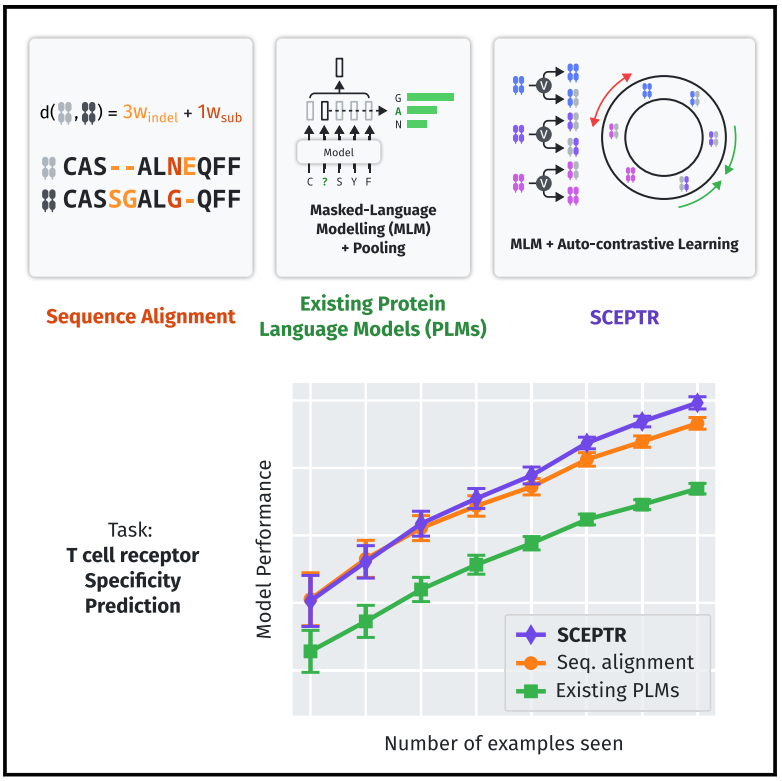
Cell Systems

Contrastive learning of T cell receptor representations

Graphical abstract



Authors

Yuta Nagano, Andrew G.T. Pyo, Martina Milighetti, James Henderson, John Shawe-Taylor, Benny Chain, Andreas Tiffeau-Mayer

Correspondence

andreas.mayer@ucl.ac.uk

In brief

How should we train protein language models to predict protein function? This study shows that pre-training using an autocontrastive loss greatly improves transfer learning compared with masked-language modeling alone for T cell receptor specificity prediction.

Highlights

- Existing language models underperform sequence alignment for predicting TCR specificity
- SCEPTR is a TCR language model that closes this gap via contrastive pre-training
- Contrastive fine-tuning further enhances discrimination between known ligands

Nagano et al., 2025, Cell Systems *16*, 101165
January 15, 2025
https://doi.org/10.1016/j.cels.2024.12.006

CellPress

Cell Systems

CellPress
OPEN ACCESS

Methods

Contrastive learning of T cell receptor representations

Yuta Nagano,[1,2] Andrew G.T. Pyo,[3] Martina Milighetti,[1,4] James Henderson,[1,5] John Shawe-Taylor,[6] Benny Chain,[1,6,7] and Andreas Tiffeau-Mayer[1,5,7,8,*]

[1]Division of Infection and Immunity, University College London, London WC1E 6BT, UK
[2]Division of Medicine, University College London, London WC1E 6BT, UK
[3]Center for the Physics of Biological Function, Princeton University, Princeton, NJ 08544, USA
[4]Cancer Institute, University College London, London WC1E 6DD, UK
[5]Institute for the Physics of Living Systems, University College London, London WC1E 6BT, UK
[6]Department of Computer Science, University College London, London WC1E 6BT, UK
[7]These authors contributed equally
[8]Lead contact
*Correspondence: andreas.mayer@ucl.ac.uk
https://doi.org/10.1016/j.cels.2024.12.006

SUMMARY

Computational prediction of the interaction of T cell receptors (TCRs) and their ligands is a grand challenge in immunology. Despite advances in high-throughput assays, specificity-labeled TCR data remain sparse. In other domains, the pre-training of language models on unlabeled data has been successfully used to address data bottlenecks. However, it is unclear how to best pre-train protein language models for TCR specificity prediction. Here, we introduce a TCR language model called SCEPTR (simple contrastive embedding of the primary sequence of T cell receptors), which is capable of data-efficient transfer learning. Through our model, we introduce a pre-training strategy combining autocontrastive learning and masked-language modeling, which enables SCEPTR to achieve its state-of-the-art performance. In contrast, existing protein language models and a variant of SCEPTR pre-trained without autocontrastive learning are outperformed by sequence alignment-based methods. We anticipate that contrastive learning will be a useful paradigm to decode the rules of TCR specificity. A record of this paper's transparent peer review process is included in the supplemental information.

INTRODUCTION

Antigen-specific T cells play important protective and pathogenic roles in human disease.[1] The recognition of peptides presented on major histocompatibility complexes (pMHCs) by $\alpha\beta$ T cell receptors (TCRs) determines the specificity of cellular immune responses.[2] Hyperdiverse $\alpha\beta$ TCRs are generated during T cell development in the thymus by genetic recombination of germline-encoded V, D (for TCRβ), and J gene segments with additional diversification by trimming of the germ line and insertions of non-template nucleotides at gene segment junctions.

A major goal of systems immunology is to uncover the rules governing which TCRs interact with which pMHCs.[3] Advances in high-throughput functional assays of TCR specificity[4–6] have made the use of machine learning a promising prospect to discover such rules.

The most direct approach for applying machine learning to TCR specificity prediction has been to train pMHC-specific models that take an arbitrary TCR and predict binding.[7–12] More ambitiously, model architectures have been proposed that can in principle generalize predictions to arbitrary pMHCs as well.[13–24] Independent benchmarking studies have shown that both approaches are effective for predicting TCR binders against pMHCs for which many TCRs have been experimentally determined,[25] but generalization to pMHCs not seen during training has largely remained elusive[26] and prediction accuracy is limited for pMHCs with few known binders.[27] This severely limits the utility of current predictive tools given that to date, only $\sim 10^3$ of the $>10^{15}$ possible pMHCs are annotated with any TCRs in VDJdb,[28] and given that for >95% of them fewer than 100 specific TCRs are known.

Meanwhile, there is abundant unlabeled TCR sequence data that may be exploited for unsupervised representation learning. A TCR representation model that compactly captures important features would provide embeddings useful for data-efficient training of downstream specificity predictors.

In natural language processing (NLP), unsupervised pre-trained transformers have demonstrated capacity for transfer learning to diverse downstream tasks.[29–31] This has spurred substantial work applying transformers to protein analysis. Protein language models (PLMs) such as those of the ESM[32,33] and ProtTrans[34] families have been successfully used in

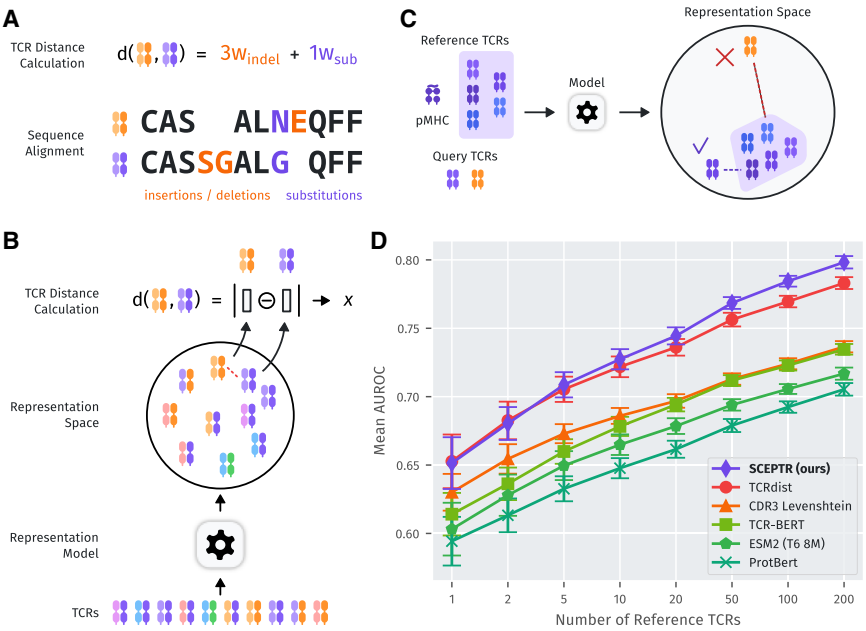


Figure 1. Benchmarking TCR language models against sequence alignment-based approaches on few-shot TCR specificity prediction

(A) TCR similarity can be quantified using sequence alignment by taking a (weighted) count of how many sequence edits turn one TCR into another.

(B) Learned sequence representations allow alignment-free sequence comparisons based on distances in the embedding feature space.

(C) Sketch of our standardized benchmarking approach to allow side-by-side comparison of sequence-alignment and embedding methods. Using a reference set of known TCR binders to a pMHC of interest, we propose nearest neighbor prediction as a task for unbiased comparison of the quality of embeddings for specificity prediction.

(D) Performance of six different models on TCR specificity prediction as a function of the number of reference TCRs. Specificity predictions were made by the nearest neighbor method sketched in (C) against six different pMHCs and performance is reported as the AUROC averaged across the pMHCs. The error bars represent standard deviations of model AUROCs relative to the average across all models within a data split.

structure-prediction pipelines and for protein property prediction.[35–37] PLMs have also been applied to TCR-pMHC interaction prediction,[11,22–24] and the related problem of antibody-antigen interaction prediction.[38,39] However, there has been limited systematic testing of how competitive PLM embeddings are in the *few-shot* setting typical for most ligands—that is, where only few labeled data points are available for transfer learning.

To address this question, we benchmarked existing PLMs on a standardized few-shot specificity prediction task and found that they are inferior to state-of-the-art sequence alignment-based methods. This motivated us to develop SCEPTR (simple contrastive embedding of the primary sequence of T cell receptors), a TCR PLM, which closes this gap. Our key innovation is a pretraining strategy involving an autocontrastive learning procedure adapted for $\alpha\beta$ TCRs, which we show is the primary driver behind SCEPTR's improved performance.

RESULTS

Benchmarking PLM embeddings on TCR specificity prediction

Given the scarcity of specificity-labeled TCR data, it is of practical importance to evaluate model performance where access to such data is limited. Therefore, we set up a benchmarking framework focused on few-shot TCR specificity prediction.

To conduct our benchmark, we curated a set of specificity-labeled $\alpha\beta$ TCR data from VDJdb.[28] We only included human TCRs with full α and β chain information and excluded data from an early 10x Genomics whitepaper,[40] as there are known issues with data reliability in this study.[41,42] This left us with a total of 7,168 $\alpha\beta$ TCRs annotated to 864 pMHCs. Of these, we used the six pMHCs with greater than 300 distinct binder TCRs for our benchmarking task.

