Restoring data balance via generative models of T-cell receptors for antigen-binding prediction

Emanuele Loffredo[a,1], **Mauro Pastore**[a,1], **Simona Cocco**[a,2], **and Rémi Monasson**[a,2]

This manuscript was compiled on July 10, 2024

Unveiling the specificity in T-cell-receptor and antigen recognition represents a major step to understand the immune system response. Many supervised machine learning approaches have been designed to build sequence-based predictive models of such specificity using binding and non-binding examples of data. Due to the presence of few specific and many non-specific T-cell receptors for each antigen, available datasets are heavily imbalanced and make the goal of achieving solid predictive performances very challenging. Here, we propose to restore data balance through data augmentation using generative unsupervised models. We then use these augmented data to train supervised models for prediction of peptide-specific T-cell receptors and binding pairs of peptide and T-cell receptors sequences. We show that our pipeline yields increased performance in terms of T-cell receptors specificity prediction tasks. More broadly, our work provides a general framework to restore balance in computational problems involving biological sequence data.

adaptive immune system | antigen TCR binding predictions | data augmentation | data imbalance | generative models | deep neural networks

Significance Statement

The adaptive immune system carries a diverse set of T-cell receptors capable of recognizing pathogens and protect the host from diseases. Predicting whether a receptor binds a pathogenic peptide is a fundamental computational problem, made difficult by the imbalance in available data: relatively few binding pairs are known compared to all possible pairs of receptors and peptides. Here, we propose to mitigate this imbalance problem by generating putative binding pairs through data augmentation machine-learning methods. We show that these extra data helps training binding prediction models and improves their performances. Our framework for sequence data augmentation is generic and could be applied to other biological computational problems.

1. Introduction

Cytotoxicity T lymphocytes (T–cells) play a crucial role in the adaptive immune response of organisms against pathogens and/or malfunctioning cells (1). Short pathogen protein regions (peptide antigens) interact with the Major Histocompatibility Complex proteins (MHC) and form peptide-MHC epitopes (pMHC). Binding of CD8+ T-cell receptor (TCR) with pMHC enables the killer machinery against such pathogen. Given the high specificity of the interaction, TCRs only bind a limited number of presented pMHC. Therefore, achieving a reliable prediction of TCR-pMHC binding represents a major goal in the field, in particular for the development of vaccines and the improvement of personalized immunotherapies.

Over the recent years, much progress in this direction has been made with computational approaches (2–6), benefiting both from the strong power of machine learning (ML) methods and the large-scale amount of experimentally tested data. Such works typically use the sequences of the Complementarity–Determining Region-3 beta (CDR3β) and alpha (CDR3α) chains paired with peptide sequences to reveal the TCR-pMHC binding affinity. The CDR3 region is the most variable one in TCRs and is recognized to be the major actor influencing TCR specificity for peptide binding. Though recent works have shown that use of both α and β chains leads to better predictions (7, 8), many works still focus on β chains solely, because they primarily drive the immune response (9) and are more abundant in most databases.

Predicting TCRs specificity is a computationally challenging problem for several reasons. First, CDR3β sequences binding different target epitopes exhibit very strong similarities. As an illustration, we show in Figure 1a the t-distributed Stochastic Neighbor Embedding (tSNE) visualization of CDR3β sequences known to bind three chosen epitopes and of other, unlabelled CDR3β sequences: the absence of well separated epitope-related clusters make specificity prediction highly non trivial. Second, in order to produce accurate predictions, ML predictive models are trained on labelled (experimentally tested) TCR-pMHC sequence data through supervised learning. Public databases containing TCR specificity, such as IEDB (10), VDJdb (11, 12), Mc-PAS TCR (13) and PIRD (14), mostly include limited amount of TCR sequences with positive interaction, *i.e.* known to bind some peptides of interest. Considerable efforts have been accomplished to build *ad hoc* negative sequence datasets, for instance by mismatch pairing and/or taking bulk (unlabelled) data from healthy donors. While the produced negative data may suffer from potential biases affecting predictions (15), they exceed in quantity, by several orders of magnitude, the amount of experimentally tested positive binding pairs.

Author affiliations: [a]Laboratory of Physics of the Ecole Normale Supérieure, CNRS UMR 8023 and PSL Research, Sorbonne Université, 24 rue Lhomond, 75005 Paris, France

E.L., M.P., S.C., R.M., designed research; E.L., M.P., S.C., R.M., performed research; E.L., M.P. analyzed data; E.L., M.P., S.C., R.M., wrote the paper.

No competing interest is declared.

[1]E.L. contributed equally to this work with M.P.
[2]S.C. contributed equally to this work with R.M.

As a result, binding prediction models are trained from datasets with strong imbalance between the positive and negative classes. Class imbalance is a widespread problem affecting both bio- and non bio-related sources of data and is notably an Achilles' heel in applying machine learning methods to achieve reliable predictions (see, for example, (16–19)). Informally speaking, models adapt to the most represented class in the data and are less accurate in capturing the features of under represented examples, degrading the quality of predictions. The effect of class imbalance is visible in Figure 1b, which show two metrics assessing the capability of predicting binding of CDR3β sequences to one fixed epitope, the unbiased balanced accuracy (ACC), *i.e.* the accuracy evaluated on a balanced test set, and the area under the receiver operating curve (AUC), an integrated measure over varying classification thresholds. Both ACC and AUC reach high values when the numbers of binding and background sequences in the training set are comparable, and drop for strongly unbalanced datasets. The strong impact of class imbalance to the performances of inference methods predicting TCR-peptide interactions has been recently assessed in literature (20).

While achieving balance in the dataset could be easily obtained by limiting the number of negative data, it is expected that discarding data might not be optimal in general, and better balancing strategies are needed. The present work proposes to restore balance with a mixed strategy, combining data removal (subsampling of the negative class) and, crucially, data augmentation (oversampling of the positive class), see Figure 1c. We implement this mixed strategy for both

(i) *peptide-specific* models, which are trained on one or more selected epitopes and for which the peptide sequence is only used as a label during the training;

(ii) *pan-specific* models, where combinations of peptides and TCR sequences are presented during training, together with binary labels expressing if the peptide-TCR pairs are binding or not.

Though pan-specific models are harder to build as they require more data than their peptide-specific counterparts, they can, in principle, leverage the diversity of the peptide space to capture the underlying features of TCR-peptide interactions and potentially recognize binding to new, unseen antigens. We explore this last possibility by studying the out-of-sample performances of our predictive models.

2. Learning pipeline

We hereafter propose a generative machine-learning framework to mitigate the scarcity of peptide-specific CDR3β sequences available in order to enhance TCR specificity predictions. The approach is not only limited to the setting examined here but can be applied to any case where one aims at discriminating an *under-represented* class of sequence data with a supervised learning algorithm.

The learning pipeline proposed in this work is fully presented in Figure 2 for both peptide- and pan-specific cases and consists of two steps. First, data balance is restored by randomly subsampling the negative data class and by augmenting the positive class of peptide CDR3β sequences through data generation. To do so, we learn an unsupervised model over the limited number of peptide-specific CDR3β sequences and enlarge the positive class by generating surrogate sequences. We use two unsupervised generative models, namely

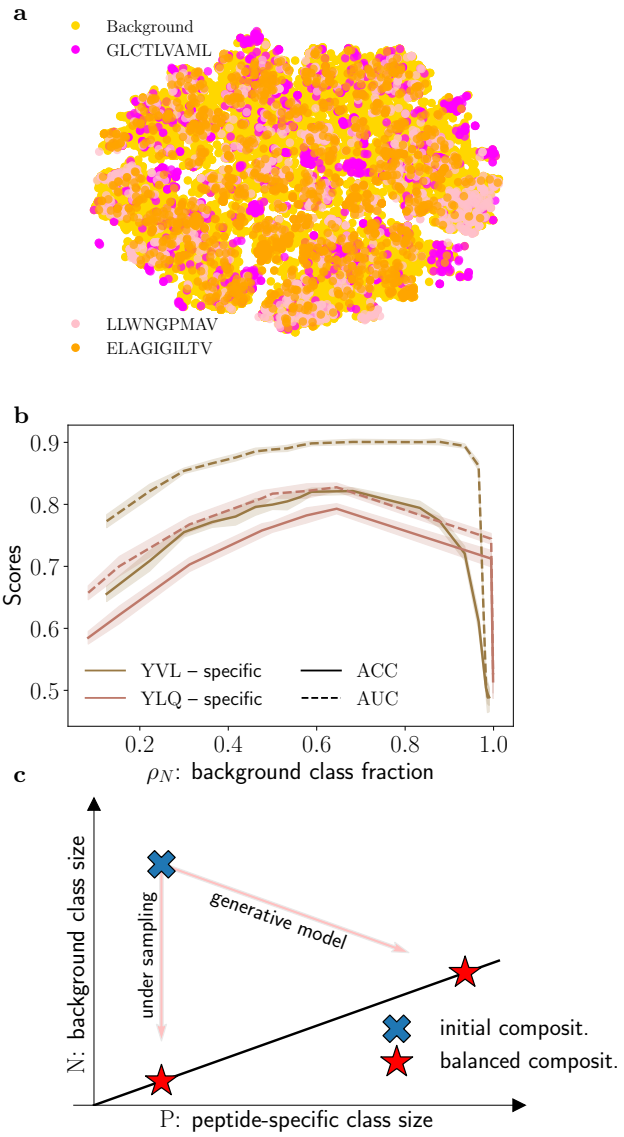


Fig. 1. Graphical representation of imbalanced learning on TCR specificity predictions. a) tSNE visualization of the top three most abundant epitopes in the aggregated full dataset (see Methods) (GLCTLVAML, LLWNGPMAV, ELAGIGILTV), together with unlabelled CDR3β sequences (Background). The high overlap among data points demonstrates the challenge in deriving accurate specificity predictions. Notice that background sequences span the whole feature space making identification of key binding properties hard. **b)** Accuracy and AUC scores for a predictive model trained to distinguish YVLDHLIVV- and YLQPRTFLL-specific CDR3β sequences from bulk CDR3βs as a function of the fraction ρ_N of background data in the training set. In practice we fix the class size of peptide-specific sequences and vary the size of the background sequences class to change ρ_N. Performances (evaluated on a balanced test set) are optimal when the two class sizes are of roughly equal sizes, *i.e.* when $\rho_N \simeq 0.5$. **c)** Graphical visualization of the imbalanced composition of TCRs datasets, in a two-class setting where P, N represent the class sizes. Our work proposes to restore class balance (*i.e.* $P = N$, straight line) by introducing generative model power to sample new sequences compatible with the positive class, for which few data are experimentally available.

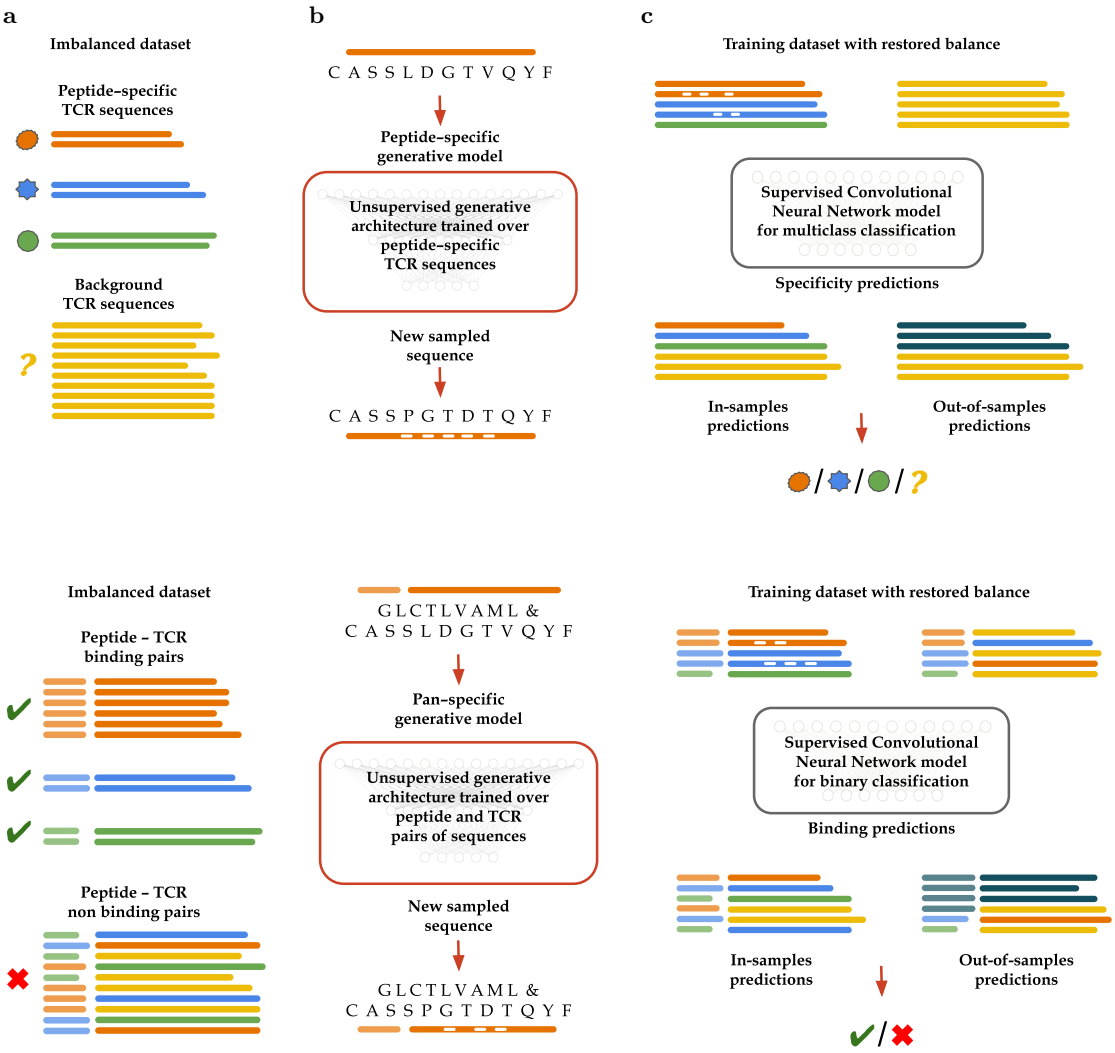


Fig. 2. Learning pipelines for peptide-specific (top) and pan-specific (bottom) models. Top, peptide-specific model. Column (a): Data consists of few CDR3β sequences, known to bind some epitopes (colored symbols and segments) and of many 'negative' sequences (yellow). Column (b): A generative model is trained over peptide-specific CDR3β sequences, here, corresponding to the orange epitope. After training, Gibbs sampling of the inferred probability landscape allows us to generate putative peptide-specific sequences. Column (c): A supervised CNN architecture is trained over (natural and generated) peptide-specific CDR3βs and background CDR3βs; after learning, the network is used as predictive model for TCR specificity over in- and out-of-sample (black sequences) test data. Bottom, Pan-specific models. Column (a): Compared to the pipeline above, input data are joint sequences of peptides (left, lighter color) and of TCR (right). Background sequences are obtained through mismatch pairing. Column (b): The generative models produce putative binding pairs of peptide and TCR sequences. Column (c): Supervised classifier trained to carry out TCR-epitope binding predictions.

