, in Vitro Display Technologies, and Machine Learning for Antibody Discovery. Trends Biotechnol. (2021) doi:10.1016/j.tibtech.2021.03.003.
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). [PubMed: 33764977]
28. Eraslan G, Avsec Ž, Gagneur J. & Theis FJ Deep learning: new computational modelling techniques for genomics. Nat. Rev. Genet 20, 389–403 (2019). [PubMed: 30971806]
29. Esteva A. et al. A guide to deep learning in healthcare. Nat. Med 25, 24–29 (2019). [PubMed: 30617335]
30. Vamathevan J. et al. Applications of machine learning in drug discovery and development. Nat. Rev. Drug Discov 1 (2019) doi:10.1038/s41573-019-0024-5.
31. Wainberg M, Merico D, Delong A. & Frey BJ Deep learning in biomedicine. Nat. Biotechnol 36, 829–838 (2018). [PubMed: 30188539]
32. Lythe G, Callard RE, Hoare RL & Molina-París C. How many TCR clonotypes does a body maintain? J. Theor. Biol 389, 214–224 (2016). [PubMed: 26546971]
33. Mora T. & Walczak AM How many different clonotypes do immune repertoires contain? Curr. Opin. Syst. Biol 18, 104–110 (2019).
34. Briney B, Inderbitzin A, Joyce C. & Burton DR Commonality despite exceptional diversity in the baseline human antibody repertoire. Nature 566, 393–397 (2019). [PubMed: 30664748]
35. Greiff V. et al. Learning the High-Dimensional Immunogenomic Features That Predict Public and Private Antibody Repertoires. J. Immunol ji1700594 (2017) doi:10.4049/jimmunol.1700594.
36. Parameswaran P. et al. Convergent Antibody Signatures in Human Dengue. Cell Host Microbe 13, 691–700 (2013). [PubMed: 23768493]
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). [PubMed: 25095879]
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, (2020).
40. Sidhom J-W, Larman HB, Pardoll DM & Baras AS DeepTCR is a deep learning framework for revealing sequence concepts within T-cell repertoires. Nat. Commun 12, 1605 (2021). [PubMed: 33707415]
41. Chen KM, Cofer EM, Zhou J. & Troyanskaya OG Selene: a PyTorch-based deep learning library for sequence data. Nat. Methods 16, 315 (2019). [PubMed: 30923381]
42. Kopp W, Monti R, Tamburrini A, Ohler U. & Akalin A. Deep learning for genomics using Janggu. Nat. Commun 11, 3488 (2020). [PubMed: 32661261]
43. Feng J. et al. Firmiana: towards a one-stop proteomic cloud platform for data processing and analysis. Nat. Biotechnol 35, 409–412 (2017). [PubMed: 28486446]
44. Gessulat S. et al. Prosit: proteome-wide prediction of peptide tandem mass spectra by deep learning. Nat. Methods 16, 509–518 (2019). [PubMed: 31133760]
45. Tomic A. et al. SIMON: Open-Source Knowledge Discovery Platform. Patterns 2, (2021).
46. Wu Z. et al. MoleculeNet: a benchmark for molecular machine learning. Chem. Sci 9, 513–530 (2018). [PubMed: 29629118]
47. Pedregosa F. et al. Scikit-learn: Machine Learning in Python. J. Mach. Learn. Res 12, 2825–2830 (2011).