We created a benchmarking task that allowed us to directly compare sequence alignment-based distance metrics such as the state-of-the-art TCRdist[4,43] (Figure 1A) to distances in PLM embedding spaces (Figure 1B). For each pMHC, we tested models on their ability to distinguish binder TCRs from non-binders using embedding distances between a query TCR and its closest neighbor within a reference set (Figure 1C). We call this *nearest neighbor prediction*. This framework is simple and attractive for benchmarking models in the few-shot regime, since it remains well defined for as few as a single reference TCR and does not require model specific fine-tuning.

We conducted multiple benchmarks for each pMHC, varying the number of its cognate TCRs used as the reference set. In each case, we combined the remaining TCRs for the target with the rest of the filtered VDJdb dataset (including TCRs annotated to pMHCs other than the six target pMHCs) to create a test set (see STAR Methods). By studying how performance depends on the size of the reference set, we are effectively probing representation alignment with TCR co-specificity prediction at different scales.

We benchmarked six models: two alignment-based TCR metrics (complementarity-determining region 3 or CDR3 Levenshtein distance and TCRdist[4]), two general-purpose PLMs (ProtBert[34] and ESM2[33]), and two TCR domain-specific language models (TCR-BERT[11] and our own model SCEPTR). We report performance using the area under the receiver operator characteristic (AUROC) averaged over the tested pMHCs.

Our results show that TCR-BERT, ESM2, and ProtBert all fail to outperform the baseline sequence alignment method (CDR3 Levenshtein) and are significantly inferior to TCRdist (Figures 1D and S1). A repeat of the benchmarking with a broader set of epitopes obtained by including post-processed 10x Genomics whitepaper data[42] recapitulated these results, demonstrating

CellPress
OPEN ACCESS

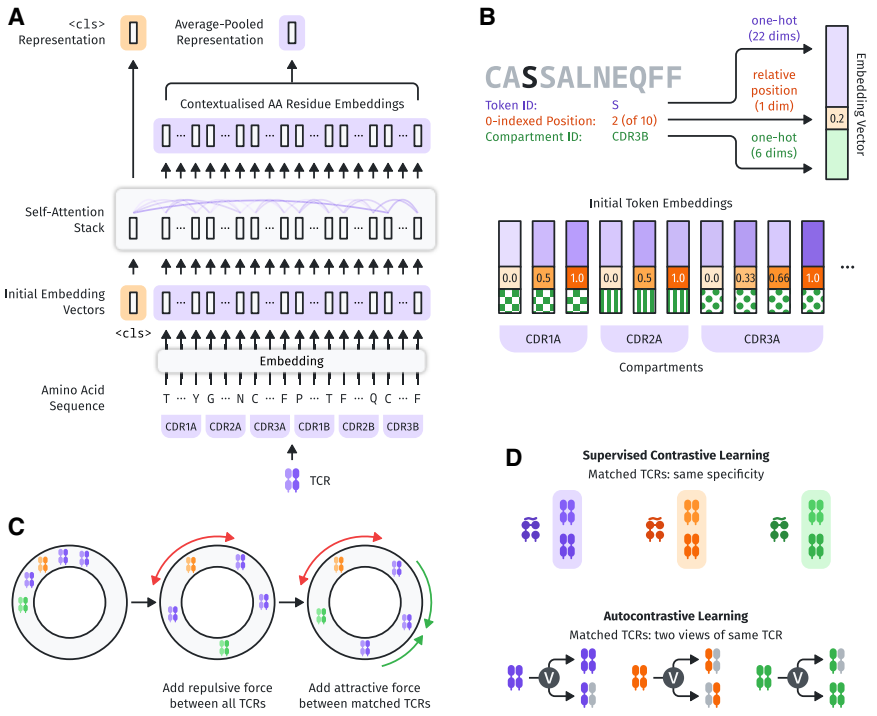


Figure 2. A visual introduction to how SCEPTR works

(A) SCEPTR featurizes an input TCR as the amino acid sequences of its six CDR loops. Each amino acid residue is vectorized to $\mathbb{R}^{64}$ (see B) and are passed along with the special <cls> token vector through a stack of three self-attention layers. SCEPTR uses the contextualized embedding of the <cls> token as the overall TCR representation, in contrast to the average-pooling representations used by other models.

(B) SCEPTR's initial token embedding module uses a simple one-hot system to encode a token's amino acid identity and CDR loop number and allocates one dimension to encode the token's relative position within its CDR loop as a single real-valued scalar.

(C) Contrastive learning allows us to explicitly optimize SCEPTR's representation mapping for TCR co-specificity prediction. At a high level, contrastive learning encourages representation models to make full use of the available representation space while keeping representations of similar input samples close together.

(D) Contrastive learning generalizes to both the supervised and unsupervised settings. In the supervised setting, positive pairs can be generated by sampling pairs of TCRs that are known to bind the same pMHC. In the unsupervised setting, positive pairs can be generated by generating two independent "views" of the same TCR. We implement this by only showing a random subset of the input data features for every view—namely, we remove a proportion of input tokens and sometimes drop the α or β chain entirely (see STAR Methods).

the robustness of our findings (Figure S2). In contrast to existing PLMs, SCEPTR performs on par with or better than TCRdist (Figures 1D and S1, Table S1). For a reference set of size 200, SCEPTR performs better than TCRdist for five out of six tested peptides ($p = 0.11$, binomial test), and for all six peptides when compared with all other models ($p = 0.015$, binomial test).

We additionally compared models using the average distance between a query TCR and all references, instead of only the nearest neighbor (Figure S3). In this case, SCEPTR outperforms other models by a wider margin. We also find that all models perform worse compared with their nearest neighbor counterpart. This finding might be explained mechanistically by the multiplicity of viable binding solutions with distinct sequence-level features, which are thought to make up pMHC-specific TCR repertoires.[44,45] We hypothesized that the poor performance of prior PLMs might be overcome by learning projections from their high-dimensional representation space that align better with the TCR co-specificity prediction task. We thus used the embeddings as input to linear support vector classifiers (see STAR Methods). Optimized linear probing does improve prediction performance (Figure S4), but accuracy remains inferior to nearest neighbor predictions using SCEPTR, further illustrating the usefulness of SCEPTR embeddings for data-efficient transfer learning.

In some use cases, we may want to apply SCEPTR to the analysis of single-chain TCR data. In benchmarking on either α or β chain reference data alone, prediction accuracy drops somewhat regardless of distance measure (Figure S5), as expected given that both receptor chains provide non-redundant information about specificity.[45] However, SCEPTR distances continue to provide comparable or better prediction accuracy than TCRdist on the single-chain level.

Autocontrastive learning as a pre-training strategy

We now briefly summarize SCEPTR's architecture and autocontrastive pre-training strategy (see STAR Methods for full details). SCEPTR featurizes an input TCR as the amino acid sequences of its six CDR loops. It uses a simple one-hot encoding system to embed the amino acid tokens and uses a stack of three self-attention layers to generate a 64-dimensional representation vector of the input receptor (Figures 2A and 2B). Unlike existing TCR language models, SCEPTR is jointly pre-trained using autocontrastive and masked-language modeling (MLM) (Figures 2C and 2D).

To motivate the considerations that have led us to adopt this training paradigm, some background on transformer architectures and their training by MLM needs to be introduced. The transformer is a neural network developed in NLP that uses dot-product attention to flexibly learn long-range dependencies in sequential data.[29] BERT is an encoder-only variation of the transformer useful for text analysis and processing.[30] BERT's innovation was its ability to be pre-trained in an unsupervised manner through MLM, where snippets of text are fed to the model with a certain proportion of tokens (e.g., words) masked, and the model must use the surrounding context to reconstruct the masked tokens. MLM allowed BERT and its derivative models to exploit large volumes of unlabeled data to learn grammar and syntax and achieve high performance on downstream textual tasks with comparatively little supervised fine-tuning.

While MLM-trained PLMs have been successful in some protein prediction tasks,[32–34] they have been documented to struggle with others.[37] Our benchmarking results led us to believe that MLM pre-training may not be optimal for TCR-pMHC specificity prediction. First, the majority of observed TCR sequence variation is attributable to the stochastic process of VDJ recombination. As such, MLM may not teach models much transferable knowledge for specificity prediction. Second, since the low volume of specificity-labeled TCR data provides limited opportunities for fine-tuning complex models, representation distances should ideally be directly predictive of co-specificity.

We were inspired to use contrastive learning to overcome these problems by the success of our previous work using statistical approaches to uncover patterns of sequence similarity characteristic of ligand-specific TCR repertoires.[44–46] Contrastive learning minimizes distances between model representations of positive sample pairs while maximizing distances between background pairs (Figure 2C) through a loss function of the following form[47,48]:

$$\mathcal{L}_{\text{contrastive}}(f) := \mathop{\mathbb{E}}_{\substack{(x,x^+)\sim p_{\text{pos}} \\ \{y_i\}_{i=1}^{N} \overset{\text{iid}}{\sim} p_{\text{data}}}} \left[-\log \frac{e^{f(x)^\top f(x^+)}}{e^{f(x)^\top f(x^+)} + \sum_i e^{f(x)^\top f(y_i)}} \right] \quad \text{(Equation 1)}$$

where $f : \mathcal{X} \to \mathcal{S}^{m-1}$ is a trainable embedding mapping from sample observation space $\mathcal{X}$ to points on the m-dimensional unit hypersphere $\mathcal{S}^{m-1} \subset \mathbb{R}^m$, p_{pos} is the joint distribution of positive pairs, p_{data} is the overall data distribution, and $N \in \mathbb{Z}_+$ is some fixed number of background samples.

There are several well-known variants of this learning approach. In supervised contrastive learning, positive pairs are generated by sampling observations known to belong to the same class (Figure 2D, top). In the context of TCRs, we can define positive pairs to be TCRs annotated to interact with the same pMHC, in which case contrastive learning regresses distances between TCR pairs to their probabilities of co-specificity. Autocontrastive learning approximates such positive pairs through data augmentation by generating two independent "views" of the same observation (Figure 2D, bottom).

Given the scarcity of available labeled data, we opted to use the autocontrastive approach for purely unsupervised PLM pre-training (see supervised contrastive learning as a fine-tuning strategy for an application of supervised contrastive learning, more similar to other recent applications of contrastive learning to TCRs[49,50]). For this pre-training, we used data on close to a million unique paired TCRs obtained by Tanno et al.,[51] which represents one of the largest collections of $\alpha\beta$ TCRs from a single study collected to date. These data were filtered and standardized as described in STAR Methods. We generate different "views" of a TCR by dropout noise as is standard in NLP,[52] but additionally adopted a censoring strategy inspired by MLM that randomly removes a proportion of residues or even complete α or β chains. In contrast to the only other study known to us having explored the application of autocontrastive learning to TCRs,[53] we trained SCEPTR on all six hypervariable loops of the full paired-chain $\alpha\beta$ TCR, as all contribute to TCR-pMHC specificity.[45] That being said, our chain-dropping procedure during censoring ensures that single-chain data are also in distribution for the model, giving SCEPTR flexibility for downstream applications with bulk sequenced TCR repertoires (Figure S5).