- Restricted Boltzmann Machines (RBMs), two-layers architectures extracting latent features from data;
- Bidirectional Encoder Representation Transformer (BERT)-like architectures trained over CDR3β sequences to learn their *grammatical structure*.

For peptide-specific predictions we separately train a model for each epitope, while for pan-specific predictions a collective model encompassing all the epitopes receives as input peptide and CDR3β data separated by a "&" token (*e.g.* GILGFVLT & CASSLDGTVQYF). The idea of using unsupervised generative model for data augmentation to mitigate imbalance or data scarcity has been recently proposed in a variety of settings (19, 21–26), as well as already successfully applied to biological-sequences related tasks, *e.g.* (27).

Second, we use the natural and generated data to train a model for TCR-epitope binding predictions. Models existing so far range from random forests (28, 29) to neural network architectures of different complexities, *e.g.* convolutional neural networks (8, 30), long-short term memory networks and autoencoders (31, 32); unsupervised algorithms have also been employed, such as SONIA (33) and its more precise variant soNNia (34), diffRBM (35), which implements transfer learning within Restricted Boltzmann Machines, and, in the context of Large Language Models, Transformers-based approaches (36–38).

Hereafter we resort to a one-dimensional convolutional neural network (CNN) architecture, trained over positive (natural and generated) and negative sequences – once balance has been restored between classes. The architectures for

peptide- and pan-specific models are slightly different, see Figure 2 and are detailed in Methods. Last of all, the predictive power of the model is tested over its ability to discriminate positive against negative CDR3β sequences (the in-sample test set) or against other new peptide-specific CDR3β (the out-of-sample test set).

3. Peptides-specific models for TCR binding predictions

Sources of imbalance for peptide-specific models. We collected from the IEDB database (10) the sequences of CDR3β with specificity to a set of different epitopes, listed in Table 1. Background bulk repertoires correspond to CDR3β sequences sampled from cohorts of healthy donors and are randomly sampled from the entire datasets in (39, 40). As anticipated in the introduction, the assembled dataset suffers from imbalance between peptide-specific CDR3βs and the large abundance of bulk unlabelled data. In addition, some epitopes (*e.g.* GLCTLVAML, LLWNGPMAV, YLQPRTFLL) have thousands of known CDR3β binders, while others have hundreds of positively tested CDR3βs only, which results in further imbalance between the epitope-associated classes.

To avoid misleading predictions due to imbalance biases, we balance the training set in such a way that all classes are equally represented (Figure 2). To evaluate performances in a fair way, we consider different metrics over a balanced test set containing all classes in the same proportion in order to factor out any source of bias. We use ACC and AUC metrics to assess our performances (Methods).

Epitope	Sequence
$M1_{58\text{-}66}$ from influenza A virus	GILGFVFTL
peptide from Yellow Fever virus	LLWNGPMAV
$BMLF1_{280\text{-}288}$ from Epstein-Barr virus	GLCTLVAML
peptide from Spike protein S269 of SARS-CoV-2	YLQPRTFLL
$MART1_{27\text{-}35}$ from Melanoma cancer	ELAGIGILTV
SEC24A from Melanoma cancer	FLYNLLTRV
TKT-R438W from Melanoma cancer	AMFWSVPTV
pp50 from human cytomegalovirus CMV	VTEHDTLLY
$NS3_{1436\text{-}1444}$ from Hepatitis C virus HCV	ATDALMTGY

Table 1. List of the epitopes selected to collect CDR3β specific sequences in order to form the database used in the analysis of Section 3.

In-sample predictions benefit from data augmentation with generative models. We present below results for a case where the predictive model has to learn three different classes of peptide-specific receptors (labelled with p_1, p_2 and p_3) and bulk receptors (referred to as b). To assess if an unsupervised model that generates CDR3β sequences to augment the peptide-specific classes size can yield better performances than naive undersampling of each CDR3β class down to the lowest available class size, we consider two different strategies to restore balance in the dataset:

- A first protocol, in which data are unaltered and each class size $\mathcal{D}$ is given by
$$\mathcal{D} = \min\left[\mathcal{D}(p_1), \mathcal{D}(p_2), \mathcal{D}(p_3)\right]; \qquad [1]$$
i.e. data points in over-represented classes are randomly under sampled down to the common size $\mathcal{D}$.
- A second protocol, in which small-size classes are augmented using the generative model pipeline up to a target size $\mathcal{G}$ common to all classes, with $\mathcal{G} \leq 10\,\mathcal{D}$ for computational reasons.

Notice that, for both protocols, the bulk class size $\mathcal{D}(b)$, which is orders of magnitude larger than any $\mathcal{D}(p_i)$, is randomly under sampled to match the final common size of the positive data. We use as generative framework both a RBM- and a BERT-based sampling strategy, to assess the effects of balancing through data augmentation regardless of the generative model. These data are then used to train our classifier. Notice that the last layer of the CNN architecture, which outputs the binding predictions, is designed to have as much units as the number of classes, *i.e.* of peptide labels plus the background label, with softmax activation function. This allows us to carry out multi-class classification, as the network is able to predict specificity towards more than one target.

Figure 3 shows results for multiple experiments related to different triplets of peptides-specific CDR3β sequences, confirming that the quality of predictions increases with the pipeline described in Figure 2a. In particular, the gain in performance is larger for multiclass experiments involving triplets of peptide-specific CDR3β where one (or more) class contains few sequences, *e.g.* $\mathcal{D}(p_1) \ll \mathcal{D}(p_2), \mathcal{D}(p_3)$. For example, experimental data for the epitopes AMFWSVPTV, VTEHDTLLY and GLCTLVAML have relative sizes $1.4\% - 4.1\% - 94.5\%$, which introduces a strong bias towards the GLCTLVAML-specific CDR3βs. Restoring balance by generating new AMFWSVPTV-specific CDR3β sequences, we are able to obtain a 20% performance increase compared to restoring balance by undersampling only. This gain is milder when pepitde-specific classes have more sequences and are more balanced among each other, as the case of epitopes ELAGIGILTV (9%), LLWNGPMAV (21%) and GLCTLVAML (70%) shows. Relative abundances of the initial dataset for the combinations reported in Figure 3 can be obtained from the absolute class sizes reported in Table S2. Notice that the gain in performance resulting from balancing is generally more marked for ACC than for AUC, as already seen in the wider plateau of the latter in Figure 1b. Being AUC an average measure of performance, it is however less informative about the quality of prediction of the best-threshold classifier than ACC.

Notice that, to obtain the increase of performances we observe in Figure 3, it is crucial to balance the dataset with generative models powerful enough to capture non-trivial features in the distribution of the data, a task that both RBMs and BERT-like models are able to do; for more on this aspect, see SI Section S.4.

Performances can also be assessed in an out-of-sample setting, in which the test set includes sequences binding to unseen epitopes. As expected, model accuracy strongly depends on the similarity between the out-of-sample and training data distributions, see SI Section S.2.

4. Pan-specific models for TCR-epitope binding predictions

Pan-specific models, which take as inputs both the CDR3β and the peptide sequences, have gained interest, as they offer

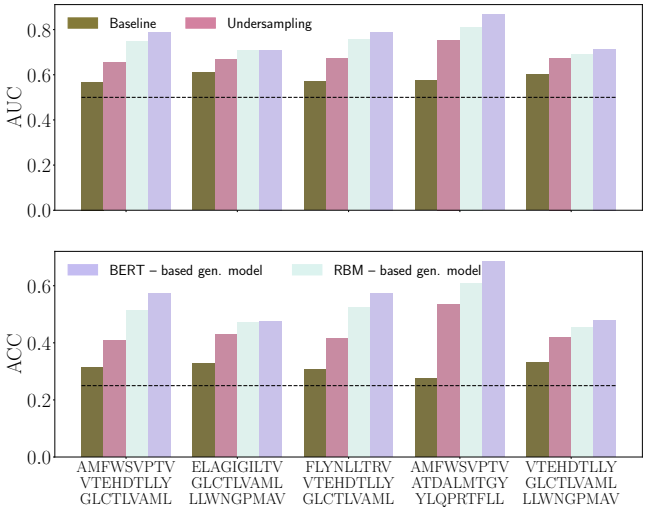


Fig. 3. In-sample performances of peptide-specific models. AUC and ACC scores of the predictive models for a multiclass classification task involving three peptides-specific sets of CDR3β and background CDR3β sequences, evaluated over a balanced test set of sequences of the same classes (in-sample case). We compare performances for different training dataset, whose balance is restored through undersampling solely, or through generating new CDR3β sequences via an RBM- or BERT-based generative architecture. The baseline scores refer to an imbalanced training dataset, whose composition can be derived from the class sizes of each epitope as reported in Table S2; 250,000 background CDR3βs are used. Results confirm the benefit of both restoring balance in the training dataset and enlarging the peptide-specific CDR3β space through generative models. Dashed black lines indicate random performance levels.

the possibility to identify binding patterns across different epitopes, and could potentially be used for binding predictions with rare or even novel peptides.

Sources of imbalance for pan-specific models. The dataset used to train and test pan-specific models is taken from (41), and contains both positive and negative interacting pairs of CDR3β and peptide sequences spanning 118 different epitopes. Natural data are taken from publicly available sources, namely IEDB, VDJdb, McPAS-TCR, MIRA (42, 43), and from 10X Genomics assays. Among them we only retain peptides with at least 150 positive binder CDR3β representatives to obtain a sufficiently large dataset for training both the generative and the classifier models. This results in a total of 122,334 experimentally tested pairs of interacting CDR3βs and peptide sequences, and a collection of 405,176 non-interacting sequence pairs. These negative examples consist for a minor part of experimentally assessed ones, while a large part of them is obtained by random mismatching of CDR3β and peptide sequences from the positive class; we further enlarge the negative set by pairing peptides with CDR3βs randomly chosen from the previous bulk repertoire. The resulting dataset is plagued with two sources of imbalance:

(i) positively interacting pairs are strongly under-represented compared to negatively interacting pairs – we refer to this as *class-level* imbalance;

(ii) within the positive class, few peptides are strongly over-represented compared to others – we refer to this as *group-level* imbalance (44, 45).

Notice that there is possibly group imbalance within the negative class based on the different ways to assemble negative sequence pairs through mismatches. We qualitatively observe that this imbalance has minor effects on performances and ignore it in the following.

Comparison of peptide- vs pan-specific generative models. Data augmentation can be done in two ways.

(i) By analogy with the peptide-specific case studied above, we should design a generative model that is trained on varied peptide and CDR3β sequences and produces new pairs. Intuitively, as the number of distinct CDR3βs dwarfs the one of epitopes, we expect a pan-specific generative model to first learn how to cluster groups of TCR sequences based on their epitope labels, then to learn the CDR3βs distribution within each group. However, such a model could suffer from group imbalance, and would learn effectively only the most-represented instances of epitopes and TCRs.

(ii) An alternative approach consists in training, separately, a generative model for each peptide-specific class of CDR3βs, so that the group imbalance present in the dataset is completely factored out. This second approach is however computationally demanding, as its running time increases linearly with the number of epitopes. In addition, overfitting could be an issue for peptide-specific groups with very little data.

As for the classifier, we closely follow the CNN architecture used in (8), where two convolutional layers process separately the CDR3β and the peptide sequence: the resulting feature vectors are then concatenated to ultimately output the positive or negative binding prediction, see Methods.

To assess the impact of group imbalance and the performance of the two approaches above, we first build an auxiliary dataset from the pan-specific dataset in (41) by retaining only CDR3βs binding to five selected epitopes (AMFWSVPTV, ELAGIGILTV, FLYNLLTRV, GLCTLVAML, LLWNGPMAV), for a total of $9,000$ datapoints. Figure 4a reports AUC and ACC scores obtained over each peptide and over the aggregate dataset comprising all five peptide-specific groups when CDR3β sequence data have been balanced with one global pan-specific generative model or with multiple peptide-specific ones. In the latter case, we have enlarged the CDR3β sequence space of the two most under represented group epitopes – AMFWSVPTV and ELAGIGILTV. We observe that predictive performances are comparable between the two approaches, suggesting that the pan-specific generative approach is not heavily impacted by group imbalance, and is capable of adequately clustering and modeling each peptide-associated group of TCR sequences.

This robustness stems from our sampling protocol, in which the generative model is carefully initialized with training sequences (Figure 2). Upon sampling, the landscape defined by the inferred probability distribution is explored in the proximity of peptide-specific region associated to the initial sequence. This procedure prevents the generative model from jumping towards other peptide-specific groups, which can have much stronger overall weight due to group imbalance. Gibbs sampling schemes that start from randomly chosen sequence pairs preferably falls within such groups, *e.g.* peptides with less than 250 binders in the training dataset are on average generated four times less frequently than peptides with more.