48. Paszke A. et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. 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). [PubMed: 29790989]
50. Rubelt F. et al. Adaptive Immune Receptor Repertoire Community recommendations for sharing immune-repertoire sequencing data. Nat. Immunol 18, 1274–1278 (2017). [PubMed: 29144493]
51. Vander Heiden JA et al. AIRR Community Standardized Representations for Annotated Immune Repertoires. Front. Immunol 9, (2018).
52. Bolotin DA et al. MiXCR: software for comprehensive adaptive immunity profiling. Nat. Methods 12, 380–381 (2015). [PubMed: 25924071]
53. Gupta NT et al. Change-O: a toolkit for analyzing large-scale B cell immunoglobulin repertoire sequencing data. Bioinformatics 31, 3356–3358 (2015). [PubMed: 26069265]
54. Vander Heiden JA et al. pRESTO: a toolkit for processing high-throughput sequencing raw reads of lymphocyte receptor repertoires. Bioinformatics 30, 1930–1932 (2014). [PubMed: 24618469]
55. Nazarov Vadim, immunarch.bot & Eugene Rumynskiy. immunomind/immunarch: 0.6.5: Basic single-cell support. Zenodo. doi: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, (2020).
57. Corrie BD 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). [PubMed: 29944754]
58. Bagaev DV et al. VDJdb in 2019: database extension, new analysis infrastructure and a T-cell receptor motif compendium. Nucleic Acids Res. 48, D1057–D1062 (2020). [PubMed: 31588507]
59. Huang H, Wang C, Rubelt F, Scriba TJ & Davis MM Analyzing the Mycobacterium tuberculosis immune response by T-cell receptor clustering with GLIPH2 and genome-wide antigen screening. Nat. Biotechnol 1–9 (2020) doi:10.1038/s41587-020-0505-4. [PubMed: 31919444]
60. Zheng GXY et al. Massively parallel digital transcriptional profiling of single cells. Nat. Commun 8, 14049 (2017). [PubMed: 28091601]
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. Res. Sq (2020) doi:10.21203/rs.3.rs-51964/v1.
62. Weber CR et al. immuneSIM: tunable multi-feature simulation of B- and T-cell receptor repertoires for immunoinformatics benchmarking. Bioinformatics 36, 3594–3596 (2020). [PubMed: 32154832]
63. Marcou Q, Mora T. & Walczak AM High-throughput immune repertoire analysis with IGoR. Nat. Commun 9, 1–10 (2018). [PubMed: 29317637]
64. Sethna Z, Elhanati Y, Callan CG, Walczak AM & Mora T. OLGA: fast computation of generation probabilities of B- and T-cell receptor amino acid sequences and motifs. Bioinformatics 35, 2974–2981 (2019). [PubMed: 30657870]
65. FAIR principles for data stewardship. Nat. Genet 48, 343–343 (2016). [PubMed: 27023771]
66. Scott JK & 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). [PubMed: 33073065]
67. Breden F. et al. Reproducibility and Reuse of Adaptive Immune Receptor Repertoire Data. Front. Immunol 8, (2017).
68. Software with impact. Nat. Methods 11, 211–211 (2014). [PubMed: 24724161]