We define SCEPTR's output representation vector to be a contextualized embedding of a special input token called <cls> (the naming convention for <cls> comes from the fact that the output of this vector is often used for downstream *classification*[30]), which is always appended to the tokenized representation of an input TCR (Figure 2A). This allows SCEPTR to fully exploit the attention mechanism when generating the overall TCR representation. Such training of a sequence-level representation is uniquely made possible by having an objective—the contrastive loss (Equation 1)—that directly acts on the representation output. In contrast, MLM-trained PLMs such as ProtBert, ESM2, and TCR-BERT generate sequence embeddings by average pooling the contextualized embeddings of each input token at some layer—a destructive operation, which risks diluting information.[37]

Ablation studies

To understand which modeling choices drive the improved performance of SCEPTR, we trained variants of SCEPTR ablating a single component of either its architecture or training at a time and benchmarked them using the framework described previously.

To establish the contribution of the autocontrastive learning to SCEPTR's performance, we trained the same model only using MLM.

> **SCEPTR (MLM only):** This variant is trained only on MLM, without jointly optimizing for autocontrastive learning. Following convention in the transformer field,[54] TCR representation vectors are generated by average pooling the contextualized vector embeddings of all constituent amino acid tokens produced by the penultimate self-attention layer, and $\ell 2$-normalizing the result.

The MLM-only variant underperforms compared with both SCEPTR and TCRdist, demonstrating that autocontrastive learning is a necessary ingredient for the increased performance of SCEPTR in few-shot specificity prediction (Figure 3A).

We next sought to determine how much our pooling strategy and training dataset choice contributed to SCEPTR's performance gain. First, we asked whether autocontrastive learning also improves embeddings generated via token average pooling.

> **SCEPTR (average pooling):** This variant receives both autocontrastive learning and MLM but uses the average-pooling method to generate TCR representations.

While SCEPTR's <cls> embeddings achieve the best results, the autocontrastive average-pooling variant still performs on par with TCRdist (Figure 3B).

Second, we determined how the performance of SCEPTR depends on the precise dataset used for pre-training. To answer this question, we trained two variants of SCEPTR using size-matched datasets.

> **SCEPTR (synthetic data):** This variant is trained on a size-matched set unlabeled $\alpha\beta$ TCRs generated by OLGA,[55] a

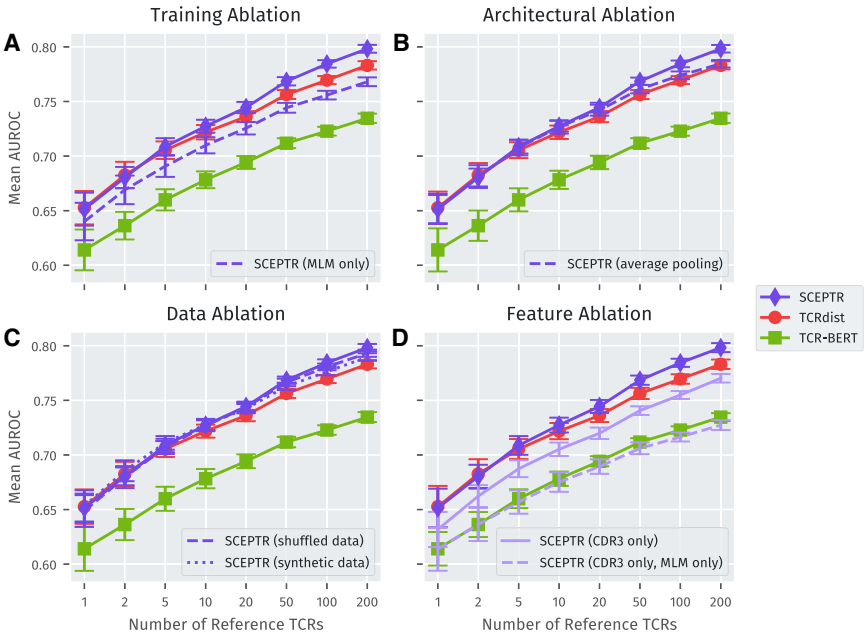


Figure 3. Autocontrastive pre-training significantly improves SCEPTR's downstream performance

The subplots show performance profiles of SCEPTR, TCRdist, TCR-BERT, and various ablation variants of SCEPTR on binary specificity prediction.

(A) Training SCEPTR solely on MLM results in worse specificity prediction performance.

(B) The baseline SCEPTR variant, which uses the <cls> pooling method performs marginally better than the variant which uses the average-pooling method. However, the average-pooling variant still performs on par with TCRdist.

(C) Replacing SCEPTR's pre-training dataset with (1) the same dataset from Tanno et al., but with α/β chain pairing shuffled, and (2) synthetic data generated by OLGA both result in similar specificity prediction performance.

(D) Restricting SCEPTR's featurization of input TCRs to the amino acids of the α and β CDR3 loops significantly worsen downstream performance. Additionally restricting training to only MLM further degrades performance and produces a model with a near-equivalent performance profile to TCR-BERT.

probabilistic model of VDJ recombination. These synthetic data model only the recombination statistics and thus estimate the TCR distribution without taking into account the imprints of thymic and peripheral selection found in real repertoires.[44]

SCEPTR (shuffled data): This variant is trained on the same set of $\alpha\beta$ TCRs as the original model, but the α/β chain pairing is randomized, thus removing pairing biases.[56]

We find that SCEPTR trained on synthetic or shuffled data performs worse for five out of six pMHCs ($p = 0.11$, binomial test), but differences in AUROCs between model variants are small and regardless of training data SCEPTR performs on par with TCRdist (Figure 3C).

Taken together, these ablation studies provide evidence that autocontrastive learning is the main factor enabling SCEPTR to close the gap between PLMs and alignment-based methods.

Information-theoretic analysis of the sequence determinants of TCR specificity demonstrates that all CDR loops and their pairing are important for determining binding specificity.[45] To understand how much SCEPTR's improved performance with respect to TCR-BERT is due to the restriction of the latter model's input to the CDR3 alone, we trained variants of SCEPTR restricted to this hypervariable loop.

SCEPTR (CDR3 only): This variant only accepts the α and β chain CDR3 sequences as input (without the knowledge of the V genes/first two CDR loops of each chain). It is jointly optimized for MLM and autocontrastive learning.

SCEPTR (CDR3 only, MLM only): This otherwise equivalent variant is only trained using the MLM objective and thus uses the average-pooling representation method.

The results demonstrate that taking into account all CDR loops leads to a performance gain as expected (Figure 3D). We also see that autocontrastive learning even when restricted to CDR3s leads to a substantial performance gain, helping the autocontrastive CDR3 variant achieve similar performance to the full-input MLM-only variant.

Comparison of SCEPTR embeddings to alignment-based TCR similarity

To gain insights into what SCEPTR has learned during pre-training, we compared its embedding distances to alignment-based distances as calculated by TCRdist. To do so, we calculated all pairwise distances between one thousand TCRs randomly sampled from the testing partition of the Tanno et al. dataset[51] (held out during SCEPTR pre-training) using both models. We find that SCEPTR and TCRdist distances are clearly correlated (Figure 4A). This shows that, in parts, the success of SCEPTR can be understood by its embedding distances providing good alignment-free approximations to traditional sequence similarity measures. Yet, there is also substantial variability between both measures for many pairs, and an inspection of such discordant pairs can provide insights into how the metrics differ.

First, we noticed that pairs of sequences judged to be similar by SCEPTR but not TCRdist were less likely to be generated during VDJ recombination (Figure 4B). We found that among sequence pairs judged to be similar by SCEPTR, TCRdist distance was strongly negatively correlated with p_{gen} (Figures 4C and S6). This implies that SCEPTR embeds TCRs closer to each other if they are in regions of sequence space that are less densely sampled by the generative distribution. As argued in detail in the discussion, this property of SCEPTR embeddings is expected on theoretical grounds due to the loss function used

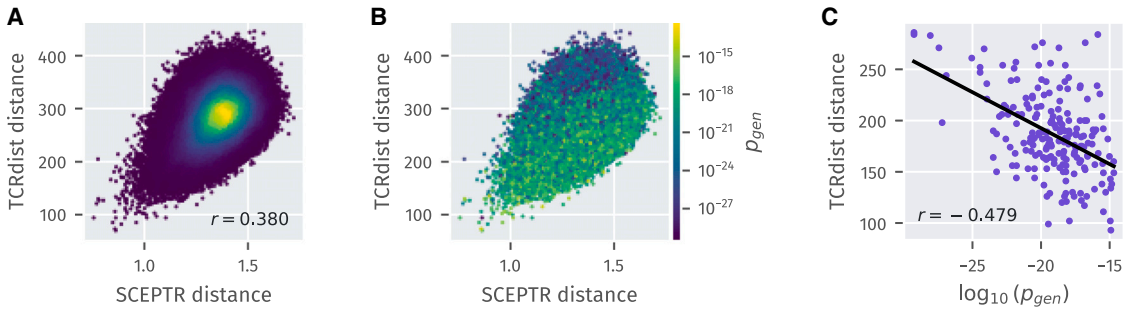


Figure 4. SCEPTR embedding distances weight sequence similarity with respect to recombination biases

(A) Scatter plot of SCEPTR and TCRdist distances between 1 million pairs of TCRs from the held-out test set of the pre-training dataset. The points are colored according to a Gaussian kernel density estimate (linear color scale).

(B) Coloring TCR pairs instead by the minimal probability of generation p_{gen} of the two TCRs as estimated by OLGA[55] (log color scale) suggests that SCEPTR embeddings locally contract regions of representation space that due to recombination biases are sparsely sampled.

(C) For sequence pairs judged to be similar by SCEPTR (distance $\in [0.98, 1.02]$), variations in p_{gen} explain a substantial fraction of the variance in TCRdist, providing statistical evidence for the hypothesized weighting of sequence similarity with respect to the local density of sequences produced by VDJ recombination (see Figure S6 for the generality of this dependence across SCEPTR bins).

for contrastive learning. This property might enable SCEPTR embeddings to capture the intuition that finding close-by nearest neighbors is more surprising for sequences with low p_{gen} and thus more informative compared with similarity between highly probable TCRs.

Second, we noticed that a high similarity on a single chain tended to be sufficient for a small SCEPTR distance (Figure S7). To quantify this effect, we analyzed how SCEPTR distances correlate with different ways of averaging the α and β chain TCRdist distances into a paired-chain measure. We found that SCEPTR distances correlate more closely with the minimum distance of the two chains rather than their arithmetic mean (Figure S8). Further investigation might focus on whether this property helps prediction performance due to the varying contributions of the TCR α and β chain to specific binding across pMHCs.[45]

Supervised contrastive learning as a fine-tuning strategy

Supervised contrastive learning provides an avenue to further optimize pre-trained embeddings for TCR specificity prediction. As a proof of concept, we fine-tuned SCEPTR to better discriminate between the six pMHC specificities used as the benchmarking targets in benchmarking PLM embeddings on TCR specificity prediction.