TCR-peptide binding predictions benefit from data augmentation with generative models. Based on the results above, we now probe the benefit of restoring balance through the use of a pan-specific generative model, which we train over the full dataset (41), including abundant peptide-specific groups of CDR3βs. We then generate new peptide-specific sequences through Gibbs sampling initialized with natural sequences. The model takes as input peptide and CDR3β pairs of sequences and proposes random mutations across the sequence pair to sample new ones. We observe that only a small fraction ($\sim$ 3%) of generated data shows mutations across the peptide sequence: data augmentation overwhelmingly consists in the production of new CDR3β attached to the same peptides as in the training dataset.

An important hyperparameter is the size of peptide-associated groups after generation of new TCR sequences. In practice, we decide to set a threshold group size $\mathcal{G}$ for data augmentation, above which peptide-specific classes are deemed as *sufficiently populated* and are thus not enlarged through the generative model. Groups having less than $\mathcal{G}$ datapoints are all enlarged up to the threshold value. $\mathcal{G}$ cannot be chosen arbitrarily large as the generation of new data points is computationally expensive, and limited by the quality of training data and of the unsupervised model.

Once all positive data have been generated, we restore balance at class level by undersampling negative peptide-CDR3β pairs. Undersampling is randomly performed within each group, so that we are guaranteed that the dataset contains, on average, equal numbers of positive and negative pairs for each group, a requirement that we have observed to be important performance-wise.

We show in Figure 4b that data generation leads to increase of performance for a vast majority of the peptides involved in the dataset. Even peptide-specific groups that have not been directly enlarged (red circles) can show a gain, arguably because the CNN architecture is leveraging the information of generated peptide and CDR3β pairs to transfer learning across all the different peptide groups. The scores also show that the generative model is particularly promising for heavily under represented groups (small triangles), with performance gains up to 20%.

Out-of-sample TCR-peptide binding performances. We now ask whether our pan-specific model generalizes well on unseen epitopes, *i.e.* is able to capture the general properties underlying the binding process of receptors to antigens. Out-of-sample analysis, in which test data is sampled from an external distribution, can be very challenging as the model has not explicitly learnt any feature of those data. We expect performances to drop consistently, depending on how much the out-of-sample data are distant from seen data in feature space. As out-of-sample test set we aggregate all the peptide-specific CDR3β sequences excluded from the training set during data pre-processing because they were strongly under represented (< 150 sequences per group); yet, we retain only group that have > 20 CDR3β sequences to have significant statistics.

Performances for this hard out-of-sample setting are unsurprisingly worse than in the in-sample case, in agreement with previous studies (46). Nonetheless, we observe a clear correlation between the scores and the similarity between the unseen epitopes and the ones in the training data, see Table 2. To study the dependence of performance upon this similarity, we cluster the CDR3β sequences based on the Levenshtein distance of their target epitope to the closest epitope in the dataset. For large Levenshtein distance (10 - 11), our classifier behaves not better than a random one.

Improving out-of-sample classification is crucial for the task of TCR-peptide binding predictions, as information about the binding properties of new antigens are rarely available. Here, we explore the possibility to predict antigen binding towards unseen epitopes, exploiting the pan-specific generative model (trained on the in-sample dataset) and a partial knowledge of the out-of-sample dataset (the sequence of the new epitope). In practice, we use the generative architecture to sample CDR3βs binding to the target epitope; then, we include these data in the training set and test the performance on the natural out-of-sample binder sequence data.

As a proof of concept, we evaluated this procedure on 8 different epitopes, whose natural CDR3β sequence have to be distinguished from other CDR3β with a different out-of-sample specificity (see Table 3). We observe an improvement of performances for epitopes at small distance from their in-sample counterparts.

Minimal model of out-of-sample classification of TCR-peptide binding pairs. To better understand how the properties of out-of-sample data impact binding predictions, we repeat this analysis on controlled, synthetic data. We resort to dimeric lattice proteins (LPs) compounds, whose native folds are shown in Figure 5a. LPs are synthetic proteins defined as self-avoiding paths over a $3 \times 3 \times 3$ lattice cube, whose vertices carry the 27 amino acids. The two LP structures in the dimer represent, in order, the epitope and the TCR.

Following (47), we build dimeric LPs starting from single monomers running Monte Carlo (MC) evolution, and collect sequence data for multiple dimeric LPs. Spanning multiple dimeric structures, by changing the conformation of the self-avoiding paths on the lattice, allowed us to model different group specificities; details in Methods. We collect MSAs of binding and non-binding pairs constituting the two classes in our dataset, and balanced both at class and group levels. A CNN classifier similar to the one introduced for natural data above is then trained over these data and reaches perfect classification in discriminating binding vs non-binding pairs – regardless of the group specificity, showing the simplicity of this in-sample task.

To assess out-of-sample performances we produce an additional dataset of binding pairs as follows. For each group of dimers in the training dataset, we collect sequence data corresponding to its closest dimer, *i.e.* with highest structural similarity. This dataset is referred to as *close out-of-sample*, as we expect it to be very close to the training data in the feature space of the classifier. Similarly, we repeat the procedure with randomly picked dimers, which will share few similarity with the dimeric structures defining the training data; we label such data as *not close out-of-sample*. The structural similarity based on the ground truth folding scores is reported in Figure 5b: for all in-sample binding pairs sequences we plot scores for the native, for close and not close dimer structures; the orange distribution is effectively closer to the green one than the blue one, confirming a stronger structural similarity.

In Figure 5c we report the tSNE 2d projections of the embeddings of the supervised architecture trained over the binary classification task for binding (green) versus non-binding (red) pairs of sequences, which allows us to visualize the neat decision boundary, giving very high ACC = 0.99. Within this feature space, on top of green and red points, we project out-of-sample data points for the close and not-close cases. As expected, the latter is harder to classify as they share less features with the training data of the model (ACC = 0.98 and ACC = 0.82, respectively). Therefore, even in the framework of artificial data, we observe that out-of-sample predictive performances depend on the degree of similarity with the in-sample dataset.

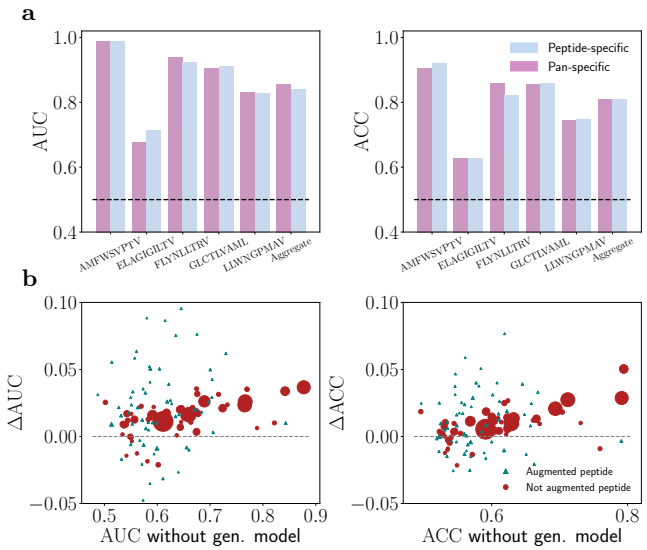


Fig. 4. Pan-specific predictive model results. **a)** TCR-peptide binding predictions with multiple peptide-specific generative models trained separately and with one pan-specific generative model trained over the entire dataset. We observe comparable performances for the AMFWSVPTV enlarged space and slightly worse performances with the pan-specific approach for the ELAGIGILTV one. The scores over the aggregate dataset are slightly better with the pan-specific generative approach. Notice that, through the peptide-specific generative architecture is equivalent to the one in the previous section, scores over similar epitopes differ from fig. 3 as the supervised architecture is pan-specific and performs binary classification. **b)** Differences in AUC and ACC scores when balance is achieved through undersampling of data only (no generation) or with data augmentation too. In the latter case, new CDR3β sequences were generated for under-represented groups of peptides only (triangular dots). Dot sizes are proportional to the raw group size of natural sequence pairs in the dataset. Here, $\mathcal{G} = 400$ (for more details on the choice of this threshold, see SI Section S.3).

5. Discussion

In this work we have introduced a framework that combines the use of unsupervised and supervised computational approaches to achieve reliable predictions of TCR specificities and TCR-epitope binding properties. Our pipeline relies on a two-step procedure, where we first leverage unsupervised network architectures to learn the probability distribution of peptide-specific CDR3β sequences and then use them to generate new informative sequence data. Second, the generated data allow us to train supervised model for the final predictive task over balanced datasets, avoiding biases induced by class imbalance. We emphasize that restoring balance within the dataset by augmenting the size of the

Dist.	AUC	σ(AUC)	ACC	σ(ACC)
1-3	0.68	0.05	0.65	0.05
4-6	0.63	0.10	0.61	0.06
7-9	0.62	0.09	0.60	0.07
10-11	0.49	0.20	0.52	0.11

Table 2. Pan-specific prediction scores for out-of-sample tasks. The AUC (second column) and ACC (fourth column) scores are averaged across many out-of-sample epitopes, with balance restored through pan-specific generation. Out-of-sample CDR3β are grouped according to the Levenshtein distance of their associated epitope to the closest one in the training dataset (first column). Predictions worsen as peptides get further away from the ones in the training dataset. The third and fifth columns show the standard deviations of the scores.

Peptide	d [in]	AUC [u]	AUC	ACC [u]	ACC
LPRWYFYYL	1	0.61	0.72	0.58	0.64
KRWIIMGLNK	1	0.48	0.54	0.50	0.52
QYIKWPWYI	2	0.53	0.59	0.52	0.57
GTSGSPIIDK	2	0.56	0.60	0.53	0.57
KLSALGINAV	4	0.34	0.45	0.39	0.48
IMDQVPFSV	4	0.56	0.49	0.55	0.50

Table 3. Out-of-sample performances evaluated with AUC and ACC metrics across a test set composed of wild type binders to the target epitope and CDR3β sequences sampled from other unseen epitopes. For each prediction, we separately train a model with an enlarged training set containing also the synthetic binder of the target out-of-sample epitope. The columns labeled with [u] refer to scores obtained balancing the training set by only undersampling the negative class, for comparison. The column d[in] represents the Levenshtein distance from the closest in-sample epitope, showing that scores degrade when moving away from in-sample data.

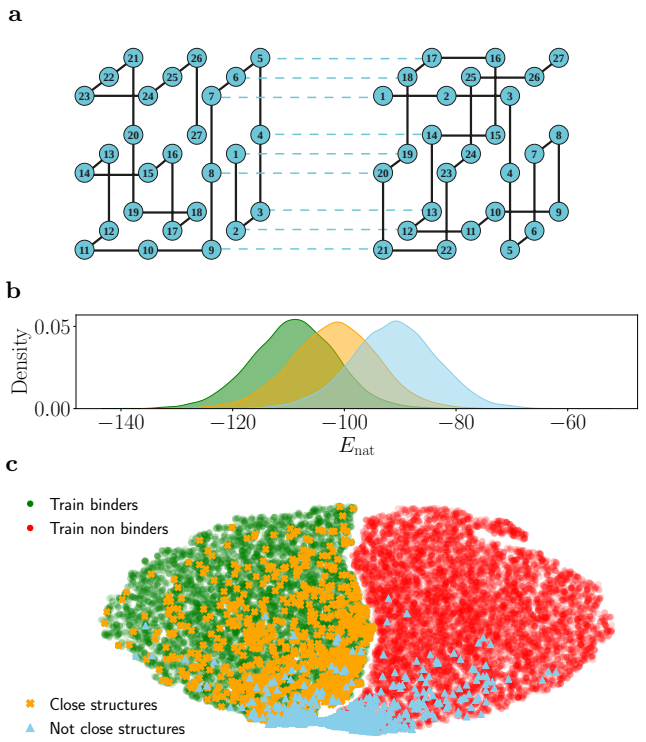


Fig. 5. Out-of-sample predictions on synthetic LP dimers for pan-specific case. **a)** The two structures of the proteins making the dimer, with amino acids defining a strong binding interaction, represented by the dotted lines. **b)** Histogram of single structure folding scores (the lower the better), computed according to the ground truth of the model (see Methods, Equation (3)). In practice we take MSA of binding sequences used for training and compute the folding scores in their native structure or as if they are in an out-of-sample close structure or not close structure. The three distributions confirm the vicinity of the sequence data orange structures to the green ones, compared to blue ones. **c)** tSNE visualization of in-sample training data (binder and non binders, green and red respectively) and out-of-sample hold-out data (close and not close structures binders) over the embeddings of our CNN architecture. In this layer the classification is linear and we can see a clear decision boundary separating green and red data points. Accuracy on in-sample data is ACC = 0.99. The close out-of-sample data are similar to training data, hence the model performs well on these ones (ACC = 0.98); conversely, it gives poorer performances on the not close out-of-sample (blue data points, ACC = 0.82).

under-represented positive data class through generation yields better performances than by simply undersampling the negative, over-represented class. The gain in performance is found not only for in-sample data, but also for out-of-sample tests. In this latter situation, the quality of predictions however decreases with the dissimilarity between the tested epitope and the ones present in the test set. This effect, and more generally the reasons why restoring balance improves classification performances, can be geometrically interpreted in the feature space of the classifier – see SI Section S.5.

The proposed pipeline resorts to unsupervised learning to balance the training set of a supervised classifier. A natural benchmark for performance is the direct use of the unsupervised model alone to classify the data. By fixing a threshold on the score that the model assigns to test sequences, measuring how likely they are sampled from the distribution of the positive data used for training, we may obtain predictions for class membership from the unsupervised model alone. This 'unsupervised classification' procedure generally yields worse generalization results than the full pipeline proposed here, as reported in SI Section S.1. The decision boundary obtained by fixing a threshold on the score of the unsupervised model, which is trained on positive data only, does not coincides with the surface separating positive and negative features in sequence space, see Section S.5.