69. Goodman SN, Fanelli D. & Ioannidis JPA What does research reproducibility mean? Sci. Transl. Med 8, 341ps12–341ps12 (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. bioRxiv 2020.12.24.424260 (2020) doi:10.1101/2020.12.24.424260.

71. Abadi M. et al. TensorFlow: a system for large-scale machine learning. in Proceedings of the 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). [PubMed: 32952933]
73. Davidsen K. et al. Deep generative models for T cell receptor protein sequences. eLife 8, e46935 (2019). [PubMed: 31487240]
74. Bareinboim E. & Pearl J. Causal inference and the data-fusion problem. Proc. Natl. Acad. Sci 113, 7345–7352 (2016). [PubMed: 27382148]
75. Pavlovic M. et al. immuneML: v2.0.2. Zenodo. doi:10.5281/zenodo.5118741 (2021).
76. Fowler M. Domain-Specific Languages. (Addison-Wesley Professional, 2010).
77. Zenger M. '1.2 Characteristics of Extensibility Mechanisms.' Programming Language Abstractions for Extensible Software Components. (Lausanne: Swiss Federal Institute of Technology, 2004).
78. Pavlović M. immuneML use case 1: Replication of a published study inside immuneML. NIRD Research Data Archive. doi:10.11582/2021.00008 (2021).
79. Ploenzke MS & Irizarry RA Interpretable Convolution Methods for Learning Genomic Sequence Motifs. bioRxiv 411934 (2018) doi:10.1101/411934.
80. Heikkilä N. et al. Human thymic T cell repertoire is imprinted with strong convergence to shared sequences. Mol. Immunol 127, 112–123 (2020). [PubMed: 32961421]
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 doi: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 doi:10.11582/2021.00005 (2021).
83. Berman HM et al. The Protein Data Bank. Nucleic Acids Res. 28, 235–242 (2000). [PubMed: 10592235]
84. Schrödinger LLC. The PyMOL Molecular Graphics System. (2015).
85. immunoSEQ Analyzer | From Sequencing Data to Insights. immunoseq.com 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, (2015).
87. Řehůřek R. & Sojka P. Software Framework for Topic Modelling with Large Corpora. in Proceedings of the 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). [PubMed: 14597658]
89. 10x Genomics. A New Way of Exploring Immunity - Linking Highly Multiplexed Antigen Recognition to Immune Repertoire and Phenotype. https://www.10xgenomics.com/resources/application-notes/a-new-way-of-exploring-immunity-linking-highly-multiplexed-antigen-recognition-to-immune-repertoire-and-phenotype/.
90. Campos-Lima PO, Levitsky V, Imreh MP, Gavioli R. & Masucci MG 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). [PubMed: 9207000]
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). [PubMed: 28423320]
92. Grant EJ 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). [PubMed: 27645996]
93. Wang Z. et al. Clonally diverse CD38+HLA-DR+CD8+ T cells persist during fatal H7N9 disease. Nat. Commun 9, 824 (2018). [PubMed: 29483513]