For this task, we took all the TCRs annotated against the target pMHCs from our labeled TCR dataset and split them into a training, a validation, and a testing set. We ensured that no study used for training or validation contributed any data to the test set, so that the fine-tuned model would not be able to achieve good performance simply by exploiting inter-dataset biases. The training set included 200 binders against each target pMHC, totaling to 1,200 TCRs. The rest of the TCRs from the same studies were used to construct the validation set. TCRs from all remaining studies were used for the testing set, which composed of 5,670 TCRs. SCEPTR was fine-tuned on the training set with supervised contrastive learning, using the validation loss for early stopping.

We used the framework from benchmarking PLM embeddings on TCR specificity prediction to benchmark the performance of fine-tuned SCEPTR, using the training set as the references. The results show that fine-tuning can greatly improve the ability of the model to discriminate between pMHCs (Figure 5). Improvements are most noticeable for the pMHCs against which other methods achieve relatively low performance. When filtering all TCRs with greater than 90% or 80% sequence similarity to any training sequence from the test set, the fine-tuned model still improves performance significantly (Figure S9) showing that learning goes beyond memorization of public TCRs.

Unlike other models, fine-tuned SCEPTR makes better inferences by measuring the average distance between a query TCR and all reference TCRs instead of only the nearest TCR (Figure S10). This suggests an ability of supervised contrastive fine-tuning to help the model discover the commonalities between the multiple different binding solutions thought to exist for each pMHC. We thus analyzed how fine-tuning changes the SCEPTR embedding distances between co-specific and cross-pMHC TCR pairs (Figure 6). Unexpectedly, we found that a major difference of fine-tuned SCEPTR distances concerns cross-pMHC pairs. We observe that fine-tuning allows the model to identify a subset of "easy" negative pairs. These presumably involve TCRs that the model is highly confident are specific to different pMHCs, thus illustrating that discrimination between a fixed set of potential target pMHCs is easier than binary classification with respect to TCRs of arbitrary specificity. Conversely, the fine-tuned model's performance degrades with respect to unseen pMHCs (Figure S11), perhaps unsurprisingly given the very limited number of pMHCs represented in the training data.

Another notable feature of Figure 5 is that performance varies substantially by ligand. That is, prediction is easier for some pMHCs, regardless of the method. This is a phenomenon that has also been observed in public benchmarks.[25] Consequently, we used coincidence (order two Rényi) entropy measures[44,45] to discover intrinsic properties of the pMHC-specific TCR repertoires that determine model AUROCs. We find that TCRs annotated against the "easier" pMHCs have lower V/J gene diversity (Figure S12) and lower average sequence distances between pairs of TCRs (Figure S13). This indicates that differences in

models' ability to predict TCR-pMHC specificity are linked to the diversity of epitope-specific repertoires.

DISCUSSION

In this study, we have introduced SCEPTR, a pre-trained TCR PLM that achieves state-of-the-art few-shot TCR-pMHC specificity prediction accuracy. Through SCEPTR, we demonstrate that joint autocontrastive and masked-language pre-training is a paradigm for learning PLMs better aligned with TCR specificity prediction tasks. Our model can be readily used for alignment-free TCR analysis in downstream applications (see data and code availability) including the unsupervised discovery of antigen-specific T cell groups (metaclonotypes) by sequence-based clustering.[4,43]

A limitation of our study is that we did not undertake a complete exploration of training and architectural hyperparameters, as well as training dataset choice. We envisage multiple avenues that may improve SCEPTR further. First, training could be made more efficient by optimizing the distribution of masked/dropped tokens during pre-training, taking into account the variable relevance of different parts of the sequence in determining specificity.[45] Second, as certain sequence motifs appear recurrently (e.g., CDR3 loops often begin with CAS), a more intelligent tokenization scheme could offload learning of these primary sequence statistics into the tokenization process. Finally, with the emergence of increasingly large paired-chain TCR datasets,[57–59] retraining SCEPTR on data from multiple sources could eliminate biases inherent to specific experimental approaches and donor MHC restrictions.

Pre-trained PLMs have achieved high performance on protein stability and structural predictions.[33,34] However, we find that existing PLMs fail to confer similar benefits to predicting TCR-pMHC interactions. This finding adds to recent work showing that current PLM pre-training is not well-aligned with certain downstream tasks.[37] We address this problem by demonstrating that autocontrastive pre-training can overcome misalignment. This provides a constructive path out of this impasse of potential general applicability outside of the TCR domain.

What determines whether a certain downstream task is aligned with MLM pre-training? MLM teaches PLMs to predict the conditional distribution of tokens given sequence context. Thus, it stands to reason that amenable downstream tasks involve predictions of properties that determine the distribution of observed proteins on sequence space. Observed proteins tend to concentrate in areas of sequence space with higher protein stability, as evolution on average selects for this property.[60] For datasets containing protein families whose members have a conserved structure despite primary sequence variation, co-evolutionary couplings driven by structural constraints influence allowed sequence variability.[33,61,62] These data distributional properties might explain how MLM can teach PLMs features related to both stability and structure.

In contrast, the distribution of TCRs over sequence space is primarily shaped by the biases of VDJ recombination with antigen-specific selection playing an important, but likely second-order effect.[44] While long-term evolutionary pressures may act to align recombination statistics with TCR function,[63,64] empirical evidence so far suggests that recombination biases primarily anticipate thymic selection for stability and folding.[65] In contrast, studies to date have found no clear relationship between probabilities of recombination and the likelihood of receptors engaging specific pMHCs.[55] Given these considerations, we expect MLM pre-training to align better to tasks concerning VDJ recombination than to TCR specificity prediction. Indeed, previous work training PLMs on adaptive immune receptors has demonstrated that embeddings strongly depend on V/J gene usage and can be used to predict primarily generation-related properties such as receptor publicity.[24,38]

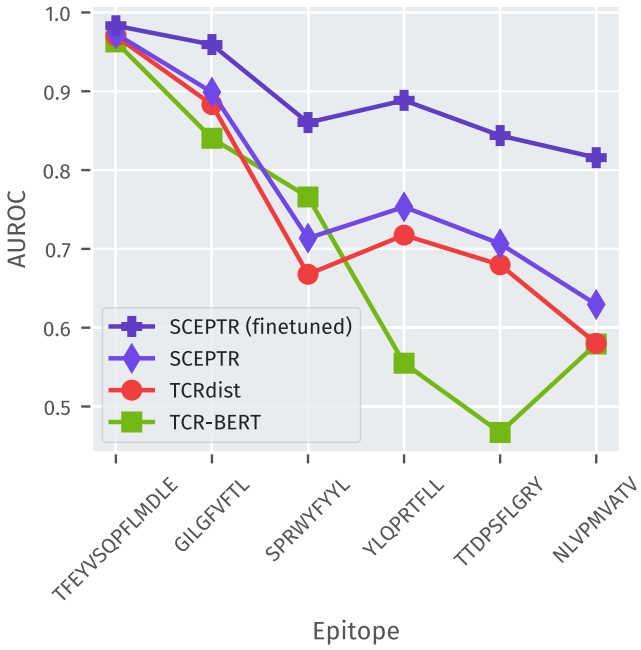


Figure 5. Supervised contrastive learning improves discrimination between pMHCs

Prediction performance as measured by AUROC on binary one-versus-rest classification for each of six pMHCs for different models. The fine-tuned model improves performance by exploiting the discriminative nature of the classification task.

Why does autocontrastive learning help to generate embeddings better suited for specificity prediction? An interesting insight comes from an asymptotic decomposition of the contrastive loss function into the *uniformity* and *alignment* terms[47]:

$$\text{Unif.}(f) := \log \mathop{\mathbb{E}}_{x,y \overset{\text{iid}}{\sim} p_{\text{data}}} \left[e^{-\|f(x)-f(y)\|^2} \right] \qquad \text{(Equation 2)}$$

$$\text{Align.}(f) := \mathop{\mathbb{E}}_{(x,x^+)\sim p_{\text{pos}}} \left[\| f(x) - f(x^+) \| \right] \qquad \text{(Equation 3)}$$

Uniformity incentivizes the model to make use of the full representation space, while alignment minimizes the expected distance between positive pairs (e.g., co-specific TCRs).[47] From this view, contrastive learning on adaptive immune receptor data encourages PLMs to undo the large-scale distributional biases created by VDJ recombination through the uniformity term, while helping to identify features relating to TCR (co-)specificity via the alignment term. While autocontrastive learning approximates the alignment term through the generation of pairs of

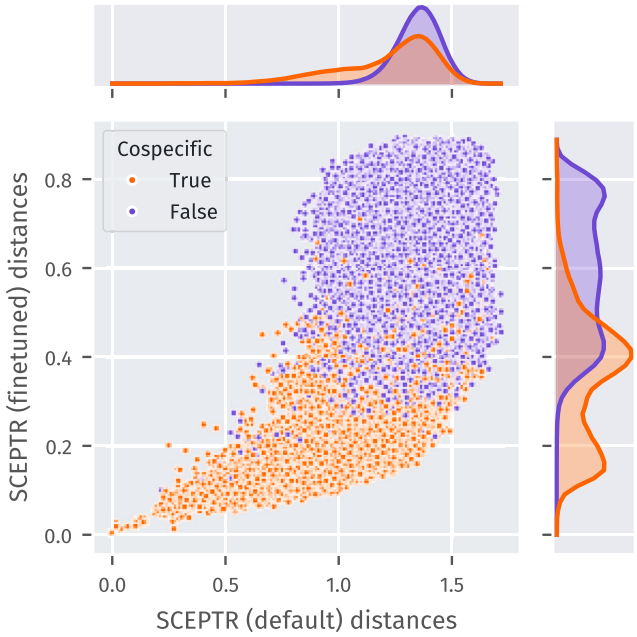


Figure 6. Supervised contrastive learning reshapes the embedding space

Scatter plot of distance between pairs of TCRs as measured by the pre-trained and fine-tuned versions of SCEPTR, and their marginal conditional probability density functions. Points are colored based on whether the corresponding TCR pair involves TCRs that are annotated against the same pMHC (orange) or against different pMHCs (purple). Pairs of TCRs were obtained by sampling one receptor from the training partition and the other from the test partition of the labeled TCR data used during fine-tuning.