Restoring balance in the data is crucial to improve predictive power not only when one designs peptide-specific models, a case in which the imbalance is present at class level only, but also for pan-specific models, for which the amount of data associated to each epitope may largely vary, and some peptide are heavily under–represented. Restoring balance between the different epitope groups via generative models can be done with peptide- and pan-specific generative models. Though we have shown that performances are comparable with the two approaches, we believe that our conclusion is dependent on the number and sizes of epitope groups involved in the analysis, as it is clearly harder to learn probability distributions of peptide-specific classes with few known CDR3β binders. An alternative approach could be that of combining peptide- and pan-specific models not only for the supervised predictive task – as already proposed in (48) – but also for data generation; in this way, a limited number of generative models would be trained separately over the most abundant peptide-specific classes, while a single generative model would be trained over peptide and CDR3β pairs of sequences. Depending on the peptide, one of the two generative models could then be adopted. Furthermore, the effect of group imbalance could be reduced introducing a re-weighting factor for each group in the loss function of the generative model during training – inversely proportional to the group size.

The training data used in this study for peptide- and pan-specific models have been collected from publicly available databases, which curate and provide T-cell receptor sequences and their cognate targets published in literature. Despite recent advantages and efforts to make available also negative assays of TCR-epitope bindings, such resources remains biased towards positive interactions, as negative interactions are rarely reported in experiments. To obtain negatively/non–interacting pairs of sequences practitioners resort to various strategies, including the reshuffling of pairs of epitopes and CDR3β. However, recent works have shown that the production of negative data may induce biases and impact the predictive power of models (15, 49). An alternative approach for obtaining negative data could be the use of antigen complexes with low binding affinity generated by computational frameworks intended to design synthetic lattice-based receptor (50, 51). Together with the increasing amount of available negative assays, we believe that these studies will contribute to better understand and limit the sources of biases stemming from negative samples within the context of TCR-epitope binding analysis.

In conclusion, our results demonstrate the benefit of reducing imbalance for both peptide– and pan–specific models, while suggesting to be more important when there is an heavy imbalance in the initial dataset of natural sequences. Preprocessing the dataset to restore balance is effective across multiple strategies, ranging from simple undersampling of abundant sequences to the generation of new peptide-specific CDR3β sequences with unsupervised architectures, including energy-based models and Transformers. The

robustness of these gains suggest that even better TCR specificity predictions could be achieved with more powerful generative models and/or hyperparameter optimization. For instance, our learning pipeline could be easily extended by including the possibility to feed CDR3α chains, CDR1 and CDR2 sequences or MHC class information, likely leading to performance improvement as shown in (8, 30). Similarly, better performances could be reached by simultaneously training the generative and the classifier models, rather than one after the other. In addition, to study the impact of the training set composition on the performances alone, we did not fine-tune hyperparameters and trained all our models with the same number of total iterations over the batch, *i.e.* by rescaling the epochs according to the training set size.

To end with, let us emphasize that our generative models allow for the possibility of producing new, putative CDR3β sequences with desired binding specificity that could be tested experimentally. Due to its methodological simplicity and flexibility, our learning framework is not limited to the context of T-cell receptors specificity predictions, and could be applied to other sequence–based computational problems, where heavy imbalance impedes the proper training of predictive models (*e.g.* (52–55)).

Materials and Methods

Datasets collection. Supervised and unsupervised models developed in this work have been scored using multiple type of synthetic and real data. In all cases – after preprocessing – positive and negative data were merged together in a final dataset with a proportion ρ_P:ρ_N that can be tuned. The dataset was then split into three parts for training, validation and test. Subsampling from the previous test set, we also construct a balanced test set having the same numbers of positive and negative examples. We refer to performance on the latter set as in-sample performance. We then assess model predictions on unseen (balanced) examples to define out-of-sample performance.

We provide below details on data collection and preparation for synthetic and real data.

Synthetic Lattice Protein dimer data. In this work we consider Lattice Proteins (LPs) as a fully controlled setting, where the ground truth distribution of data is known and its properties are tunable *ad hoc*. LPs consist of a computationally tractable model introduced to study the folding and binding properties of proteins (56, 57). A LP monomer is defined as a self-avoiding path over a $3 \times 3 \times 3$ lattice cube, whose conformation defines a structure S. There are $\mathcal{N} = 103,406$ distinct structures (up to global symmetries) on the cube. The probability that a sequence $\mathbf{v}$ of $L = 27$ amino acids folds into structure S is

$$P_{\rm nat}(S|\mathbf{v}) = \frac{e^{-E(\mathbf{v}|S)}}{\sum_{S'} e^{-E(\mathbf{v}|S')}}, \qquad [2]$$

where the sum runs over a representative subset of all distinct structures. The energy of the sequence in a structure, $E(\mathbf{v}|S)$, is given by

$$E(\mathbf{v}|S) = \sum_{i<j} c^S_{ij}\, E_{\rm MJ}(v_i, v_j), \qquad [3]$$

where c^S is the contact map of the structure ($c^S_{ij} = 1$ if i, j are in contact, $c^S_{ij} = 0$ otherwise) and $E_{\rm MJ}$ is the Miyazawa-Jernigan energy matrix, a proxy for the effective interaction energy between pairs of amino acids.

Sequence data for LP dimers are obtained from (47), where two monomer sequences $\mathbf{v}_1$, $\mathbf{v}_2$ – folded in, respectively, structures S_1, S_2 – form a dimeric complex via the interaction energy

$$I(\mathbf{v}_1, \mathbf{v}_2|S_1 + S_2, \pi) = \sum_{i<j} c^{(S_1+S_2,\pi)}_{ij}\, E_{\rm MJ}(v_i, v_j), \qquad [4]$$

where the sum runs over all sites of both structures. The index π in Equation (4) labels a specific orientation of the interaction, see (47). In analogy with Equation (2), the probability that the sequences $\mathbf{v}_1, \mathbf{v}_2$ fold into the dimer $S_1 + S_2$ is

$$P_{\rm int}(\pi, S_1 + S_2|\mathbf{v}_1, \mathbf{v}_2) = \frac{e^{-I(\mathbf{v}_1,\mathbf{v}_2|S_1+S_2,\pi)}}{\sum_{\pi'} e^{-I(\mathbf{v}_1,\mathbf{v}_2|S_1+S_2,\pi')}}, \qquad [5]$$

where the sum runs over all possible orientations.

Given two structures S_1, S_2, the authors collect sequences through MCMC dynamic by accepting or rejecting a mutation at each evolution step based on the total probability

$$\mathcal{P} \propto P_{\rm nat}(S_1|\mathbf{v_1})P_{\rm nat}(S_2|\mathbf{v_2})P_{\rm int}(\pi = 0, S_1 + S_2|\mathbf{v}_1, \mathbf{v}_2). \qquad [6]$$

The MSAs are collected at steps $t = 0$, $t = 100$, $t = 500$ and represent non binder, weak and strong binder dimers used in this work. We refer the interested reader to (47) for an extensive presentation of the LP model and details on how to obtain sequence data. In particular to generate the synthetic dataset used for the pan–specific model we select 12 pairs of structures and for each of them collect an MSA of binding and non binding sequences with the same length to ensure balance. The pairs of structures compose the dataset used in this work have the following ids $1100 + 1701, 249 + 7801, 2789 + 7511, 333 + 794, 3412 + 9422, 456 + 259, 514 + 3894, 5809 + 6682, 9432 + 8754$.

The out-of-sample data for close structures is collected by first finding the closest structures to those used in the in-sample dataset by minimizing the mean energy in Equation (3) over the in-sample binder MSA across all the structures available in (47, 58). We then generate the MSA for the selected close structure pair $(2237 + 304)$, and this constitutes the close out-of-sample dataset. Next we pick a random pair of structures and collect a MSA that will be our not close out-of-sample dataset.

TCR-peptide binding data. Sequence data for TCR-peptide predictions were retrieved from the Immune Epitope Database (IEDB) as of June 2023, filtering the Database entries as follows. We focus our attention on human host immune response and set the pMHC restriction to the HLA-A*02:01 complex, limiting the peptide length to $8 - 11$ amino acids. Paired CDR3β-epitope sequences were thus identified as those for which a T-cell assay was reported 'Positive' or 'Positive-High' and never 'Negative'. Background sequences of CDR3β considered as those non-self reactive are taken from the database assembled by (34), which merges together (i) unique clones from the 743 donors of the cohort in (39) and (ii) the dataset of CDR3β sequences from healthy donors in (40). The full dataset is then sub-sampled at random for computational purpose and we retain 10^6 background sequences. Whenever sequence are fed into the RBM learning, they are required to have the same input length. Therefore, CDR3β sequences are aligned with an Hidden Markov Model (HMM) using as alignment profile the one built in (59). Aligned CDR3β sequences have fixed length of 20 amino acids. To train the supervised model with RBM-generated sequences we drop all the gaps inserted by the alignment procedure, as our deep classifier can handle sequences of different lengths.

Unsupervised generative models. Generally speaking, unsupervised machine learning aims to learn an energy landscape by inferring parameters of a user-defined probabilistic model $P_{\rm model}$ over the data presented (in our case, the peptide specific CDR3β sequences). The model is trained (or fine-tuned) separately over each class of peptide specific receptors and is then asked to generate new sequences compatible with real ones by sampling from the learnt probability landscape.
To generate new CDR3β sequences we make use of Restricted Boltzmann Machines (RBMs) (35, 60) and of Large Language Models (LLMs), specifically we use architectures based on Bidirectional Encoder Representations from Transformer (BERT) adapted to handle CDR3β sequences (37).

Restricted Boltzmann Machines. RBMs are bipartite graphical models including a set of L visible units $\mathbf{v} = (v_1, \cdots, v_L)$ and M hidden (or latent) units $\mathbf{z} = (z_1, \cdots, z_M)$. Only connections

between visible and latent units are allowed through the interaction weights $w_{i\mu}$. RBMs define a joint probability distribution over $\mathbf{v}$ and $\mathbf{z}$ as the Gibbs distribution

$$P(\mathbf{v},\mathbf{z}) = \frac{1}{Z}\exp\left\{\sum_{i=1}^{L} g_i(v_i) - \sum_{\mu=1}^{M}\mathcal{U}_\mu(z_\mu) + \sum_{\mu,i}^{M,L} z_\mu w_{i\mu}(v_i)\right\}, \quad [7]$$

The joint probability above is specified by a set of parameters whose values are inferred from the data (here the CDR3β sequences). They consist of i) the set of single-site biases g_i's acting on visible units that capture the amino acid usage at each sequence position; ii) the potentials $\mathcal{U}_\mu$'s acting on hidden units, here assumed to have dReLU shape (60), and iii) the set of weights $w_{i\mu}$ coupling the hidden and visible layers.
The probability of data $P(\mathbf{v})$ is defined as the marginal of the joint probability over the data itself

$$P(\mathbf{v}) = \int \prod_{\mu}^{M} dz_\mu P(\mathbf{v},\mathbf{z}). \quad [8]$$

All the parameters in the model are learnt by maximizing the marginal log-likelihood

$$\frac{1}{\|D\|}\sum_{\mathbf{v}\in D}\log P(\mathbf{v}) = \langle \log P(\mathbf{v})\rangle_D, \quad [9]$$

where D is our dataset and $\|D\|$ its size. Due to the nature of RBMs, all data needs to have the same dimension to be accepted: in practice, we align the CDR3β receptors before feeding them into the model.
Let us define the RBM energy as minus the exponential term present in the joint probability in Equation (7)

$$\mathcal{H}(\mathbf{v}) = -\left(\sum_{i}^{L} g_i(v_i) + \sum_{\mu}^{M}\Gamma_\mu(I_\mu(\mathbf{v}))\right), \quad [10]$$

where we introduced the shorthand notation $I_\mu(\mathbf{v}) = \sum_i w_{i\mu}(v_i)$ and $\Gamma_\mu(I(\mathbf{v})) = \log\int dz\, e^{-\mathcal{U}_\mu(z)+hI(\mathbf{v})}$. For a generic parameter $\boldsymbol{\theta} = \{g_i(v_i), w_{i\mu}(v_i), U_\mu(\mathbf{z})\}$ the rule to infer its value is given by the gradient-ascent equations stemming from log-likelihood maximization,

$$\frac{\partial}{\partial\boldsymbol{\theta}}\langle\log P(\mathbf{v})\rangle_D = \left\langle\frac{\partial}{\partial\boldsymbol{\theta}}\mathcal{H}(\mathbf{v})\right\rangle_D - \left\langle\frac{\partial}{\partial\boldsymbol{\theta}}\mathcal{H}(\mathbf{v})\right\rangle_m, \quad [11]$$

where $\langle\cdot\rangle_m$ stands for the average over the model, $\langle u(\mathbf{v})\rangle_m = \sum_{\mathbf{v}} P(\mathbf{v})u(\mathbf{v})$. In addition, regularization terms can be added to control the values of inferred parameters, by enforcing sparsity on the weights through L_1-penalty to avoid overfitting and by controlling their norm through L_2-penalty to prevent divergences. In practice we resort to a L_1^2 regularization scheme that consists in adding to the log-likelihood of data the following penalty term

$$-\lambda\sum_{\mu}^{M}\left(\sum_{i}^{L}\sum_{v_i}|w_{i\mu}(v_i)|\right)^2, \quad [12]$$

where λ sets the regularization strength. It has suggested that such regularization also helps to improve the generative properties of RBMs. The parameters used for learning of peptide-specific CDR3β distributions are:

$$\begin{aligned} L &= 20 && \text{number of visible units,}\\ M &= 20 && \text{number of hidden units,}\\ Q &= 21 && \text{size of the alphabet,}\\ \lambda &= 0.01 && \text{regularization strength.}\end{aligned} \quad [13]$$

Generative protocol for RBMs. Once the full set of parameters has been inferred from the training data, we can sample from the probability distribution $P(\mathbf{v})$ to obtain new data (in our case, new peptide specific CDR3β receptors). Here we sample using Alternate Gibbs Sampling (AGS), which consists in alternatively sampling from the RBM's visible layer while keeping the hidden layer fixed from $P(\mathbf{v}|\mathbf{z})$ and viceversa from $P(\mathbf{z}|\mathbf{v})$. The Monte Carlo Markov Chain is initialized starting from sequence data in the dataset and new sequences are collected after some steps of thermalization to avoid sampling correlated data with real ones.