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). [PubMed: 29997621]

95. Lehner PJ 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). [PubMed: 7807026]

Focus Box 1:

Getting started with immuneML

- Visit the project website at immuneml.uio.no. immuneML may be used (i) online via the Galaxy web interface (galaxy.immuneml.uio.no), (ii) through a Docker container, or (iii) 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 & 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 see all 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).

Focus Box 2:

How to contribute to immuneML

There exist multiple avenues for contributing 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 may be reported on the immuneML GitHub under “Issues” (github.com/uio-bmi/immuneML/issues).
- To improve or extend the immuneML platform, obtain the source code 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 may 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 back to the community, either by making their own versions public or by submitting their contributions as GitHub “pull requests” to the main immuneML codebase.

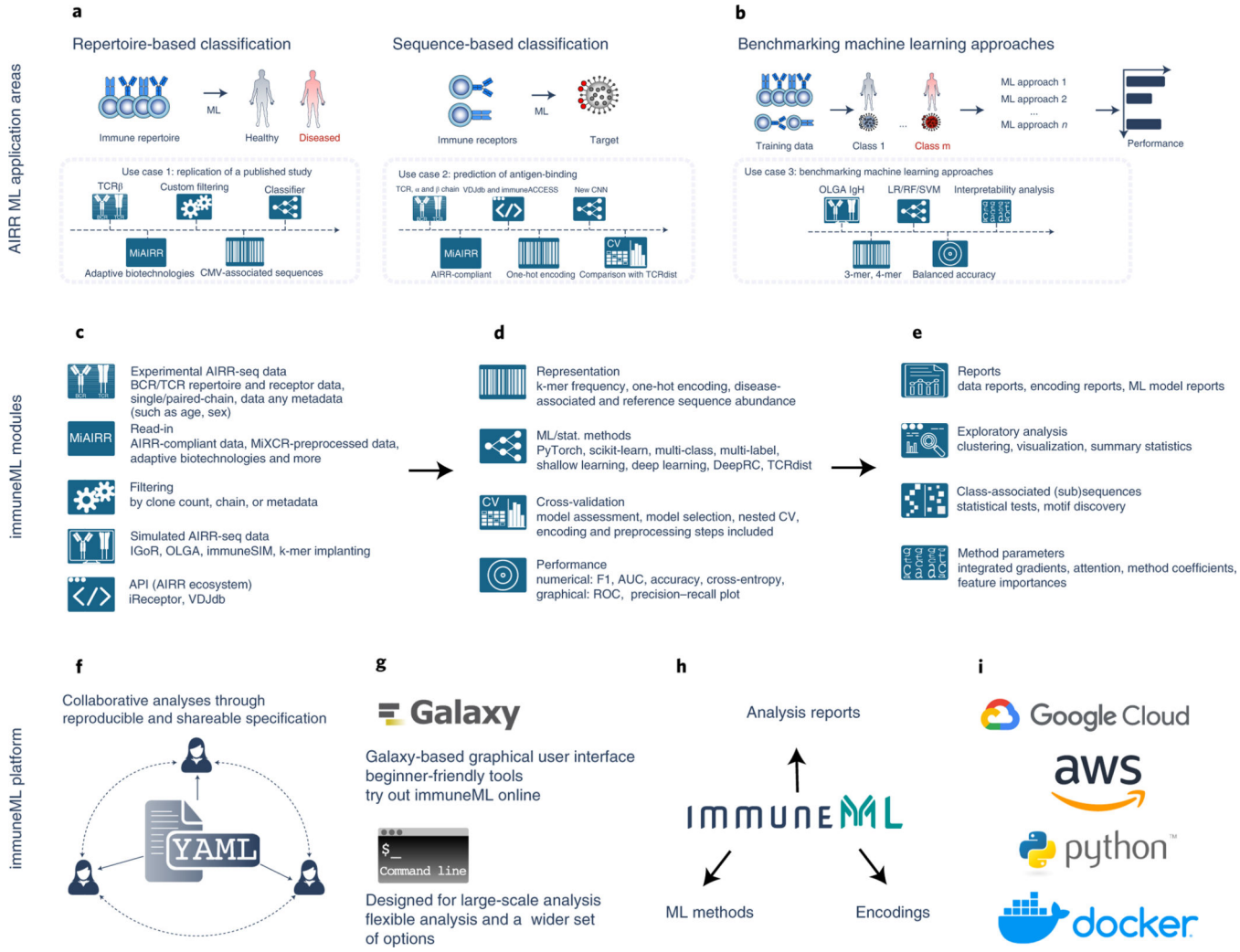

Figure 1 |. Overview of immuneML.

The main immuneML application areas are sequence- and repertoire-based prediction of AIRR with application to **(a)** immunodiagnostics and therapeutics research, as well as to **(b)** develop AIRR-based methods. We show three use cases belonging to these application areas. Use case 1: reproduction of the study by Emerson et al.[6] on repertoire classification, use case 2: extending the platform with a novel convolutional neural network (CNN) classifier for prediction of TCR-pMHC binding that allows paired-chain input, use case 3: benchmarking ML methods with respect to their ability to recover a sequence-implanted signal corresponding to the simulated immune event. The immuneML core is composed of three pillars, which are **(c)** AIRR-seq data input and filtering, **(d)** ML, and **(e)** Interpretability analysis. Each of these pillars has different modules that may be interconnected to build an immuneML workflow. **(f)** immuneML uses a specification file (YAML), which is customizable and allows full reproducibility and shareability with collaborators or the broader research community. An overview of how immuneML analyses can be specified is given in Supplementary Figure 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 (e.g., AWS, Google Cloud). Abbreviations: CMV (cytomegalovirus).