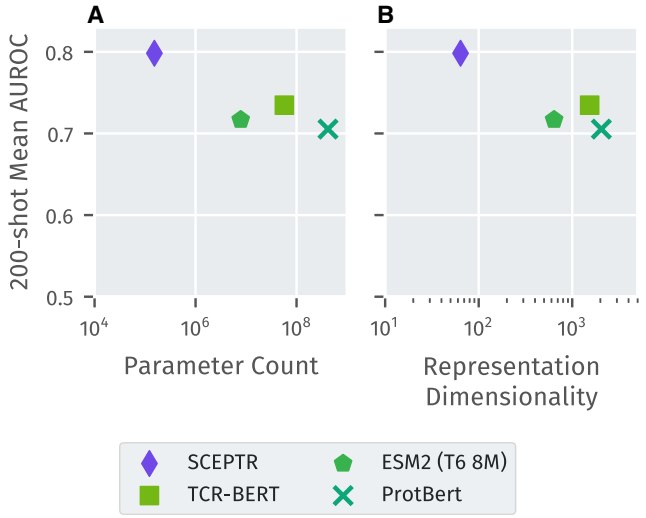


Figure 7. Model complexity does not correlate with downstream performance

Model performance as measured by mean 200-shot AUROC (benchmarking PLM embeddings on TCR specificity prediction) does not scale with model complexity as measured by either (A) parameter count or (B) representation dimensionality. Despite being the smallest PLM by a wide margin, SCEPTR performs better than alternative models.

views, it still provides a direct empirical estimate for the uniformity term. Thus, a key benefit of autocontrastive learning may be that it reduces the confounding effects of VDJ recombination in embedding space. SCEPTR's ability to "adjust" its distances for p_{gen} as demonstrated in Figures 4 and S6 lends support to this conjecture.

Comparing SCEPTR to other PLMs suggests that model complexity as measured by either parameter count or representation dimensionality is not currently the limiting factor for TCR-pMHC prediction performance (Figure 7). This is directly supported by how the CDR3-only, MLM-only variant of SCEPTR performs almost equivalently to the much larger but similarly pre-trained TCR-BERT (Figure 3D). This finding stands in contrast to observations of performance scaling with model size in general PLMs[33] and antibody language modeling.[39] While the focus of the current study was on training a simple model, it would be interesting in future work to investigate performance scaling with model complexity and training dataset size with our training procedure.

Looking forward, there are many exciting avenues to further develop contrastive learning as a paradigm to crack the TCR code. For example, there may be ways to exploit the uniformity (Equation 2) and alignment (Equation 3) decomposition to simultaneously train on unlabeled and specificity-labeled data. A practical benefit of our contrastive learning formulation is that it does not require any optimization with respect to the true negative distribution (i.e., TCRs that are explicitly not co-specific)—a non-trivial distribution to estimate for TCRs.[66]

Another interesting avenue is the use of labels other than pMHC specificity—such as phenotypic annotations from single-cell data—as additional supervised contrastive training signals. While supervised contrastive learning does not currently lead to generalizable learning beyond training pMHCs, we expect a transition toward generalization as larger volumes of specificity-labeled TCR data become available, as has been the case with supervised contrastive learning in other fields.[52,67–70]

Finally, while the focus of this work given current data limitations has been on learning TCR embeddings, contrastive learning may also help us learn effective joint TCR-pMHC embeddings in the future when the joint space (and particularly the pMHC space) is better sampled, and thus ultimately enable the zero-shot prediction of TCR-pMHC specificity.

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources should be directed to and will be fulfilled by the lead contact, Andreas Tiffeau-Mayer (andreas.mayer@ucl.ac.uk).

Materials availability

This study did not generate new materials.

Data and code availability

- A readily usable distribution of our model SCEPTR can be found at https://github.com/yutanagano/sceptr.
- This paper analyzes existing, publicly available data. Links to the datasets are listed in the key resources table.
- All original code has been deposited at Zenodo and is publicly available as of the date of publication. DOIs are listed in the key resources table.
- Any additional information required to reanalyze the data reported in this paper is available from the lead contact upon request.

ACKNOWLEDGMENTS

The authors thank Ned Wingreen, Chris Watkins, Trevor Graham, Sergio Quezada, Machel Reid, Linda Li, Rudy Yuen, Sankalan Bhattacharyya, and Matthew Cowley for useful discussions. Y.N. and M.M. were supported by Cancer Research UK studentships under grants BCCG1C8R and A29287, respectively. The work of A.T.-M. was supported in part by funding from the Royal Free Charity.

AUTHOR CONTRIBUTIONS

Y.N., J.S.-T., B.C., and A.T.-M. designed the project. Y.N. managed the project, wrote the software, visualized the results, and led the formal analysis. Y.N. and A.T.-M. curated the data and wrote the original draft. A.G.T.P., M.M., and J.H. contributed to the formal analysis and development of the methodology. B.C. and A.T.-M. supervised the project. All authors reviewed and edited the paper.

DECLARATION OF INTERESTS

The authors have no competing interests.

STAR★METHODS

Detailed methods are provided in the online version of this paper and include the following:

- KEY RESOURCES TABLE
- METHOD DETAILS
  - SCEPTR architecture
  - SCEPTR Pre-training
  - Fine-tuning by supervised contrastive learning
  - Generating TCR vector embeddings using existing protein language models
  - Learning features within embedding spaces
  - Effects of position embedding methods
- QUANTIFICATION AND STATISTICAL ANALYSIS
- ADDITIONAL RESOURCES

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.cels.2024.12.006.

Received: June 20, 2024
Revised: October 9, 2024
Accepted: December 6, 2024
Published: January 7, 2025

REFERENCES

1. Chi, H., Pepper, M., and Thomas, P.G. (2024). Principles and therapeutic applications of adaptive immunity. Cell *187*, 2052–2078. https://doi.org/10.1016/j.cell.2024.03.037.
2. Davis, M.M., and Bjorkman, P.J. (1988). The T cell receptor genes and T cell recognition. Nature *334*, 395–402. https://doi.org/10.1038/334395a0.
3. Hudson, D., Fernandes, R.A., Basham, M., Ogg, G., and Koohy, H. (2023). Can we predict T cell specificity with digital biology and machine learning? Nat. Rev. Immunol. *23*, 511–521. https://doi.org/10.1038/s41577-023-00835-3.
4. Dash, P., Fiore-Gartland, A.J., Hertz, T., Wang, G.C., Sharma, S., Souquette, A., Crawford, J.C., Clemens, E.B., Nguyen, T.H.O., Kedzierska, K., et al. (2017). Quantifiable predictive features define epitope-specific T cell receptor repertoires. Nature *547*, 89–93. https://doi.org/10.1038/nature22383.
5. Dobson, C.S., Reich, A.N., Gaglione, S., Smith, B.E., Kim, E.J., Dong, J., Ronsard, L., Okonkwo, V., Lingwood, D., Dougan, M., et al. (2022). Antigen identification and high-throughput interaction mapping by reprogramming viral entry. Nat. Methods *19*, 449–460. https://doi.org/10.1038/s41592-022-01436-z.
6. Joglekar, A.V., Leonard, M.T., Jeppson, J.D., Swift, M., Li, G., Wong, S., Peng, S., Zaretsky, J.M., Heath, J.R., Ribas, A., et al. (2019). T cell antigen discovery via signaling and antigen-presenting bifunctional receptors. Nat. Methods *16*, 191–198. https://doi.org/10.1038/s41592-018-0304-8.
7. Gielis, S., Moris, P., Bittremieux, W., De Neuter, N., Ogunjimi, B., Laukens, K., and Meysman, P. (2019). Detection of enriched T cell epitope specificity in full T cell receptor sequence repertoires. Front. Immunol. *10*, 2820. https://doi.org/10.3389/fimmu.2019.02820.
8. Fischer, D.S., Wu, Y., Schubert, B., and Theis, F.J. (2020). Predicting antigen specificity of single T cells based on TCR CDR3 regions. Mol. Syst. Biol. *16*, e9416. https://doi.org/10.15252/msb.20199416.
9. Jokinen, E., Huuhtanen, J., Mustjoki, S., Heinonen, M., and Lähdesmäki, H. (2021). Predicting recognition between T cell receptors and epitopes with TCRGP. PLOS Comput. Biol. *17*, e1008814. https://doi.org/10.1371/journal.pcbi.1008814.
10. Montemurro, A., Schuster, V., Povlsen, H.R., Bentzen, A.K., Jurtz, V., Chronister, W.D., Crinklaw, A., Hadrup, S.R., Winther, O., Peters, B., et al. (2021). NetTCR-2.0 enables accurate prediction of TCR-peptide binding by using paired TCRalpha and beta sequence data. Commun. Biol. *4*, 1060. https://doi.org/10.1038/s42003-021-02610-3.
11. Wu, K., Yost, K.E., Daniel, B., Belk, J.A., Xia, Y., Egawa, T., Satpathy, A., Chang, H.Y., and Zou, J. (2021). TCR-BERT: learning the grammar of T-cell receptors for flexible antigen-xbinding analyses. Preprint at bioRxiv. https://doi.org/10.1101/2021.11.18.469186.
12. Croce, G., Bobisse, S., Moreno, D.L., Schmidt, J., Guillame, P., Harari, A., and Gfeller, D. (2024). Deep learning predictions of tcr-epitope interactions reveal epitope-specific chains in dual alpha T cells. Nat. Commun. *15*, 3211. https://doi.org/10.1038/s41467-024-47461-8.
13. Weber, A., Born, J., and Rodriguez Martínez, M., (2021) T-cell receptor specificity prediction with bimodal attention networks, Bioinformatics 37 (Supplement_1) i237–i244. 10.1093/bioinformatics/btab294.
14. Jiang, Y., Huo, M., and Li, S.C. (2023). TEINet: a deep learning framework for prediction of TCR–epitope binding specificity. Brief. Bioinform. *24*, bbad086. https://doi.org/10.1093/bib/bbad086.
15. Lu, T., Zhang, Z., Zhu, J., Wang, Y., Jiang, P., Xiao, X., Bernatchez, C., Heymach, J.V., Gibbons, D.L., Wang, J., et al. (2021). Deep learning-based prediction of the T cell receptor–antigen binding specificity. Nat. Mach. Intell. *3*, 864–875. https://doi.org/10.1038/s42256-021-00383-2.
16. Lin, X., George, J.T., Schafer, N.P., Chau, K.N., Birnbaum, M.E., Clementi, C., Onuchic, J.N., and Levine, H. (2021). Rapid assessment of T-cell receptor specificity of the immune repertoire, Nat. Comput. Sci. *1*, 362–373. https://doi.org/10.1038/s43588-021-00076-1.
17. Springer, I., Tickotsky, N., and Louzoun, Y. (2021). Contribution of T Cell Receptor Alpha and Beta CDR3, MHC Typing, V and J Genes to Peptide Binding Prediction. Front. Immunol. *12*, 664514. https://doi.org/10.3389/fimmu.2021.664514.
18. Cai, M., Bang, S., Zhang, P., and Lee, H. (2022). ATM-TCR: TCR-Epitope Binding Affinity Prediction Using a Multi-Head Self-Attention Model. Front. Immunol. *13*, 893247. https://doi.org/10.3389/fimmu.2022.893247.
19. Moris, P., De Pauw, J., Postovskaya, A., Gielis, S., De Neuter, N., Bittremieux, W., Ogunjimi, B., Laukens, K., and Meysman, P. (2021). Current challenges for unseen-epitope TCR interaction prediction and a new perspective derived from image classification. Brief. Bioinform. *22*, bbaa318. https://doi.org/10.1093/bib/bbaa318.
20. Pham, M.-D.N., Nguyen, T.-N., Tran, L.S., Nguyen, Q.-T.B., Nguyen, T.-P.H., Pham, T.M.Q., Nguyen, H.-N., Giang, H., Phan, M.-D., and Nguyen, V. (2023). epiTCR: a highly sensitive predictor for TCR–peptide binding. Bioinformatics *39*, btad284. https://doi.org/10.1093/bioinformatics/btad284.