BERT-based architectures. Our generative model of peptide-specific CDR3β sequences is based on the Bidirectional Encoder Representations from Transformer model architecture (61). Transformers models are build through a series of attention blocks and feed-forward layers, whose aim is to capture interactions across the input sequence through learning how strongly the embedding of each token in the input is affected by the other tokens (the context). Stacking multiple attention blocks allow the model to learn complex structures within the embedding of the input data, such semantic, causal and grammatical properties of the data.
The model takes as input for training CDR3β sequences for the peptide-specific case and pairs of peptide and CDR3β sequences for the pan-specific case; the sequences are formatted via the tokenizer retrieved from (37), which contains a total of 26 tokens spanning the 20 amino acids and some special tokens, such as the prefix and suffix token and the masking one. An additional token & is included for the pan-specific case to model the peptide – CDR3β separation in input data. Sequences do not need to be aligned, and the inputs are padded with an attention mask. The input tokens are thus padded through the embedding layer to get a continuous representation vector of the data: such embedded vector (together with a positional encoding vector that retains the positional information of amino acids in the input sequence) goes through the model attention blocks. The output feature vector leaves in a $L\times 768$ dimensional space, where L is the input length, and is fed to a task-specific head for masked language modelling.

The training over our dataset is carried out performing masked language modelling (MLM) objective, where we mask a fraction of input token in the dataset and train the architecture to predict the correct ones. This allows the model to learn the grammar underlying the CDR3β and peptide sequence patterns. For the peptide-specific case, the generative model is obtained by fine-tuning the TCR-BERT model from (37) through few epochs of MLM objective over the peptide-specific CDR3β data; this leverages the transfer-learning ability of the model that has trained over unlabelled TCR sequences to capture distinctive features of the peptide-specific CDR3β sequences. For the pan-specific model, we train the model from BERT weights over the full training dataset; at variance with BERT model, we set the maximal positional embbeding length allowed to 64, due to the shortness of sequence data compared to text data.

For the (fine-)tuning procedure, given a TCR dataset $\mathcal{M}$ and a masking pattern $\hat{q}$, we minimize the following training loss

$$\mathcal{L}_{\text{MLM}}(\mathcal{M},\hat{q};\theta) = -\sum_{(i,r)\in\hat{q}}\log p(a_{i,r}|\hat{q};\theta), \quad [14]$$

where θ is the set of all parameters in the model and (i,r) run over the residue position and sequences, respectively. The conditional probabilities p for each (i,r) are computed using the softmax-normalized model output values - the *logits* - for each symbol in the alphabet. For each input sequence, we mask 15% of amino acids as done in the original training (61). All hyperparameters during fine-tuning are the same as the ones used for training in (61).

Generative protocol for BERT-like models. Once the model has been (fine-)tuned on the MLM downstream task, we leverage its generative power following the iterative masking scheme proposed in (62, 63) for protein sequences. In practice, each step of this procedure works as follows:

(i) we randomly mask with probability q each entry of the CDR3β sequence or we leave it unchanged with probability $1-q$. This defines a masking pattern $\hat{q}$;

(ii) we feed the masked sequence to the model and replace masked entries sampling from the softmax (normalized) distribution of the model logits at inverse temperature β;

(iii) we repeat steps (i) and (ii) for T times and then we store a new sample sequence.

In practice, we set $q = 0.2$, $\beta = 1$ and $T = 20$ and we repeat this scheme many times starting from the last configuration to obtain the desired number of TCR sequences. At the end we merge all the generated CDR3β and eventually drop duplicates: the remaining samples constitute the new set of sequences that will augment peptide specific sets.
Note that our protocol does not allow insertion or deletions across the sequence, as only residues tokens can be masked: in this way, the length distribution of training and generated data remains the same.

Model architecture for TCRs supervised predictions. Generally speaking, supervised machine learning model aims at finding the best feature map that connects data properties to their label, based on the observation data points (in our case, CDR3β sequences) in the training data. Once the model has been trained on labelled data, it can apply the feature map on unseen examples and classify them as belonging to a specific class. A pletora of models – ranging from simple to higher complexity architectures – can accurately solve this task and many of them are widely implemented for TCR specificity predictions. Here we use a specific architecture, but we the results on restoring balance through generative models of CDR3β are general and hold beyond the particular network architecture.

We implemented a 1-D Convolutional Neural Network (CNN) to make binary and multiclass predictions on TCRs specificity. Our architecture takes as input raw CDR3β sequences and returns a single value corresponding to the target class label assigned by the model (*i.e.* if the shown CDR3β binds one of the epitopes or it belongs to the bulk repertoires).
CDR3β sequences are first padded to a max length $L = 30$, then translated into score matrices using the BLOSUM50 encoding matrix with no gaps or special amino acids included, see (64). Hence each sequence is mapped into a score matrix of size $L \times 20$. The encoded input is then fed into the network architecture, consisting of five convolutional layers having 16 filters and different kernel sizes $\{1, 3, 5, 7, 9\}$. The resulting feature vectors go through a batch-normalization layer to reduce overfitting and are then concatenated to pass through a dense layer with 16 hidden neurons. An additional batch-normalization layer is applied and the resulting 16 dimensional feature vector is used to visualize the embeddings constructed by the model starting from raw data. The final classification is performed over such embeddings by feeding the feature vector to a layer with a number of neurons equal to the number of classes in the dataset and having softmax activation function (for binary classification tasks, we actually use one neuron and sigmoid activation function). We use the ReLU activation function throughout the network.
Adam optimizer with learning rate $\eta = 0.001$ and categorical (or binary) cross-entropy loss function are used for learning with a batch size of 128 samples.

Performances are evaluated using accuracy (ACC) and Area Under the receiver operating characteristic Curve (AUC) metrics on balanced test sets, unless otherwise specified. The former is defined as

$$\mathrm{ACC} = \frac{\text{number of correctly classified test data}}{\text{number of test data}} . \qquad [15]$$

In the case of binary classification, the receiver operating characteristic (ROC) curve is defined as the parametric curve in the space False Positive Rate–True Positive Rate as a function of the threshold γ used to assign the binary label to the output of the sigmoid activation function of the readout layer. In the case of multiclass classification, AUC is defined as a uniform average of the binary AUCs of all possible combinations of classes (one-vs-one):

$$\mathrm{AUC} = \frac{1}{c(c-1)} \sum_{j=1}^{c} \sum_{k>j}^{c} [\mathrm{AUC}(j|k) + \mathrm{AUC}(k|j)], \qquad [16]$$

c being the number of classes.

Code and data availability. Code to reproduce the analysis in this paper will be available at https://github.com/Eloffredo/TCRbalance. Sequence data for TCR-peptide predictions are public and were retrieved from the Immune Epitope Database (IEDB) (10) as of June 2023 and from TChard (41), available at https://zenodo.org/records/6962043; utilities to collect them can be found in the above GitHub repository.

ACKNOWLEDGMENTS. We acknowledge funding from the CNRS - University of Tokyo "80 Prime" Joint Research Program and from the Agence Nationale de la Recherche (ANR-19 Decrypted CE30-0021-01 to S.C. and R.M.). E.L. thanks Andrea Di Gioacchino for interesting suggestions during the early stages of this work and his help with the use of the alignment software of receptor sequences. M.P. thanks Riccardo Capelli for discussions. The authors thank Victor Greiff, Eugen Ursu and Aygul Minnegalieva for comments and for the careful reading of the manuscript.

References

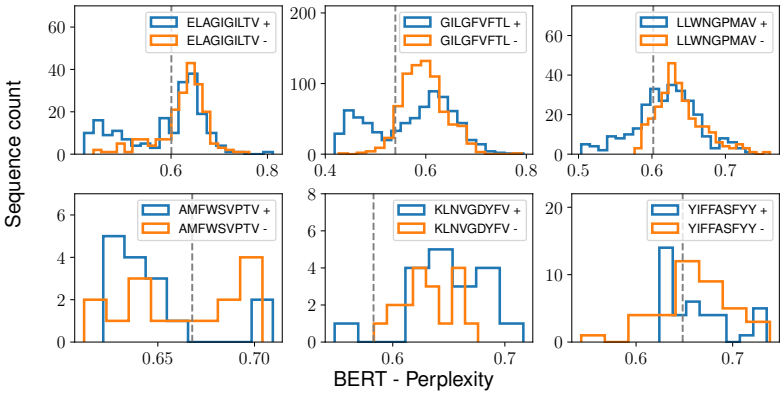
1. N Zhang, MJ Bevan, CD8+ T cells: foot soldiers of the immune system. *Immunity* **35**, 161–168 (2011).
2. MJ Sim, TCRs and AI: the future is now. *Nat. Rev. Immunol.* **24**, 3–3 (2024).
3. P Meysman, et al., Benchmarking solutions to the T-cell receptor epitope prediction problem: IMMREP22 workshop report. *ImmunoInformatics* **9**, 100024 (2023).
4. ZS Ghoreyshi, JT George, Quantitative approaches for decoding the specificity of the human T cell repertoire. *Front. Immunol.* **14**, 1228873 (2023).
5. A Weber, A Pélissier, MR Martínez, T cell receptor binding prediction: A machine learning revolution. *arXiv* (2023).
6. Y Nagano, et al., Contrastive learning of T cell receptor representations. *arXiv* (2024).
7. P Dash, et al., Quantifiable predictive features define epitope-specific T cell receptor repertoires. *Nature* **547**, 89–93 (2017).
8. A Montemurro, et al., NetTCR-2.0 enables accurate prediction of TCR-peptide binding by using paired TCRα and β sequence data. *Commun. biology* **4**, 1060 (2021).
9. I Springer, N Tickotsky, Y Louzoun, Contribution of T cell receptor alpha and beta CDR3, MHC typing, V and J genes to peptide binding prediction. *Front. immunology* **12**, 664514 (2021).
10. R Vita, et al., The immune epitope database (IEDB): 2018 update. *Nucleic acids research* **47**, D339–D343 (2019).
11. M Shugay, et al., VDJdb: a curated database of T-cell receptor sequences with known antigen specificity. *Nucleic acids research* **46**, D419–D427 (2018).
12. DV Bagaev, et al., VDJdb in 2019: database extension, new analysis infrastructure and a T-cell receptor motif compendium. *Nucleic Acids Res.* **48**, D1057–D1062 (2020).
13. N Tickotsky, T Sagiv, J Prilusky, E Shifrut, N Friedman, McPAS-TCR: a manually curated catalogue of pathology-associated T cell receptor sequences. *Bioinformatics* **33**, 2924–2929 (2017).
14. W Zhang, et al., PIRD: pan immune repertoire database. *Bioinformatics* **36**, 897–903 (2020).
15. E Ursu, et al., Training data composition determines machine learning generalization and biological rule discovery. *bioRxiv* (2024).
16. A Fernández, et al., *Learning from Imbalanced Data Sets.* (Springer Cham), (2018).
17. E Francazi, M Baity-Jesi, A Lucchi, A theoretical analysis of the learning dynamics under class imbalance in *Proceedings of the 40th International Conference on Machine Learning*, Proceedings of Machine Learning Research, eds. A Krause, et al. (PMLR), Vol. 202, pp. 10285–10322 (2023).
18. SS Mannelli, F Gerace, N Rostamzadeh, L Saglietti, Bias-inducing geometries: an exactly solvable data model with fairness implications. *arXiv* (2023).
19. E Loffredo, M Pastore, S Cocco, R Monasson, Restoring balance: principled under/oversampling of data for optimal classification in *Forty-first International Conference on Machine Learning.* (2024).
20. L Deng, et al., Performance comparison of TCR-pMHC prediction tools reveals a strong data dependency. *Front. Immunol.* **14** (2023).
21. M Zięba, JM Tomczak, A Gonczarek, RBM-SMOTE: Restricted boltzmann machines for synthetic minority oversampling technique in *Intelligent Information and Database Systems*, eds. NT Nguyen, B Trawiński, R Kosala. (Springer International Publishing, Cham), pp. 377–386 (2015).
22. Z Wan, Y Zhang, H He, Variational autoencoder based synthetic data generation for imbalanced learning in *2017 IEEE Symposium Series on Computational Intelligence (SSCI).* pp. 1–7 (2017).
23. G Douzas, F Bacao, Effective data generation for imbalanced learning using conditional generative adversarial networks. *Expert. Syst. with Appl.* **91**, 464–471 (2018).
24. B Mirza, D Haroon, B Khan, A Padhani, TQ Syed, Deep generative models to counter class imbalance: A model-metric mapping with proportion calibration methodology. *IEEE Access* **9**, 55879–55897 (2021).
25. AK Mondal, L Singhal, P Tiwary, P Singla, P AP, Minority oversampling for imbalanced data via class-preserving regularized auto-encoders in *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research, eds. F Ruiz, J Dy, JW van de Meent. (PMLR), Vol. 206, pp. 3440–3465 (2023).