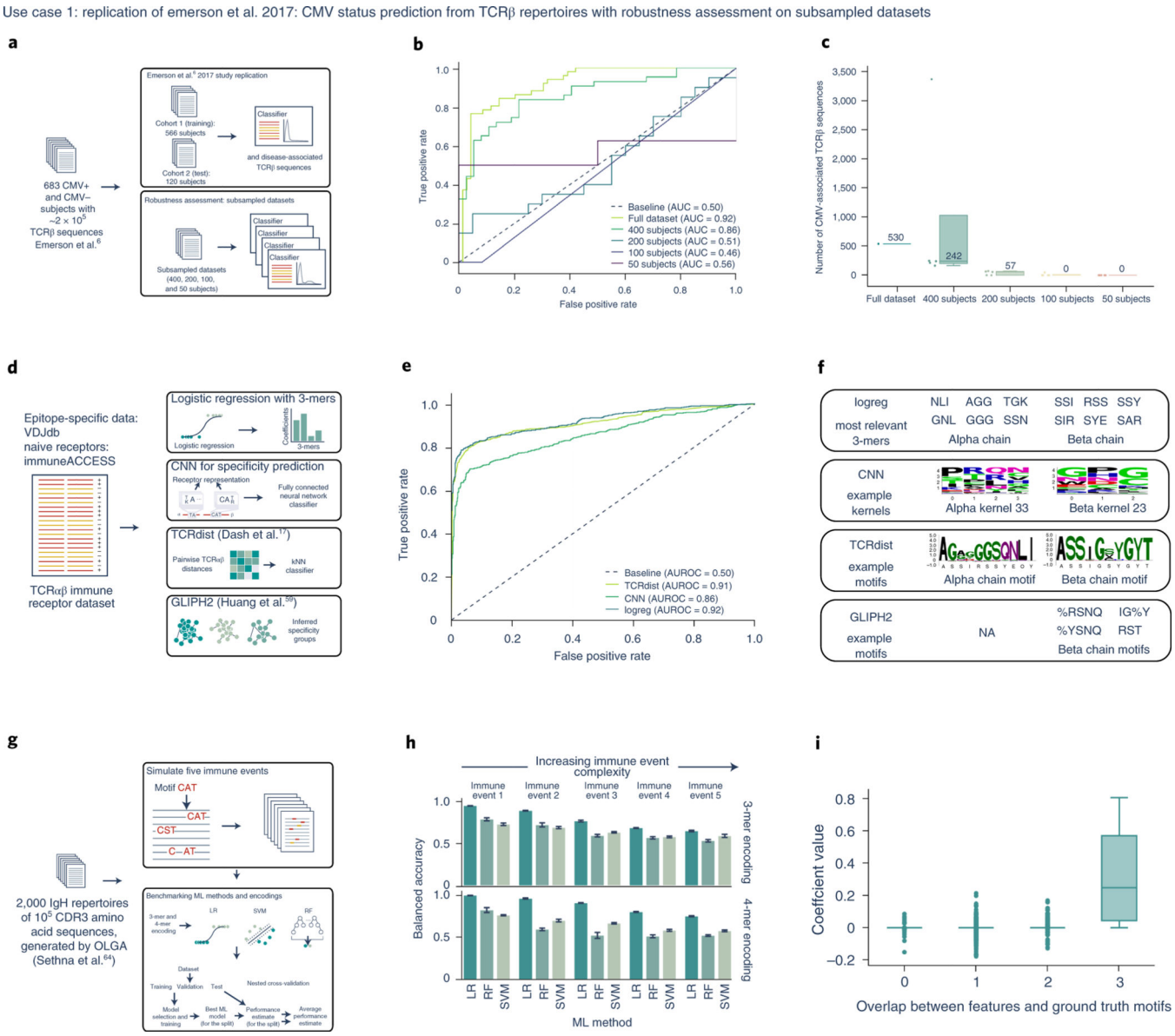

Figure 2 |. Use cases demonstrating ML model training, benchmarking, and platform extension. We showcase three use cases to exemplify immuneML usage. **(a-c)** Use case 1: Reproduction of a published study[6] where the task consisted in distinguishing between TCRβ repertoires from CMV (cytomegalovirus) positive and negative individuals, as well as the identification of TCRβ sequences that are associated with CMV status. In addition, we assessed the robustness of the respective statistical approach, measured by the predictive performance, as a function of decreasing dataset size. We show how a lower number of repertoires (400, 200, 100, and 50) leads to decreased prediction accuracy (AUROC: 0.86–0.46) and a lower number of CMV-associated TCRβ sequences (with almost none found in datasets of 100 and 50 subjects). **(d-f)** Use case 2: We developed a new ML method for antigen-specificity prediction on paired-chain T-cell receptor data using a convolutional neural network (CNN) architecture. The method separately detects motifs in paired chains and combines the motif scores corresponding to kernel activations to obtain

the receptor representation which is then used as input to a classifier. We compared the CNN method with the TCRdist-based k-nearest neighbor 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. We also show the motifs recovered by CNN, TCRdist, and GLIPH2 among the epitope-specific sequences. **(g-i)** Use case 3: We show how ground-truth synthetic data may be used to benchmark AIRR ML methods. The dataset consists of 2000 immune repertoires generated by OLGA[64]. Using immuneML, five immune events of increasing complexity are simulated by implanting synthetic signals into the OLGA-generated repertoires. This dataset is subsequently used to benchmark three different ML methods (logistic regression (LR), support vector machine (SVM), and random forest (RF)) in combination with two encodings (3-mer and 4-mer encoding) inside immuneML, showing the classification performance with standard deviation that drops as the immune event complexity increases. The quality of the ML models was further assessed by comparing the feature coefficient sizes with how well these features represent the ground-truth signals. This revealed that models with a good classification performance were indeed able to recover the ground-truth signals. Error bars in (h) represent standard deviation.