21. Gao, Y., Gao, Y., Fan, Y., Zhu, C., Wei, Z., Zhou, C., Chuai, G., Chen, Q., Zhang, H., and Liu, Q. (2023). Pan-Peptide Meta Learning for T-cell receptor–antigen binding recognition. Nat. Mach. Intell. *5*, 236–249. https://doi.org/10.1038/s42256-023-00619-3.

22. Kwee, B.P.Y., Messemaker, M., Marcus, E., Oliveira, G., Scheper, W., Wu, C.J., Teuwen, J., and Schumacher, T.N. (2023). STAPLER: Efficient learning of TCR-peptide specificity prediction from full-length TCR-peptide data. Preprint at bioRxiv. https://doi.org/10.1101/2023.04.25.538237.

23. Meynard-Piganeau, B., Feinauer, C., Weigt, M., Walczak, A.M., and Mora, T. (2024). TULIP: a Transformer based Unsupervised Language model for Interacting Peptides and T-cell receptors that generalizes to unseen epitopes. Proc. Natl. Acad. Sci. USA *121*, 20–25. https://doi.org/10.1101/2023.07.19.549669.

24. Goldner Kabeli, R., Zevin, S., Abargel, A., Zilberberg, A., and Efroni, S. (2024). Self-supervised learning of t cell receptor sequences exposes core properties for T cell membership. Sci. Adv. *10*, eadk4670. https://doi.org/10.1126/sciadv.adk4670.

25. Nielsen, M., Eugster, A., Jensen, M.F., Goel, M., Tiffeau-Mayer, A., Pelissier, A., Valkiers, S., Martínez, M.R., Meynard-Piganeeau, B., Greiff, V., et al. (2024). Lessons learned from the immrep23 tcr-epitope prediction challenge. ImmunoInformatics *16*, 100045.

26. Grazioli, F., Mösch, A., Machart, P., Li, K., Alqassem, I., O'Donnell, T.J., and Min, M.R. (2022). On tcr binding predictors failing to generalize to unseen peptides. Front. Immunol. *13*, 1014256. https://doi.org/10.3389/fimmu.2022.1014256.

27. Deng, L., Ly, C., Abdollahi, S., Zhao, Y., Prinz, I., and Bonn, S. (2023). Performance comparison of tcr-pmhc prediction tools reveals a strong data dependency. Front. Immunol. *14*, 1128326. https://doi.org/10.3389/fimmu.2023.1128326.

28. Bagaev, D.V., Vroomans, R.M.A., Samir, J., Stervbo, U., Rius, C., Dolton, G., Greenshields-Watson, A., Attaf, M., Egorov, E.S., Zvyagin, I.V., et al. (2020). VDJdb in 2019: database extension, new analysis infrastructure and a T-cell receptor motif compendium. Nucleic Acids Res. *48*, D1057–D1062. https://doi.org/10.1093/nar/gkz874.

29. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., and Polosukhin, I. (2017). Attention Is All You Need. Preprint at arXiv. https://doi.org/10.48550/arXiv.1706.03762.

30. Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Preprint at arXiv. https://doi.org/10.48550/arXiv.1810.04805.

31. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. Adv. Neural Inf. Process. Syst. *33*, 1877–1901.

32. Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C.L., Ma, J., et al. (2021). Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. Proc. Natl. Acad. Sci. USA *118*, e2016239118. https://doi.org/10.1073/pnas.2016239118.

33. Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., et al. (2023). Evolutionary-scale prediction of atomic-level protein structure with a language model. Science *379*, 1123–1130. https://doi.org/10.1126/science.ade2574.

34. Elnaggar, A., Heinzinger, M., Dallago, C., Rehawi, G., Wang, Y., Jones, L., Gibbs, T., Feher, T., Angerer, C., Steinegger, M., et al. (2022). ProtTrans: Toward Understanding the Language of Life Through Self-Supervised Learning. IEEE Trans. Pattern Anal. Mach. Intell. *44*, 7112–7127. https://doi.org/10.1109/TPAMI.2021.3095381.

35. Elnaggar, A., Essam, H., Salah-Eldin, W., Moustafa, W., Elkerdawy, M., Rochereau, C., and Rost, B. (2023). Ankh: Optimized Protein Language Model Unlocks General- Purpose Modelling. Preprint at arXiv.

36. Wu, K.E., Chang, H., and Zou, J. (2024). ProteinCLIP: enhancing protein language models with natural language. Preprint at bioRxiv. https://doi.org/10.1101/2024.05.14.594226.

37. Li, F.-Z., Amini, A.P., Yue, Y., Yang, K.K., and Lu, A.X. (2024). Feature Reuse and Scaling: Understanding Transfer Learning with Protein Language Models. Preprint at bioRxiv. https://doi.org/10.1101/2024.02.05.578959.

38. Wang, M., Patsenker, J., Li, H., Kluger, Y., and Kleinstein, S.H. (2024). Language model-based B cell receptor sequence embeddings can effectively encode receptor specificity. Nucleic Acids Res. *52*, 548–557. https://doi.org/10.1093/nar/gkad1128.

39. Barton, J., Gaspariunas, A., Yadin, D.A., Dias, J., Nice, F.L., Minns, D.H., Snudden, O., Povall, C., Tomas, S.V., Dobson, H., et al. (2024). A generative foundation model for antibody sequence understanding. Preprint at bioRxiv. https://doi.org/10.1101/2024.05.22.594943.

40. 10x Genomics. (2020). Application note: A New Way of Exploring Immunity – Linking Highly Multiplexed Antigen Recognition to Immune Repertoire and Phenotype. https://support.10xgenomics.com/datasets.

41. Zhang, W., Hawkins, P.G., He, J., Gupta, N.T., Liu, J., Choonoo, G., Jeong, S.W., Chen, C.R., Dhanik, A., Dillon, M., et al. (2021). A framework for highly multiplexed dextramer mapping and prediction of T cell receptor sequences to antigen specificity. Sci. Adv. *7*, eabf5835. https://doi.org/10.1126/sciadv.abf5835.

42. Montemurro, A., Povlsen, H.R., Jessen, L.E., and Nielsen, M. (2023). Benchmarking data-driven filtering for denoising of TCRpMHC single-cell data. Sci. Rep. *13*, 16147. https://doi.org/10.1038/s41598-023-43048-3.

43. Mayer-Blackwell, K., Schattgen, S., Cohen-Lavi, L., Crawford, J.C., Souquette, A., Gaevert, J.A., Hertz, T., Thomas, P.G., Bradley, P., and Fiore-Gartland, A. (2021). Tcr meta-clonotypes for biomarker discovery with tcrdist3 enabled identification of public, hla-restricted clusters of sars-cov-2 tcrs. eLife *10*, e68605. https://doi.org/10.7554/eLife.68605.

44. Mayer, A., and Callan, C.G. (2023). Measures of epitope binding degeneracy from T cell receptor repertoires. Proc. Natl. Acad. Sci. USA *120*, e2213264120. https://doi.org/10.1073/pnas.2213264120.

45. Henderson, J., Nagano, Y., Milighetti, M., and Tiffeau-Mayer, A. (2024). Limits on inferring T cell specificity from partial information. Proc. Natl. Acad. Sci. USA *121*, e2408696121. https://doi.org/10.1073/pnas.2408696121.

46. Tiffeau-Mayer, A. (2024). Unbiased estimation of sampling variance for simpson's diversity index. Phys. Rev. E *109*, 064411. https://doi.org/10.1103/PhysRevE.109.064411.

47. Wang, T., and Isola, P. (2022). Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere. Preprint at arXiv. https://doi.org/10.48550/arXiv.2005.10242.

48. Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., and Krishnan, D. (2021). Supervised Contrastive Learning. Preprint at arXiv. https://doi.org/10.48550/arXiv.2004.11362.

49. Drost, F., Schiefelbein, L., and Schubert, B. (2022). meTCRs – Learning a metric for T-cell receptors. Preprint at bioRxiv. https://doi.org/10.1101/2022.10.24.513533.

50. Pertseva, M., Follonier, O., Scarcella, D., and Reddy, S.T. (2024). TCR clustering by contrastive learning on antigen specificity. Preprint at bioRxiv. https://doi.org/10.1101/2024.04.04.587695v1.

51. Tanno, H., Gould, T.M., McDaniel, J.R., Cao, W., Tanno, Y., Durrett, R.E., Park, D., Cate, S.J., Hildebrand, W.H., Dekker, C.L., et al. (2020). Determinants governing T cell receptor alpha/beta-chain pairing in repertoire formation of identical twins. Proc. Natl. Acad. Sci. USA *117*, 532–540. https://doi.org/10.1073/pnas.1915008117.

52. Gao, T., Yao, X., Chen, D., and Sim, C.S.E. (2022). Simple Contrastive Learning of Sentence Embeddings. Preprint at arXiv. https://doi.org/10.48550/arXiv.2104.08821.

53. Fang, Y., Liu, X., and Liu, H. (2022). Attention-aware contrastive learning for predicting T cell receptor–antigen binding specificity. Brief. Bioinform. *23*, bbac378. https://doi.org/10.1093/bib/bbac378.

54. Li, B., Zhou, H., He, J., Wang, M., Yang, Y., and Li, L. (2020). On the sentence embeddings from pre-trained language models. In Proceedings of

the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), B. Webber, T. Cohn, Y. He, and Y. Liu, eds. (Association for Computational Linguistics), pp. 9119–9130. https://doi.org/10.18653/v1/2020.emnlp-main.733.

55. Sethna, Z., Elhanati, Y., Callan, C.G., Jr., Walczak, A.M., and Mora, T. (2019). Olga: fast computation of generation probabilities of b-and T-cell receptor amino acid sequences and motifs. Bioinformatics *35*, 2974–2981. https://doi.org/10.1093/bioinformatics/btz035.

56. Milighetti, M., Nagano, Y., Henderson, J., Hershberg, U., Tiffeau-Mayer, A., Bitbol, A.-F., and Chain, B. (2024). Intra-and inter-chain contacts determine tcr specificity: applying protein co-evolution methods to tcr$\alpha\beta$ pairing. Preprint at bioRxiv.