26. Q Ai, et al., Generative oversampling for imbalanced data via majority-guided VAE in *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research, eds. F Ruiz, J Dy, JW van de Meent. (PMLR), Vol. 206, pp. 3315–3330 (2023).
27. M Marouf, et al., Realistic in silico generation and augmentation of single-cell RNA-seq data using generative adversarial networks. *Nat. Commun.* **11**, 166 (2020).
28. S Gielis, et al., Detection of enriched T cell epitope specificity in full T cell receptor sequence repertoires. *Front. immunology* **10**, 2820 (2019).
29. MDN Pham, et al., epiTCR: a highly sensitive predictor for TCR–peptide binding. *Bioinformatics* **39**, btad284 (2023).
30. JW Sidhom, HB Larman, DM Pardoll, AS Baras, DeepTCR is a deep learning framework for revealing sequence concepts within T-cell repertoires. *Nat. communications* **12**, 1605 (2021).
31. I Springer, H Besser, N Tickotsky-Moskovitz, S Dvorkin, Y Louzoun, Prediction of specific TCR-peptide binding from large dictionaries of TCR-peptide pairs. *Front. immunology* **11**, 534364 (2020).
32. T Lu, et al., Deep learning-based prediction of the T cell receptor–antigen binding specificity. *Nat. machine intelligence* **3**, 864–875 (2021).
33. Z Sethna, et al., Population variability in the generation and selection of T-cell repertoires. *PLOS Comput. Biol.* **16**, e1008394 (2020).
34. G Isacchini, AM Walczak, T Mora, A Nourmohammad, Deep generative selection models of T and B cell receptor repertoires with soNNia. *Proc. Natl. Acad. Sci.* **118**, e2023141118 (2021).
35. B Bravi, et al., A transfer-learning approach to predict antigen immunogenicity and T-cell receptor specificity. *ELife* **12**, e85126 (2023).
36. S Yadav, DS Vora, D Sundar, JK Dhanjal, TCR-ESM: Employing protein language embeddings to predict TCR-peptide-MHC binding. *Comput. Struct. Biotechnol. J.* **23**, 165–173 (2024).
37. K Wu, et al., TCR-BERT: learning the grammar of T-cell receptors for flexible antigen-xbinding analyses. *bioRxiv* pp. 2021–11 (2021).
38. B Meynard-Piganeau, C Feinauer, M Weigt, AM Walczak, T Mora, Tulip: A transformer-based unsupervised language model for interacting peptides and t cell receptors that generalizes to unseen epitopes. *Proc. Natl. Acad. Sci.* **121**, e2316401121 (2024).
39. RO Emerson, et al., Immunosequencing identifies signatures of cytomegalovirus exposure history and HLA-mediated effects on the T cell repertoire. *Nat. genetics* **49**, 659–665 (2017).
40. J Dean, et al., Annotation of pseudogenic gene segments by massively parallel sequencing of rearranged lymphocyte receptor loci. Genome Med 7: 123 (2015).
41. F Grazioli, et al., On TCR binding predictors failing to generalize to unseen peptides. *Front. Immunol.* **13** (2022).
42. M Klinger, et al., Multiplex identification of antigen-specific T cell receptors using a combination of immune assays and immune receptor sequencing. *PloS one* **10**, e0141561 (2015).
43. S Nolan, 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. square* (2020).
44. S Sagawa, PW Koh, TB Hashimoto, P Liang, Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *CoRR* **abs**/**1911.08731** (2019).
45. BY Idrissi, M Arjovsky, M Pezeshki, D Lopez-Paz, Simple data balancing achieves competitive worst-group-accuracy in *Proceedings of the First Conference on Causal Learning and Reasoning*, Proceedings of Machine Learning Research, eds. B Schölkopf, C Uhler, K Zhang. (PMLR), Vol. 177, pp. 336–351 (2022).
46. G Croce, et al., Deep learning predictions of TCR-epitope interactions reveal epitope-specific chains in dual alpha T cells. *Nat. Commun.* **15**, 3211 (2024).
47. E Loffredo, et al., Evolutionary dynamics of a lattice dimer: a toy model for stability vs. affinity trade-offs in proteins. *J. Phys. A: Math. Theor.* **56**, 455002 (2023).
48. MF Jensen, M Nielsen, NetTCR 2.2 - improved TCR specificity predictions by combining pan- and peptide-specific training strategies, loss-scaling and integration of sequence similarity. *ELife* **12**, RP93934 (2023).
49. C Dens, K Laukens, W Bittremieux, P Meysman, The pitfalls of negative data bias for the T-cell epitope specificity challenge. *bioRxiv* pp. 2023–04 (2023).
50. R Akbar, et al., In silico proof of principle of machine learning-based antibody design at unconstrained scale. mAbs 14. *mAbs* (2022).
51. PA Robert, et al., Unconstrained generation of synthetic antibody–antigen structures to guide machine learning methodology for antibody specificity prediction. *Nat. Comput. Sci.* **2**, 845–865 (2022).
52. MM Haque, MK Skinner, LB Holder, Imbalanced class learning in epigenetics. *J. Comput. Biol.* **21**, 492–507 (2014).
53. Z Ming, et al., HostNet: improved sequence representation in deep neural networks for virus-host prediction. *BMC bioinformatics* **24**, 455 (2023).
54. P Rana, A Sowmya, E Meijering, Y Song, Imbalanced classification for protein subcellular localization with multilabel oversampling. *Bioinformatics* **39**, btac841 (2023).
55. M Li, et al., Protein-protein interaction sites prediction based on an under-sampling strategy and random forest algorithm. *IEEE/ACM transactions on computational biology bioinformatics* **19**, 3646–3654 (2021).
56. H Li, R Helling, C Tang, N Wingreen, Emergence of preferred structures in a simple model of protein folding. *Science* **273**, 666–669 (1996).
57. L Mirny, E Shakhnovich, Protein folding theory: from lattice to all-atom models. *Annu. review biophysics biomolecular structure* **30**, 361–396 (2001).
58. H Jacquin, A Gilson, E Shakhnovich, S Cocco, R Monasson, Benchmarking inverse statistical approaches for protein structure and design with exactly solvable models. *PLoS Comput. Biol* **12**, e1004889 (2016).
59. B Bravi, et al., Probing T-cell response by sequence-based probabilistic modeling. *PLoS Comput. Biol.* **17**, e1009297 (2021).
60. J Tubiana, S Cocco, R Monasson, Learning protein constitutive motifs from sequence data. *Elife* **8**, e39397 (2019).
61. J Devlin, MW Chang, K Lee, K Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
62. A Wang, K Cho, BERT has a mouth, and it must speak: BERT as a Markov random field language model. *arXiv preprint arXiv:1902.04094* (2019).
63. D Sgarbossa, U Lupo, AF Bitbol, Generative power of a protein language model trained on multiple sequence alignments. *Elife* **12**, e79854 (2023).
64. S Henikoff, JG Henikoff, Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci.* **89**, 10915–10919 (1992).
65. JC Skipper, et al., Mass-spectrometric evaluation of HLA-A* 0201-associated peptides identifies dominant naturally processed forms of CTL epitopes from MART-1 and gp100. *Int. journal cancer* **82**, 669–677 (1999).
66. K Chaudhuri, K Ahuja, M Arjovsky, D Lopez-Paz, Why does throwing away data improve worst-group error? in *International Conference on Machine Learning*. (PMLR), pp. 4144–4188 (2023).

6. Supplementary Material

S.1. Unsupervised classification. The pipeline explained in Section 2 relies on the generative power of an unsupervised ML model trained on the positive examples and used to augment and balance the dataset, successively passed to a supervised classifier for training. To better understand how performances depend on the unsupervised and supervised steps, we report below predictions made by the unsupervised model alone. For simplicity, we consider hereafter the case of binary classification. By fixing a threshold on the scores, we can discriminate between positive and negative examples. This simple procedure is expected to be sub-optimal, as the unsupervised model is trained on the positive class alone and has no knowledge about the distribution of negative examples, which is used only to fix the threshold maximizing the accuracy on the two classes. Thus, a decision boundary based on the score of this model does not necessarily aligns with the separating surface of the two classes (see Section S.5 for more details on the geometry of the classification problem).

In Figure S1, we report the histograms of the scores assigned by the unsupervised model trained on positive examples of different peptide-CDR3β pairs, randomly chosen among the ones in Figure 4b. The threshold (gray dashed line) is fixed by maximizing the accuracy on a validation set composed of positive and negative examples of the given peptide. Performances are consistently lower than the ones obtained using both the unsupervised model (to augment the data) and the CNN (to classify).



Peptide	ACC [BERT]	ACC [BERT+CNN]
ELAGIGILTV	0.61	0.62
GILGFVFTL	0.64	0.74
LLWNGPMAV	0.61	0.62
AMFWSVPTV	0.7	0.65
KLNVGDYFV	0.53	0.58
YIFFASFYY	0.59	0.62

Fig. S1. Unsupervised vs. supervised classification. (Left) Histograms of the scores assigned by the unsupervised model to balanced test sets of in-sample epitopes; lower perplexity corresponds to better score. The gray dashed line locates the threshold obtained by maximizing the accuracy of prediction on positive and negative data. Epitopes in the top row are frequent ones and do not get enlarged during training of the supervised classifier. **(Right)** Comparison of the accuracy scores between unsupervised classification only ([BERT]) and the full pipeline ([BERT+CNN]) in Figure 2 for the same peptides as in the left panel. Values in the column [BERT+CNN] can be obtained from Figure 4b.

S.2. Out-of-sample TCR specificity predictions. Similarly to what done for the pan-specific framework in Section 4, we study out-of-sample performances for single-epitope TCR specificity. Our peptide-specific model has learnt the key properties of CDR3β binding to a selected epitope as reported in Section 3 and it remains to be seen to what extent such features can be transferred to predict binding towards other target epitopes against bulk repertoires. We start by considering synthetic data.

Benchmarking on synthetic data reveals strong dependence of performance upon out-of-sample data distribution. We take data from (47), where the authors built dimeric LPs starting from single monomers running Monte Carlo (MC) evolution at variable intra-dimer interaction strengths (see Methods). We collect sequence data in the form of Multiple Sequence Alignments (MSA) at three steps during MC evolution, namely at beginning, intermediate and endpoint of evolution: the MSAs will constitute non binder, weak binder and strong binder data, respectively. We train a supervised model over strong and weak binder classes and use non binders as out-of-sample data, which are thus closer to weak binders than to strong ones. We plot in Figure S2a the binding probability distributions for these three classes of binders.

The trained classifier achieve great performances on the in-sample test set for strong-vs-weak binders classification (AUC = 0.95) and do even better for the out-of-sample strong-vs-non binder classification (AUC = 0.98). Performance are much poorer for the out-of-sample weak-vs-non-binder classification (AUC = 0.69), as shown in Figure S2b; note that for this task we switched labels, *i.e.* weak binders are presented as strong ones and non-binders are deemed as weak, otherwise we would get AUC = 0.31).

We display in Figure S2c the 2d projections of the embedded sequences in the last layer of our architecture (before linear classification) using tSNE. The tSNE visualization shows that the predictive power of the model depends on the location of the out-of-sample cluster in the feature space compared to the in-sample data. The out-of-sample sequence distribution has a large overlap with the one of weak binders, which makes discrimination hard. Conversely, in this feature space, strong binders are well separated from the rest and hence the decision boundary learnt over strong-vs-weak is efficient even against out-of-sample data.

Similar behaviour is observed on out-of-sample natural CDR3β. To assess if the results derived in the controlled framework of synthetic data also holds for natural TCRs, we consider the epitope ELAGIGILTV, which is 2 mutations away from the primary epitope AAGIGILTV expressed on the surface of Melanoma-cancer responsible cells (65). Our dataset includes 2,082 sequences of CDR3β experimentally labelled as binders to this epitope. We train the CNN model to distinguish peptide-specific sequences from bulk ones. We then select as out-of-sample sequences (i) CDR3βs that positively bind the epitope EAAGIGILTV, one mutation away from the wildtype (WT) ELAGIGILTV; (ii) we also select CDR3β binding a very different peptide having Levenshtein distance 8 from our WT, namely VQELYSPIFLIV. We expect that our model be predictive for out-of-sample specificity predictions for the first epitope and not for the second one. Results confirm this guess with values of AUC equal to, respectively, 0.79 and 0.54. Similarly to the case of synthetic data in Figure S2d, this difference in performance is visualized in Figure S3 by the distinct locations of the corresponding sequence distributions in the feature space of the last embedding layer of our classifier.

S.3. Hyperparameter tuning in pan-specific models. In Figure 4b we show results for a specific value of the threshold size $\mathcal{G}$, above which peptide-specific classes are not augmented. The choice of the threshold value can affect the performances and it is task-dependent. Here, we report in Figure S4 results of AUC scores before and after data augmentation through the pan-specific model has been applied.

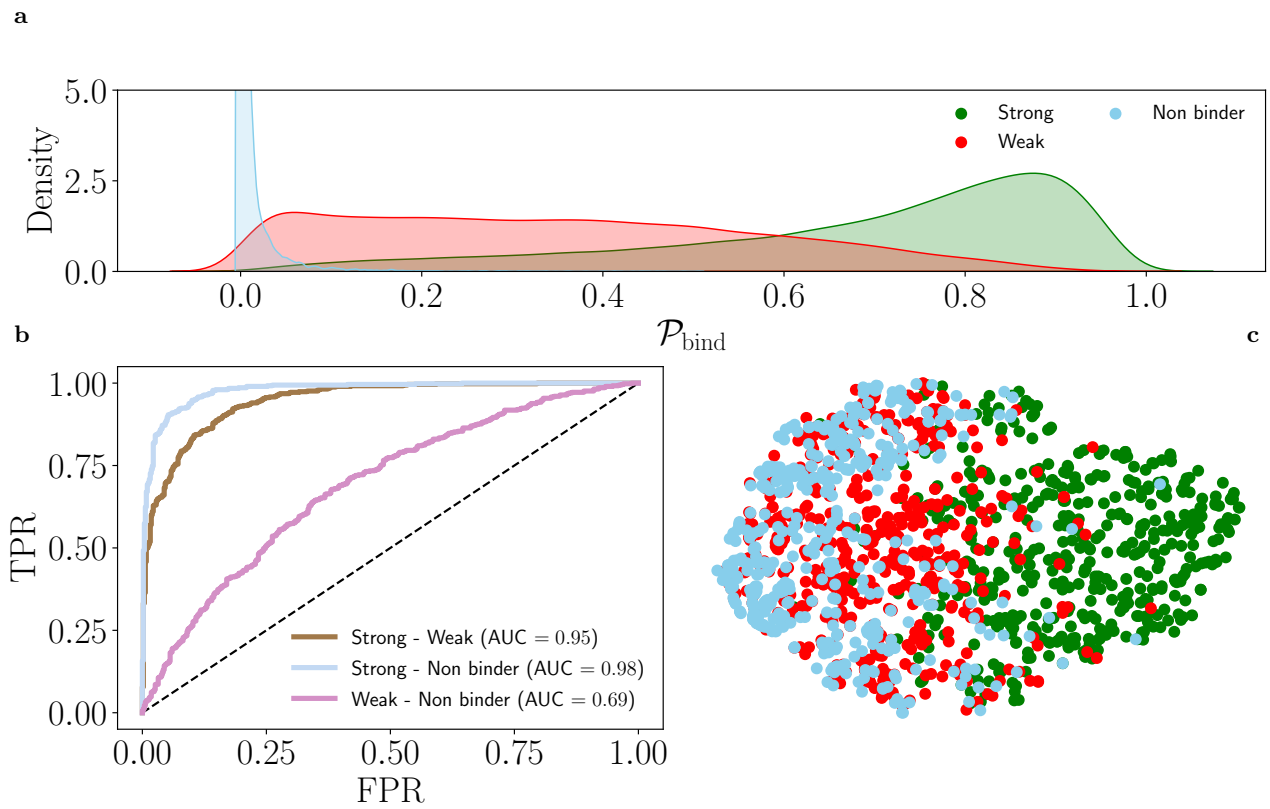


Fig. S2. Out-of-sample analysis on synthetic Lattice-Protein dimers. a) Densities of binding scores $\mathcal{P}_{\text{bind}}$ for strong, weak and non-binding compounds. The y-axis is cut for visualization purpose, as "Non binder" compounds concentrates around zero score. **b)** Receiver Operating Characteristic (ROC) curves for in-sample (Strong - Weak) and out-of-sample (Strong - Non binder and Weak - Non binder) test sets. The classification weak-vs-non binders has worst performances. **c)** tSNE visualization of the embeddings (last feature layer) produced by our CNN architecture for in-sample test data (strong and weak binders) and out-of-sample hold-out data (non binders). Classification is carried out from linear combinations of the embeddings of the input data points; better separation of clusters reflects higher model performances.

Epitopes	AUC	ACC
AMFWSVPTV – VTEHDTLLY – GLCTLVAML	0.57	0.55
ELAGIGILTV – GLCTLVAML – LLWNGPMAV	0.58	0.58
FLYNLLTRV – VTEHDTLLY – GLCTLVAML	0.57	0.55
AMFWSVPTV – ATDALMTGY – YLQPRTFLL	0.57	0.55
VTEHDTLLY – GLCTLVAML – LLWNGPMAV	0.58	0.58

Table S1. Performances scores for peptide-specific models with PM generative model. The supervised architecture is the same and it is trained as in Figure 3 on the same datasets with balance restored via generation of new CDR3β samples.

Undersampling the populated classes is done down to size 5000 CDR3β sequences. As we can see, despite quantitative values depend on $\mathcal{G}$, the picture confirms the overall concept that generative methods of peptide-specific CDR3β sequences help specificity predictions, particularly for under represented classes (small triangles).

S.4. On the choice of the generative model. Our learning framework is based on the idea that restoring balance through a combination of undersampling the strongly over-represented classes and augmenting under-represented classes via generative models. We believe such approach can yield better performances as the supervised network is provided with new informative peptide-specific CDR3β sequences; yet, the quality of generated sequences is crucial to prevent a loss of information. To discuss this point, we benchmark our specificity prediction performances with a simple model that requires zero training: a profile model (PM) that generates new CDR3β sequences by independently sampling each site of the sequence based on the peptide-specific class sequence profile (see Figure S5). Since this model captures only first-order statistics of the sequences we expect it to generate less informative CDR3βs and thus we end with poor specificity predictions (see Table S1 for the scores obtained using this method for the cases reported in Figure 3 of the main text).
The quality of data generation is also based on the stringency of the Gibbs sampling procedure, *i.e.* how easily we allow the generative process to accept random mutations. The degree of randomness in the sampling scheme is set by the temperature value β in the softmax outputs of the model: rescaling the model logits by β, low values of β flatten the softmax output distribution so that amino acids mutations are randomly accepted regardless from the underlying CDR3β distribution learnt. To visualize this effect, we take a peptide-specific classification task and restore balance by generating random CDR3β sequences (see Figure S6); at some point ($\rho_{\text{rand.}} > 50\%$) the random sequences completely take over the CDR3β natural sequences and make the specificity prediction problem impossible to solve, yielding random predictions.

S.5. Geometrical interpretation of restoring balance. Supervised architectures solve the classification problem through the learning of a decision boundary in the high dimensional embedded feature space of input data. In this regard, achieving good performances means

Table S2. List of peptide-specific classes in the dataset. We report sizes and imbalance ratio details on the peptide-specific classes defined by peptide sequences in the TCR dataset of pan-specific models. Only few peptide-specific classes have more than 1% of CDR3β binding sequences, causing heavily imbalance. Details refer to the dataset before training/test splitting.

Peptide sequence	Class size	IR (%)	Peptide sequence	Class size	IR (%)
AELAKNVSLDNVL	1794	1.41	ALRKVPTDNYITTY	346	0.27
ALSKGVHFV	170	0.13	AMFWSVPTV	182	0.14
APHGVVFLHVTYV	244	0.19	APKEIIFLEGETL	1783	1.41
ATDALMTGY	218	0.17	AVFDRKSDAK	1967	1.55
AYKTFPPTEPK	337	0.27	CINGVCWTV	186	0.15
CRVLCCYVL	435	0.34	CTFEYVSQPFLM	196	0.15
EAAGIGILTV	505	0.4	EIYKRWII	180	0.14
ELAGIGILTV	2074	1.63	FGEVFNATRFASVY	418	0.33
FIAGLIAIV	204	0.16	FLCLFLLPSLATV	244	0.19
FLKEKGGL	159	0.13	FLNGSCGSV	2568	2.02
FLPFFSNVTWFHAI	299	0.24	FLPRVFSAV	867	0.68
FLRGRAYGL	138	0.11	FPPTSFGPL	681	0.54
FRCPRRFCF	266	0.21	FTISVTTEIL	198	0.16
FVDGVPFVV	2705	2.13	GDAALALLLLDRLNQL	609	0.48
GILGFVFTL	7419	5.85	GLCTLVAML	7422	5.85
GMEVTPSGTWLTY	995	0.78	GNYTVSCLPFTI	176	0.14
GTSGSPIINR	173	0.14	GTSGSPIVNR	176	0.14
GYQPYRVVVLSF	193	0.15	HTTDPSFLGRY	5787	4.56
ILGLPTQTV	236	0.19	ILHCANFNV	199	0.16
IMLIIFWFSL	1278	1.01	IQYIDIGNY	169	0.13
ITEEVGHTDLMAAY	180	0.14	IVTDFSVIK	621	0.49
KAFSPEVIPMF	253	0.2	KAYNVTQAF	807	0.64
KLGGALQAK	14589	11.5	KLNVGDYFV	169	0.13
KLPDDFTGCV	1319	1.04	KLSYGIATV	2458	1.94
KLWAQCVQL	312	0.25	KPLEFGATSAAL	362	0.29
KRWIILGLNK	401	0.32	KTAYSHLSTSK	474	0.37
LEPLVDLPI	417	0.33	LITGRLQSLQTYV	261	0.21
LITLATCELYHYQECV	251	0.2	LLLDDFVEII	968	0.76
LLLGIGILV	232	0.18	LLQTGIHVRVSQPSL	309	0.24
LLWNGPMAV	2559	2.02	LPRRSGAAGA	2138	1.69
LSPRWYFYYL	1751	1.38	LVVDFSQFSR	1871	1.47
MGYINVFAFPFTIYSL	2918	2.3	MPASWVMRI	777	0.61
MVMCGGSLYV	437	0.34	NLVPMVATV	9278	7.31
NPLLYDANYFLCW	548	0.43	NRDVDTDFVNEFYAY	285	0.22
PKYVKQNTLKLAT	412	0.32	QECVRGTTVL	151	0.12
QLMCQPILLL	980	0.77	RAKFKQLL	996	0.78
RFYKTLRAEQASQ	282	0.22	RLRAEAQVK	464	0.37
RNPANNAAIVL	311	0.25	RPHERNGFTVL	207	0.16
RQLLFVVEV	892	0.7	RSVASQSIIAYTMSL	469	0.37
SEHDYQIGGYTEKW	3424	2.7	SELVIGAVIL	900	0.71
SEVGPEHSLAEY	270	0.21	SFHSLHLLF	186	0.15
SMWSFNPETNIL	199	0.16	SNEKQEILGTVSWNL	451	0.36
SPFHPLADNKFAL	248	0.2	SPRWYFYYL	214	0.17
STDTGVEHVTFFIYNK	243	0.19	STLPETAVVRR	924	0.73
SYFIASFRLFA	219	0.17	TLIGDCATV	568	0.45
TLVPQEHYV	164	0.13	TPINLVRDL	266	0.21
TPRVTGGGAM	2557	2.02	TTDPSFLGRY	244	0.19
TVATSRTLSYYK	152	0.12	TVLSFCAFAV	613	0.48
VEAEVQIDRLITGR	163	0.13	VLHSYFTSDYYQLY	483	0.38
VLPFNDGVYFASTEK	1297	1.02	VLPPLLTDEMIAQYT	674	0.53
VLWAHGFEL	731	0.58	VPHVGEIPVAYRKVLL	528	0.42
VQELYSPIFLIV	1063	0.84	VTEHDTLLY	275	0.22
VYSTGSNVFQTR	286	0.23	WICLLQFAY	590	0.46
YEDFLEYHDVRVVL	874	0.69	YEQYIKWPWYI	537	0.42
YFPLQSYGF	398	0.31	YIFFASFYY	353	0.28
YLDAYNMMI	221	0.17	YLNTLTLAV	432	0.34
YLQPRTFLL	687	0.54	YSEHPTFTSQY	131	0.1
YTMADLVYAL	216	0.17	YVLDHLIVV	8184	6.45
YVVDDPCPIHFY	248	0.2	YYVGYLQPRTFLL	365	0.29

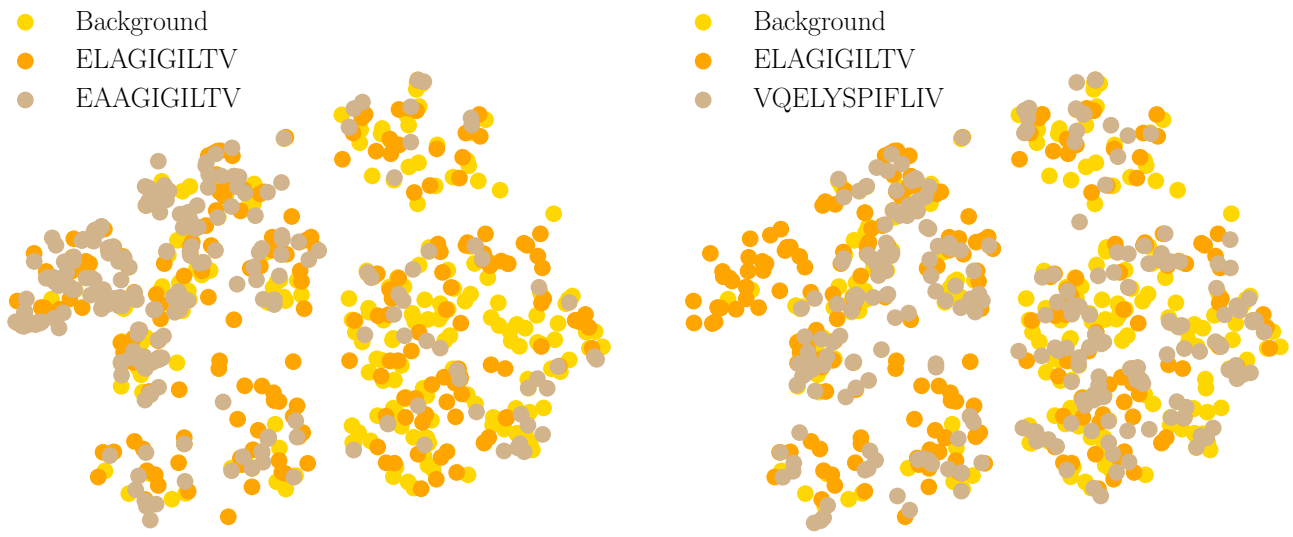


Fig. S3. Out-of-sample performances are related to the distribution of features. tSNE visualization of the feature vectors (in the second-last layer of the classifier architecture) of out-of-sample data for a model trained over the WT epitope ELAGIGILTV, with a balanced dataset containing 1500 CDR3β per class. Top: As the WT epitope, EAAGIGILTV is responsible for the Melanoma cancer and thus is targeted by TCRs sharing similar features. Bottom: the VQELYSPIFLIV peptide is 8 mutations away from the WT and is involved in SARS-CoV-2 infections. Our model predictions are reliable in the first epitope (AUC = 0.79), and are not for the second peptide (AUC = 0.54). The tSNE plots support the claim that out-of-sample specificity predictions drop when the feature vectors of the test data are far away from the training ones.

finding a good feature map where the input data points are well clustered based on their class identities. How does our restoring balance procedure via generative models of sequences affect the supervised learning process? We aim to visualize such effect to seek for a geometrical interpretation of re-balancing data points among different classes.