57. Spindler, M.J., Nelson, A.L., Wagner, E.K., Oppermans, N., Bridgeman, J.S., Heather, J.M., Adler, A.S., Asensio, M.A., Edgar, R.C., Lim, Y.W., et al. (2020). Massively parallel interrogation and mining of natively paired human tcr$\alpha\beta$ repertoires. Nat. Biotechnol. *38*, 609–619. https://doi.org/10.1038/s41587-020-0438-y.

58. Raybould, M.I.J., Greenshields-Watson, A., Agarwal, P., Aguilar-Sanjuan, B., Olsen, T.H., Turnbull, O.M., Quast, N.P., and Deane, C.M. (2024). The Observed T Cell Receptor Space database enables paired-chain repertoire mining, coherence analysis, and language modeling. Cell Rep. *43*, 114704. https://doi.org/10.1016/j.celrep.2024.114704.

59. Sureshchandra, S., Henderson, J., Levendosky, E., Bhattacharyya, S., Kastenschmidt, J.M., Sorn, A.M., Mitul, M.T., Benchorin, A., Batucal, K., Daugherty, A., et al. (2024). Tissue determinants of the human T cell receptor repertoire. Preprint at bioRxiv. https://doi.org/10.1101/2024.08.17.608295.

60. Bloom, J.D., Labthavikul, S.T., Otey, C.R., and Arnold, F.H. (2006). Protein stability promotes evolvability. Proc. Natl. Acad. Sci. USA *103*, 5869–5874. https://doi.org/10.1073/pnas.0510098103.

61. Weigt, M., White, R.A., Szurmant, H., Hoch, J.A., and Hwa, T. (2009). Identification of direct residue contacts in protein–protein interaction by message passing. Proc. Natl. Acad. Sci. USA *106*, 67–72. https://doi.org/10.1073/pnas.0805923106.

62. Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. (2021). Highly accurate protein structure prediction with alphafold. Nature *596*, 583–589. https://doi.org/10.1038/s41586-021-03819-2.

63. Mayer, A., Balasubramanian, V., Mora, T., and Walczak, A.M. (2015). How a well-adapted immune system is organized. Proc. Natl. Acad. Sci. USA *112*, 5950–5955. https://doi.org/10.1073/pnas.1421827112.

64. Thomas, P.G., and Crawford, J.C. (2019). Selected before selection: a case for inherent antigen bias in the T-cell receptor repertoire. Curr. Opin. Syst. Biol. *18*, 36–43. https://doi.org/10.1016/j.coisb.2019.10.007.

65. Elhanati, Y., Murugan, A., Callan, C.G., Mora, T., and Walczak, A.M. (2014). Quantifying selection in immune receptor repertoires. Proc. Natl. Acad. Sci. USA *111*, 9875–9880. https://doi.org/10.1073/pnas.1409572111.

66. Dens, C., Laukens, K., Bittremieux, W., and Meysman, P. (2023). The pitfalls of negative data bias for the T-cell epitope specificity challenge. Nat. Mach. Intell. *5*, 1060–1062. https://doi.org/10.1038/s42256-023-00727-0.

67. Schroff, F., Kalenichenko, D., and Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 815–823. https://doi.org/10.1109/CVPR.2015.7298682.

68. Liu, W., Wen, Y., Yu, Z., Li, M., Raj, B., and Song, L. (2017). SphereFace: deep hypersphere embedding for face recognition. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6738–6746. https://doi.org/10.1109/CVPR.2017.713.

69. Chen, S., Liu, Y., Gao, X., and Han, Z. (2018). MobileFaceNets: efficient CNNs for Accurate Real-Time Face Verification on Mobile Devices. Preprint at arXiv. https://doi.org/10.48550/arXiv.1804.07573.

70. Deng, J., Guo, J., Yang, J., Xue, N., Kotsia, I., and Zafeiriou, S. (2022). ArcFace: additive angular margin loss for deep face recognition. In IEEE Trans. Pattern Anal. Mach. Intell., *44*, pp. 5962–5979. https://doi.org/10.1109/TPAMI.2021.3087709.

71. Nagano, Y., and Chain, B. (2023). tidytcells: standardizer for TR/MH nomenclature. Front. Immunol. *14*, 1276106. https://doi.org/10.3389/fimmu.2023.1276106.

72. Heather, J.M., Spindler, M.J., Alonso, M.H., Shui, Y.I., Millar, D.G., Johnson, D.S., Cobbold, M., and Hata, A.N. (2022). Stitchr: stitching coding TCR nucleotide sequences from V/J/ CDR3 information. Nucleic Acids Res. *50*, e68. https://doi.org/10.1093/nar/gkac190.

73. Kingma, D.P., and Ba, J. (2017). Adam: A Method for Stochastic Optimization. Preprint at arXiv. https://doi.org/10.48550/arXiv.1412.6980.

STAR★METHODS

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Deposited data		
TCR-pMHC experimental data	VDJdb[28]	https://github.com/antigenomics/vdjdb-db/releases/2023-06-01/
Paired chain alpha-beta TCRs from blood samples	Tanno et al.[51]	https://doi.org/10.5281/zenodo.14288119
Software and algorithms		
SCEPTR – model release v1.1.0	This study	https://doi.org/10.5281/zenodo.14286003
tcrlm – inference and testing of TCR language models v1.0.0	This study	https://doi.org/10.5281/zenodo.14285951
libtcrlm – TCR language model library v1.0.0	This study	https://doi.org/10.5281/zenodo.14286010
Figure generation and benchmarking code	This study	https://doi.org/10.5281/zenodo.14285924
Tidytcells v2.1	Nagano and Chain[71]	https://doi.org/10.5281/zenodo.14286097
Pyrepseq v1.4.3	Mayer and Callan[44]	https://doi.org/10.5281/zenodo.14237373
Stitchr v1.1.3.1	Heather et al. [72]	https://doi.org/10.1093/nar/gkac190
tcrdist3 v0.2.2	Mayer-Blackwell et al.[43]	https://github.com/kmayerb/tcrdist3
ESM2 (T6 8M)	Lin et al.[33]	https://huggingface.co/facebook/esm2_t6_8M_UR50D
TCR-BERT	Wu et al.[11]	https://huggingface.co/wukevin/tcr-bert
ProtBert	Elnaggar et al.[34]	https://huggingface.co/Rostlab/prot_bert

METHOD DETAILS

SCEPTR architecture

SCEPTR is a BERT-like transformer encoder that maps TCR sequences to vector embeddings. Like BERT, it is comprised of a tokeniser module, an embedder module and a self-attention stack (Figure 2A).

The tokeniser module represents each input TCR as the amino acid sequences of the first, second and third complementarity-determining regions (CDRs) of each chain, where each amino acid is a token. A special <cls> token is appended to each input TCR, as its contextualised embedding will eventually become SCEPTR's output representation vector (Figures 2A and 2B).

SCEPTR uses a simple, non-trainable embedder module, where a one-hot vector is used to encode token identity (22 dimensions for 20 amino acids plus special tokens <cls> and <mask>), and token positions are specified by first one-hot encoding the containing CDR loop number (6 dimensions), then encoding the token's relative position within the loop as a single scalar variable (Figure 2B). This results in initial token embeddings in $\mathbb{R}^{29}$, which are passed through a trainable linear projection onto $\mathbb{R}^{64}$. SCEPTR's self-attention stack (Figure S16) then operates at this fixed dimensionality. SCEPTR's self-attention stack comprises three layers, each with eight attention heads and a feed-forward dimensionality of 256, and is thus substantially simpler than existing models. Our tests suggest that relative position embedding helps SCEPTR learn better calibrated TCR co-specificity rules (see Figure S14).

SCEPTR Pre-training

The unlabelled paired-chain $\alpha\beta$ TCR sequences used to pre-train SCEPTR were taken from a study by Tanno et al.,[51] which provides 965,523 unique clonotypes sampled from the blood of 15 healthy human subjects. As opposed to traditional single-cell sequencing, Tanno et al. used a ligation-based sequencing method to resolve which α chains paired with which β chains. To mitigate potential noise from incorrect chain pairing, we applied an extra processing step to remove clonotypes that shared the same nucleotide sequence for either the α or the β chain, as previously described.[44] After filtering for functional TCRs using tidytcells, a TCR gene symbol standardiser,[71] we retained 842,683 distinct clonotypes.

A random sub-sample of 10% of this data was reserved for use as an unseen test set, containing 84,268 unique clonotypes distributed across 83,979 unique TCRs. Of the remaining 90% of the data, we filtered out any clonotypes with amino acid sequences that also appeared in the test set, resulting in a training set of 753,838 unique clonotypes across 733,070 unique TCRs.

Using this data, SCEPTR was jointly optimised for MLM and autocontrastive learning, where the total loss of a training step was calculated as the sum of the MLM and autocontrastive (Equation 4) losses.

We implemented MLM following established procedures.[30] Namely, 15% of input tokens were masked, and masked tokens had an 80% probability of being replaced with the <mask> token, a 10% probability of being replaced by a randomly chosen amino acid distinct from the original, and a 10% probability of remaining unchanged. The MLM loss was computed as the cross-entropy between SCEPTR's predicted token probability distribution and the ground truth.

Our choice of autocontrastive loss function is inspired by related work in NLP[52] and computer vision,[48] but adapted to the TCR setting. Let $B = \{\sigma_i\}_{i=1}^{N}$ be a minibatch of N TCRs. We generate two independent "views" of each TCR σ_i by passing two censored-variants of the same receptor through the model. Our censoring procedure removes a random subset of a fixed proportion (20%) of the residues from the tokenised representation of the CDR loops and with a 50% chance drops either the full α or β chain. To ensure that censoring does not fundamentally alter the underlying TCR sequence, the positional encoding for each token remains fixed relative to the original TCR. In addition to the random censoring, views also differ due to dropout noise during independent model passes. Taken together, this procedure maps the minibatch B to the set of $2N$ TCR views $V = \{v_j\}_{j=1}^{2N}$, where v_{2i} and v_{2i-1} are two independent views of the same TCR σ_i ($i \in \{1 \ldots N\}$). Where $k \in K = \{1 \ldots 2N\}$ is an arbitrary index of a view $v_k \in V$, let $p(k)$ be the index of the other view generated from the same TCR, and $A(k) = \{l \in K : l \neq k\}$ be the set of all indices apart from k itself. Let $\boldsymbol{r}_k$ denote SCEPTR's vector representation of TCR view v_k. Then the autocontrastive loss for minibatch B is computed as follows:

$$\mathcal{L}_{\mathrm{AC}}(B) = \frac{1}{2N}\sum_{k \epsilon K} -\log \frac{e^{\boldsymbol{r}_k \boldsymbol{r}_{p(k)}/\tau}}{\sum_{n \epsilon A(k)} e^{\boldsymbol{r}_k \boldsymbol{r}_n/\tau}} \quad \text{(Equation 4)}$$

Here, τ is a temperature hyper-parameter which we set to 0.05 during training, following previous literature.[52]

We used ADAM (adaptive moment estimation)[73] to perform stochastic gradient descent. We chose a minibatch size of 1024 samples and trained for 200 epochs, which equated to 143,200 training steps. The internal dropout noise of SCEPTR's self-attention stack was set to 0.1.