In particular, our supervised architectures used for both peptide- and pan-specific analyses employ a fully connected read out layer at the end of multiple convolutional layers. Thus, the model applies a series of transformations to map the input dataset into an higher dimensional space, which is passed to a fully connected layer where an hyperplane or a set of intersecting hyperplanes are found to perform binary or multi-class classification, respectively. Thanks to their simplicity, models with a single dense layer like Support Vector Machines (SVMs) allow for a mathematical formulation and some predictions on the model performance can be derived at theoretical level; within this framework, also the issue of learning under imbalanced datasets can be studied. For example, in (66) the authors showed on common imbalanced benchmark datasets that undersampling helps classification as opposed to imbalanced learning; also, in (19) we characterized the performances of an SVM under imbalance and studied the benefit of restoring data balance via under- and oversampling on synthetic data, showing that augmenting the under represented classes yields best performances. The key finding of such studies – that focus on binary classification – proves that learning under imbalance shifts the optimal decision boundary towards the under represented class and tilts it away from the optimal direction (see Figure S7).

To visualize this effect, we consider a simple 1-hidden-layer network with linear activation and sigmoidal output, which we refer to as 1-Dense Network (1DN), and the CNN model considered above. In the two cases, the input-output map is given by

$$y = \sigma\left(W \cdot H\right) , \qquad [17]$$

where W are the weights of the fully connected readout layer and the feature vector H is given by $H = W^{(0)}X$ (1DN) or $H = f_{\text{CNN}}(X)$ (CNN). We train both our supervised CNN model and the 1DN model with a binary classification task (LLWNGPMAV-specific and bulk CDR3βs) using hinge loss. After training on the same dataset, we feedforward the test datapoints in the two models: we call $H_\pm$ the two class centers of the test set,

$$H_\pm = \langle H \rangle_\pm , \qquad [18]$$

where the mean is performed over the two test set classes. The weight vector represents the direction of the decision boundary and should be aligned to the distance vector connecting the classes, $H_+ - H_-$; we quantify this alignment computing the normalized dot product

$$\varphi = \frac{W \cdot (H_+ - H_-)}{\|W\| \|H_+ - H_-\|}, \qquad [19]$$

for both models, φ^{CNN} and φ^{1DN}. Our hypotheses on the geometrical effect of learning with imbalance implies that we should find a value of φ closer to 1 when the model is trained over balanced datasets. Thus we run experiments on different training set compositions, tuning the fraction of negative examples (bulk CDR3βs) in the range $[50\%, 75\%]$, averaging the quantity φ over 50 trials. Results on the simple 1DN and the deep CNN are in agreement, with the dot product dropping from $\varphi^{\text{1DN}} = 0.87$ (ACC $= 0.62$) to $\varphi^{\text{1DN}} = 0.78$ (ACC $= 0.5$) for the 1DN and from $\varphi^{\text{CNN}} = 0.73$ (ACC $= 0.66$) to $\varphi^{\text{CNN}} = 0.62$ (ACC $= 0.5$) for the CNN.

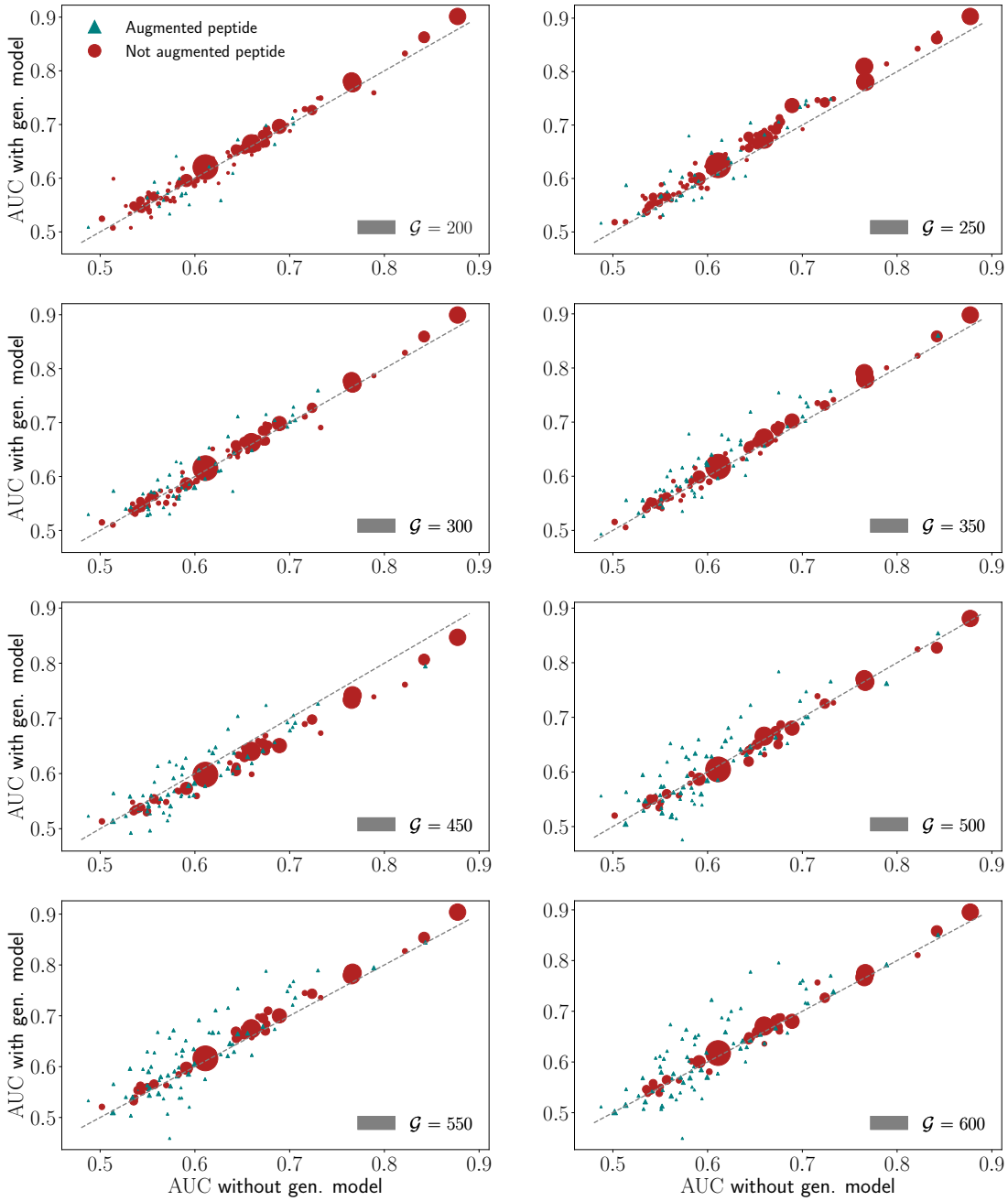


Fig. S4. AUC scores dependencies on the hyperparameter choice M_S. We report results of numerical experiments for the TCR-peptide binding prediction task, comparing AUC performances before and after the generative model has been used (x and y axis, respectively). The data point size is proportional to the size of natural peptide-specific sequences in the dataset. All the training parameters are fixed for all experiments; we rescale the number of training epochs with the training dataset size so that for each experiment we minimize the loss function exactly the same number of times: this factors out all elements, but the dependence of performances on the threshold value $\mathcal{G}$.

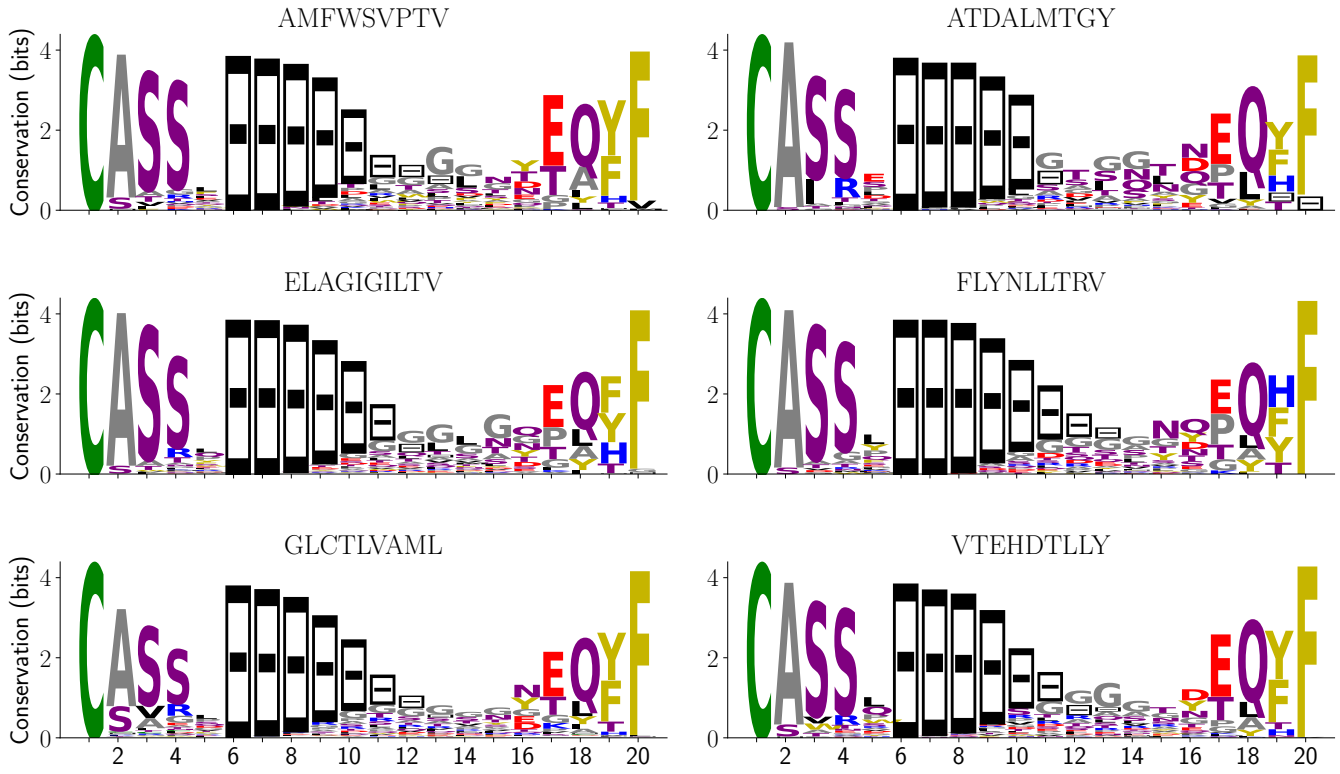


Fig. S5. Sequence logos of peptide-specific CDR3β classes. We report the sequence logos profile for peptide-specific classes of CDR3β sequences, after alignment to maximal length $L = 20$. The PM is learnt over such profiles for each class. Sequence logos show high conservation of the CASS motif and of the last amino acid, while there is more variability in the central region which is indeed responsible for the binding affinity.

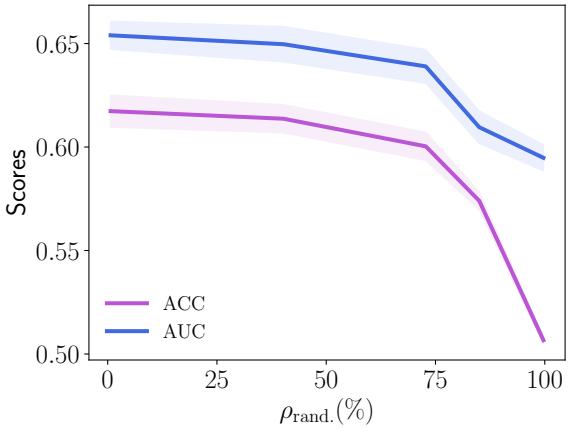


Fig. S6. Performance drop due to randomness effect. Here we report AUC and ACC scores for a VTEHDTLLY-specific model that has been trained over a dataset containing a fraction of random sequences $\rho_{\text{rand.}}$. When the generative model is not good enough and is adding noise to the under represented class of data, we can observe the performances drop down to a random classifier.

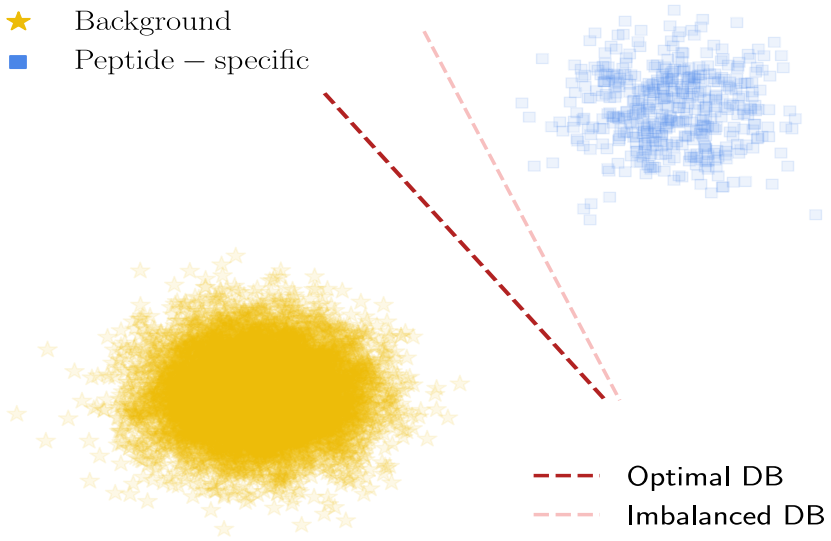


Fig. S7. Geometrical interpretation of learning under imbalance. Graphical visualization of a strongly imbalanced training set, in a binary classification setting. The optimal decision boundary (dark red line) sits perpendicularly at the middle of the two clusters, but learning the model under such imbalance yields a sub-optimal decision boundary (light red), that is tilted and shifted towards the under represented class (blue one). Restoring balance increases performances as it pushes the decision boundary to the optimal one. In the case of deep networks, datapoints should be regarded as the mapped inputs in the feature space before the linear layer.