Our methodology of randomly censoring residues and even entire chains stands in contrast to previous work in NLP by Gao et al.,[52] who found that relying only on the internal random drop-out noise of the language model was sufficient for effective autoconstrastive learning. However, our experiments suggest that in the TCR domain, residue and chain censoring leads to embeddings with better downstream TCR specificity prediction performance (Figure S5).

Fine-tuning by supervised contrastive learning

For supervised contrastive fine-tuning we took all TCR binders against the six best-sampled pMHC targets from our labelled TCR dataset, and split them into a training, a validation, and a test set such that no study used to construct the training or validation sets contributed any TCRs to the test set (Table S2).

Using this data, the fine-tuning process involved the joint optimisation of SCEPTR on MLM and supervised contrastive learning. As during pre-training, the overall loss for each training step was computed as the unweighted sum of the MLM and supervised contrastive (Equation 5) losses. The pre-trained state of SCEPTR was used as the starting point for fine-tuning. With only 200 TCRs for each target pMHC to train on, we limited the number of learnable parameters by only allowing the weights of the final self-attention layer to be trainable. Additionally, we monitored increases in validation loss for early stopping of fine-tuning, which occurred after 2 epochs, where one epoch is defined as the model seeing 100,000 binders for each pMHC. Given our a batch size of 1,024 TCRs, this corresponded to a total of 1,172 training steps.

Our implementation of supervised contrastive learning closely follows the formulation suggested by Khosla et al.[48] This approach to supervised contrastive learning combines loss contributions from true positive pairs, with those from second views of each positive instance (as in autocontrastive learning) as well as all views of all other sample points with the same pMHC label. Let $B = \{\sigma_i\}_{i=1}^{N}$ be a minibatch of N pMHC-annotated TCRs. We use the same procedure as in our autocontrastive framework (see previous methods section) to generate two views of each of the TCRs, producing a set of $2N$ views $V = \{v_j\}_{j=1}^{2N}$. Let $Y = \{y_i\}_{i=1}^{N}$ be the index-matched pMHC labels for TCRs in B, and $\bar{y}_j$ denote the labels mapped to the indices of the views in V such that $\bar{y}_{2i} = \bar{y}_{2i-1} = y_i$. Now given arbitrary sample view index k, let $P(k) = \{l \in A(k) : \bar{y}_l = \bar{y}_k\}$ be the set of all indices other than k itself whose corresponding samples have the same pMHC label as v_k, with cardinality $|P(k)|$. The supervised contrastive loss for TCR minibatch B is:

$$\mathcal{L}_{\mathrm{SC}}(B) = \frac{1}{2N}\sum_{k \epsilon K}\frac{1}{|P(k)|}\sum_{p \epsilon P(k)} -\log \frac{e^{\boldsymbol{r}_k \boldsymbol{r}_p/\tau}}{\sum_{n \epsilon A(k)} e^{\boldsymbol{r}_k \boldsymbol{r}_n/\tau}} \quad \text{(Equation 5)}$$

Each batch during fine-tuning has an equally balanced number of binders to each of the six pMHCs.

Generating TCR vector embeddings using existing protein language models

The TCR-BERT model was downloaded through HuggingFace at https://huggingface.co/wukevin/tcr-bert. Since TCR-BERT is trained to read one CDR3 sequence at a time, we generated TCR representations by generating two independent representations of the α and β chain, and concatenating them together. The TCR-BERT representation of a chain was generated by feeding the model its CDR3 sequence, then taking the average pool of the amino acid token embeddings in the 8th self-attention layer, as recommended by the study authors.[11]

The ESM2 (T6 8M) model was downloaded through HuggingFace at https://huggingface.co/facebook/esm2_t6_8M_UR50D. ESM2 is trained on full protein sequences, but not protein multimers. Therefore, we generated ESM2 representations for the α and β chains separately, and concatenated them to produce the overall TCR representation. To generate the representation of a TCR chain, we first used Stitchr[72] to reconstruct the full amino acid sequence of a TCR from its CDR3 sequence and V/J gene. Then, the resulting sequence of each full chain was fed to ESM2. We took the average-pooled result of the amino acid token embeddings of the final layer to generate the overall sequence representation, as recommended.[33]

The ProtBert model was downloaded through HuggingFace at https://huggingface.co/Rostlab/prot_bert. Similarly to ESM2, ProtBert is trained on full protein sequences. Therefore, we again used Stitchr to generate full TCR chain amino acid sequences, and fed them to ProtBert to generate independent α and β chain representations. We again as recommended average-pooled the amino acid token embeddings of the final layer.[34]

Note that while SCEPTR's architecture and training allows it to directly generate representation vectors for complete $\alpha\beta$ TCR sequences (Figure 2A), this is not the case for the other PLMs. For TCR-BERT, ESM2 and ProtBert representations for the α and β chains were independently generated, then concatenated together, and finally average-pooled to produce an embedding of the heterodimeric receptor. Similarly, in the CDR3 Levenshtein model we compute the distance between two TCRs as the sum of the Levenshtein distances between the receptors' α and β CDR3s.

Learning features within embedding spaces

The focus of the current work has been to use nearest neighbour prediction using PLM embeddings as the most direct test of data-efficient transfer learning that works with as little as a single reference sequence. If slightly more data is available, another approach is to train supervised predictors atop PLM embeddings. To test how much such training can improve prediction performance, we trained linear support vector classifiers (SVC) on the PLM embeddings provided by different models. In each instance, we trained the classifier to distinguish reference TCRs from 1000 randomly sampled background TCRs. We outline the methodology in more detail below.

We find that the SVC predictors for ProtBert, ESM2 and TCR-BERT all perform better than their nearest neighbour counterparts, but still worse than SCEPTR's nearest neighbour predictions (Figure S4). We also trained an SVC atop SCEPTR, which did not lead to further improvement upon the nearest neighbour prediction (Figure S4). These findings highlight how in the low data regime typical of most pMHCs, misalignment of pre-training to downstream tasks can only be partially remediated by training on reference TCRs.

To train the linear SVCs on top of PLM features, we sampled 1000 random background TCRs from the training partition of the unlabelled Tanno et al. dataset. We employed a similar strategy to our benchmarking in benchmarking PLM embeddings on TCR specificity prediction to split our dataset of curated specificity-annotated $\alpha\beta$ TCRs into a reference set and testing set. For each PLM-pMHC-split combination, we trained a linear SVC using the PLM embeddings of the reference TCRs as the positives and those of the 1000 background TCRs as the negatives. The same 1000 background TCRs were used across model-pMHC-split combinations to ensure consistency. We accounted for the imbalance between the number of positive and negative samples used during SVC fitting by weighting the penalty contributions accordingly. Finally, we tested SVCs using the same benchmarking classification task as previously described.

Effects of position embedding methods

To better understand TCR similarity rules as learned by PLMs, we measured the average distance penalty incurred within a model's representation space as a result of a single amino acid edit at various points along the length of the α/β CDR3 loops. To do this, we randomly sampled real TCRs from the testing partition of the Tanno et al. dataset[51] and synthetically introduced single residue edits in one of their CDR3 loops. Then, we measured the distance between the original TCR and the single edit variant according to a PLM. For each model, we sampled TCRs until we had observed at least 100 cases of: 1) each type of edit (insertions, deletions, substitutions) at each position, and 2) substitutions from each amino acid to every other. Since CDR3 sequences vary in length, we categorised the edit locations into one of five bins: C-TERM for edits within the first one-fifth of the CDR3 sequence counting from the C-terminus, then M1, M2, M3, and N-TERM, in that order. For this analysis, we investigated SCEPTR and TCR-BERT, since they are the two best performers out of the PLMs tested (Figure 1D).

Both SCEPTR and TCR-BERT generally associate insertions and deletions (indels) with a higher distance penalty compared to substitutions (Figure S14A). While SCEPTR uniformly penalises indels across the length of the CDR3, TCR-BERT assigns higher penalties to those closer to the C-terminus. We hypothesised that the variation in TCR-BERT's indel penalties is a side-effect of its position embedding system. TCR-BERT, like many other transformers, encodes a token's position into its initial embedding in a left-aligned manner using a stack of sinusoidal functions with varying periods[11,29,30] This results in embeddings that are more sensitive to indels near the C-terminus, which cause a frame-shift in a larger portion of the CDR3 loop and thus lead to a larger change in the model's underlying TCR representation. To test this hypothesis, we trained and evaluated a new SCEPTR variant:

SCEPTR (left-aligned)

This variant uses a traditional transformer embedding system with trainable token representations and left aligned, stacked sinusoidal position embeddings.

While we detect no significant difference in downstream performance between SCEPTR and its left-aligned variant (Figure S15), this may be because cases where the differences in their learned rule sets affects performance are rarely seen in our benchmarking data. The edit penalty profile of the left-aligned variant shows a similar falloff of indel penalties than TCR-BERT with higher penalties at

the C than N-terminals (Figure S14B). As their is no clear biological rationale for this observation, these results suggest that SCEPTR's relative position encoding might result in a better-calibrated co-specificity ruleset. These preliminary findings add to the ongoing discussion around how to best encode residue position information in the protein language modelling domain.[35]

Our results show that the penalty falloff seen with SCEPTR (left-aligned) is sharper than that of TCR-BERT, whose indel penalties plateau past M1. As TCR-BERT is a substantially deeper model (12 self-attention layers, 12 heads each, embedding dimensionality 768), it might be partially able to internally un-learn the left-aligned-ness of the position information. If this is true, then position embedding choices are particularly important for training smaller, more efficient models.

QUANTIFICATION AND STATISTICAL ANALYSIS

For each pMHC, we varied the number k of reference TCRs where $k \in \{1,2,5,10,20,50,100,200\}$. Within each model-pMHC-k-shot combination, we benchmarked multiple reference-test splits of the data to ensure robustness. For $k = 1$, we benchmarked every possible split. For $k \in [2, 200]$, we benchmarked 100 random splits, where we ensured that the same splits were used across all models to reduce extraneous variance.

In assessing the statistical significance of differences in average model performance, we took a paired difference approach. We expected certain pMHCs and data splits to present a more difficult prediction problem than others. As we are interested in assessing the relative performance of models, we calculated the variance across splits of the difference between each individual model's AUROC and the average across all models. For each model, we estimated this variance within each of the pMHCs, and then averaged these variances to obtain an estimate of overall variance.

ADDITIONAL RESOURCES

https://github.com/yutanagano/sceptr a readily usable deployment of SCEPTR and its variants.
https://github.com/yutanagano/tcrlm houses code used for designing and training our models.
https://github.com/yutanagano/libtcrlm library code powering the above repositories.